



**EXPERIMENTATION AND VALIDATION OPENNESS FOR LONGTERM
EVOLUTION OF VERTICAL INDUSTRIES IN 5G ERA AND BEYOND**

[H2020 - Grant Agreement No.101016608]

Deliverable D4.3

5G Exposure Capabilities for Vertical Applications (Final)

Editor G. Makropoulos (NCSRDI)

Contributors (TID), (NCSRDI), (MAG), (ATOS), (INTRA), (LNV), (FOGUS)

Version 1.1

Date August 31st, 2023

Distribution PUBLIC (PU)





DISCLAIMER

This document contains information, which is proprietary to the EVOLVED-5G ("Experimentation and Validation Openness for Longterm evolution of VERTICAL inDUSTRIES in 5G era and beyond) Consortium that is subject to the rights and obligations and to the terms and conditions applicable to the Grant Agreement number: 101016608. The action of the EVOLVED-5G Consortium is funded by the European Commission.

Neither this document nor the information contained herein shall be used, copied, duplicated, reproduced, modified, or communicated by any means to any third party, in whole or in parts, except with prior written consent of the EVOLVED-5G Consortium. In such case, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced. In the event of infringement, the consortium reserves the right to take any legal action it deems appropriate.

This document reflects only the authors' view and does not necessarily reflect the view of the European Commission. Neither the EVOLVED-5G Consortium as a whole, nor a certain party of the EVOLVED-5G Consortium warrant that the information contained in this document is suitable for use, nor that the use of the information is accurate or free from risk, and accepts no liability for loss or damage suffered by any person using this information.

The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.



REVISION HISTORY

Revision	Date	Responsible	Comment
0.3	June, 2023		ToC Finalisation
0.5	June, 26, 2023		Initial Contributions
0.7	July, 8, 2023		1 st Draft
0.8	July, 17, 2023		2 nd Draft
0.9	July, 25, 2023		Full Draft
1.0	August, 7, 2023		Internal review
1.1	August 28 th 2023		Final version

LIST OF AUTHORS

<i>Partner ACRONYM</i>	<i>Partner FULL NAME</i>	<i>Name & Surname</i>
TID	TELEFONICA INVESTIGACIÓN Y DESARROLLO	Javier Garcia, David Artunedo
NCSR	NATIONAL CENTER FOR SCIENTIFIC RESEARCH DEMOKRITOS	George Makropoulos, Harilaos Koumaras, Dimitrios Fragkos, Anastasios Gogos, N. Dimitriou, K. Kontovassilis, E. Charou
ATOS	ATOS IT SOLUTIONS AND SERVICES IBERIA SL	Ricardo Marco
INTRA	NETCOMPANY INTRASOFT SA	Angela Dimitirou
LNV	LENOVO (Deutschland) GmbH	Apostolis Salkintzis, Dimitris Dimopoulos
FOG	FOGUS INNOVATIONS & SERVICES P.C.	Dimitris Tsolkas
UMA	UNIVERSITY OF MÁLAGA	Bruno García

GLOSSARY

<i>Abbreviations/Acronym</i>	<i>Description</i>
3GPP	<i>3rd Generation Partnership Project</i>
5GC	<i>5G Core</i>
5GS	<i>5G System</i>
AF	<i>Application Function</i>
API	<i>Application Programming Interface</i>
CAPIF	<i>Common API Framework</i>
CI/CD	<i>Continuous Integration / Continuous Development</i>
CLI	<i>Command Line Interface</i>
DS-TT	<i>Device Side Translator</i>
FoF	<i>Factory of the Future</i>
GBR	<i>Guaranteed Bit Rate</i>
GW	<i>Gateway</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IIoT	<i>Industrial Internet of Things</i>
IP	<i>Internet Protocol</i>
MDBV	<i>Maximum Data Burst volume</i>
NaaS	<i>Network as a Service</i>
NON-GBR	<i>Non-Guaranteed Bit Rate</i>
NPN	<i>Non-Public Network</i>
NSA	<i>Non-Standalone</i>
NWDAF	<i>Network Data Analytics Function</i>
NW-TT	<i>Network Side Translator</i>
PCC	<i>Policy Control Charging</i>
PCF	<i>Policy Control Function</i>
PDB	<i>Packet Delay Budget</i>
PDU	<i>Protocol Data Unit</i>
PTP	<i>Precision Time Protocol</i>
QoS	<i>Quality of Service</i>
RAN	<i>Radio Access Network</i>
REST	<i>Representational State Transfer</i>
SA	<i>Stand Alone</i>
SDK	<i>Software Development Kit</i>
SDM	<i>Subscriber Data Management</i>
SEAL	<i>Service Enabler Architecture Layer</i>
SMF	<i>Session Management Function</i>
TSCF	<i>Time Synchronization Function</i>
TSN	<i>Time Sensitive Networking</i>



UDM *Unified Data Management*

UE *User Equipment*

V2X *Vehicle to Everything*

vAPP *Vertical Application*

EXECUTIVE SUMMARY

The deliverable is the result of the work carried out in Task 4.1 “5G Exposure Capabilities for Vertical Applications Development” and it also touches Tasks T4.2-T4.5 (Network Applications’ development tasks). As such, key contribution of the deliverable is the description of the work that has been performed during the second period of the project, towards the definition and updated design of the northbound APIs and the exploration of additional APIs that could hold significance and relevance throughout the lifecycle of the Network Applications (Network Apps).

An important aspect defined in the project is the integration activities with the main goal of demonstrating the functionality of the Network App when seamlessly integrated with the Vertical Application (vApp) and testing each use case overall. Therefore, integration tests were planned and executed for the EVOLVED-5G Network Apps and their related vApps. These integration efforts utilized the final version of the components developed in EVOLVED-5G and ensured a holistic approach and coherent collaboration between Network Apps and vApps, providing a comprehensive and robust solution for various use cases within the EVOLVED-5G ecosystem.

As a final point, in the context of EVOLVED-5G, it is essential to highlight that a terminology update has been implemented. Specifically, the term "Network App" is now being used instead of "NetApp," as initially selected in the first period of the project. This update reflects the shortened form of "Network Application" and has been applied consistently across all project’s documents and materials.

TABLE OF CONTENTS

1	INTRODUCTION.....	1
1.1	Purpose of the document.....	1
1.2	Structure of the document.....	1
1.3	Target Audience.....	1
2	5G Northbound APIs and Information Flows.....	3
2.1	Network Exposure Function (NEF).....	3
2.1.1	NEF/Location Reporting Event.....	3
2.1.2	NEF/Loss of Connectivity Event.....	5
2.1.3	NEF/AsSessionWithQoS.....	6
2.2	NWDAF/AF Event Exposure Service.....	7
2.3	TSN APIs.....	9
2.4	Seal Apis.....	10
2.5	CAMARA APIs.....	11
3	Implementation of 5G Northbound APIs in EVOLVED-5G.....	14
3.1	NEF APIs.....	14
3.2	TSN APIs.....	20
3.3	NWDAF.....	21
3.4	Integration of Northbound 5GS APIs with Capif.....	22
3.4.1	CAPIF APIs.....	23
3.4.2	NEF INTEGRATION WITH CAPIF.....	24
3.4.3	TSN INTEGRATION WITH CAPIF.....	25
4	DEVELOPMENT TOOLS AND OVERALL FRAMEWORK.....	27
4.1	Development Tools.....	27
4.1.1	Updates on The Development Tools.....	27
4.2	Updates on The Verification Tests.....	30
4.3	Evolution of the Network Apps.....	31
5	Vertical Application and Network Application Integration Trials.....	35
5.1	K8s Cluster Deployment of the Second Integration Round.....	38
5.2	Consultation and Guidelines to the SMEs.....	40
6	CONCLUSION.....	42
7	References.....	43

LIST OF FIGURES

Figure 1 Information flow for the Location reporting event.....	4
Figure 2 Information flow for the Monitoring event API	5
Figure 3 Information flow for the AsSessionwithQoS	6
Figure 4 API calls enabling the interaction between NWDAF and the AF.....	8
Figure 5 VAL and SEAL generic functional model.....	10
Figure 6 CAMARA architectural framework	11
Figure 7 CAPIF as reference API GW for CAMARA	13
Figure 8 MonitoringEvent API - Loss of connectivity subscription.....	15
Figure 9 MonitoringEvent API - UE reachability subscription	15
Figure 10 Network App acts as a TCP stats collector providing inputs to NWDAF.....	21
Figure 11 Abstract illustration of the CAPIF functionality, with representative examples of its applicability	22
Figure 12 NEF exposure capabilities to Networks Apps through CAPIF.....	25
Figure 13 API Invoker, TSN FrontEnd and CAPIF interaction	26
Figure 14 Network Applications classification	34
Figure 15 Timeline of first and second round integration activities between vApp and Networks Apps.....	35
Figure 16 K8s cluster Architecture in Athens Platform	39
Figure 17 K8s cluster Architecture in Malaga Platform	40



LIST OF TABLES

Table 1 APIs used in the Location reporting event	4
Table 2 APIs used in the Loss of Connectivity event	6
Table 3 Available services for Monitoring Event API	16
Table 4 Available services for AsSessionWithQos API.....	18
Table 5 TSN Frontend API endpoints	20
Table 6 Available Northbound APIs for the CAPIF services.....	23
Table 7 Versioning of the EVOLVED-5G components	29
Table 8 Evolution and versions of the Network Apps	31
Table 9 Type of Network Apps and API(s) used per-SME	32
Table 10 Differences between the 2 rounds of integration testing.....	36

1 INTRODUCTION

1.1 PURPOSE OF THE DOCUMENT

The current document, titled "5G Exposure Capabilities for Vertical Applications," represents the updated and final version of the document D4.1. This revised version incorporates the enhancements on the Application Programmable Interfaces (APIs) defined by the 3rd Generation Partnership Project (3GPP) and implemented utilising the Network Exposure Function (NEF) emulator through CAPIF. Moreover, the document highlights the research work carried out by the collaborating partners in Task 4.1, aimed at exploring additional APIs that could hold significance and relevance throughout the lifecycle of the Network Apps. It also takes into account the latest enhancements and updates that have occurred on top of EVOLVED-5G development tools, as initially described in D3.3.

1.2 STRUCTURE OF THE DOCUMENT

The core part of the document is divided into the following sections:

- Section 2 "5G Northbound APIs and Information flows" outlines the prioritized APIs, which EVOLVED-5G has emphasized for the development of the final prototype of the Network Applications, as well as the other activities within WP4.
- Section 3 "Implementation of 5G Northbound APIs in the EVOLVED-5G" focuses on the implementation's aspects of the APIs described in the previous section.
- Section 4 "Development tools and overall framework" focuses on presenting and highlighting the main updated of the development and verification tools within the Workspace of EVOLVED-5G.
- Section 5 "Vertical Application and Network Application Integration trials" comprises the two rounds of integration activities in the EVOLVED-5G platforms (Athens and Malaga) performed to demonstrate the functionality of the Network applications when seamlessly integrated with the vertical Applications.

1.3 TARGET AUDIENCE

The release of the deliverable is public, intending to expose the overall EVOLVED-5G ecosystem and Network Apps Lifecycle design to a wide variety of research individuals and communities.

From specific to broader, different target audiences for D4.3 are identified as detailed below:

- **Project Consortium:** To validate that all objectives and proposed technological advancements have been analysed and to ensure that, through the proposed NetApp Lifecycle phases and the various Environments, further work can be concretely derived. Furthermore, the deliverable sets to establish a common understanding among the consortium with regards to:
 - The CI/CD approach for the software using the open workspace, developed in WP3, leading to the development of the Network Apps
 - The actual development of the Network Apps is categorized in four different Industry 4.0 domains following the requirements and specifications introduced in Task 2.2 and Task 3.1.

- **Industry 4.0 and FoF (factories of the future) vertical groups:** To crystallise a common understanding of technologies, and design principles that underline the development of the Network Apps, and to understand the utilisation of the network APIs exposed by the 5G Infrastructure. A non-exhaustive list of Industry 4.0-related groups is as follows:
 - Manufacturing industries (including both large and SMEs) and IIoT (Industrial Internet of Things) technology providers.
 - European, national, and regional manufacturing initiatives, including funding programs, 5G-related research projects, public bodies and policy makers.
 - Technology transfer organizations and market-uptake experts, researchers, and individuals.
 - Standardisation Bodies and Open-Source Communities.
 - Industry 4.0 professionals and researchers with technical knowledge and expertise, who have an industrial professional background and work on industry 4.0-related areas.
 - Industry 4.0 Investors and business angels.
- **Telecom Service Providers:** to engage with verticals and to simplify the way 5G services can be offered to a potential customer or 3rd party service provider.
- **Other vertical industries and groups:** To seek impact on other 5G-enabled vertical industries and groups in the long run. Indeed, all the architectural components of the facility are designed to secure interoperability beyond vendor specific implementation and across multiple domains. The same categorization as the above but beyond Industry 4.0 can be of application.
- **The scientific audience, general public and the funding EC Organisation:** To document the work performed and justify the effort reported for the relevant activities. The scientific audience can also get an insight of the design of the business APIs as well as the Network Apps' development process and functionalities.

2 5G NORTHBOUND APIS AND INFORMATION FLOWS

Within 5GS, different 5G network functions are appropriate for configuring a particular monitoring event, detecting the event, and reporting to an authorized external application. UE's mobility management context, including UE position, reachability, roaming status, and loss of connectivity, can be exposed using the monitoring feature. As a result, mapping the information flows to the SBIs/APIs implies a thorough examination of the relevant 3GPP specifications. The following section presents and describes the information flows, specifically the control plane signaling of the Northbound APIs, which have been identified as valuable and prioritized within the EVOLVED-5G project. This prioritization is based on a comprehensive requirement analysis conducted during the initial reporting period, involving the eleven SMEs participating in the consortium. Notably, this analysis revealed a strong preference for APIs offering Location Reporting Events, Loss of Connectivity Events, and AsSessionWithQoS functionality. Additionally, Time Sensitive Network (TSN) control and the Network Data Analytics Function (NWDAF) were also chosen due to their significant interest among Network App developers of the corresponding SMEs. The following sub-sections provide a briefing on these prioritized APIs, which EVOLVED-5G has emphasized for its activities in WP4, along with the realization of prototype Network Apps.

2.1 NETWORK EXPOSURE FUNCTION (NEF)

2.1.1 NEF/Location Reporting Event

The event is detected based on the event reporting information parameters (i.e., LOCATION_REPORTING) received in the Monitoring request. These parameters include details like one-time reporting, the maximum number of reports, maximum reporting duration.

Location Reporting Options:

- **One-time Reporting:** The event reporting provides either the current location or the last known location of the User Equipment (UE). If the immediate reporting flag is set, Access and Mobility Management Function (AMF) reports the Last Known Location immediately. If the immediate reporting flag is not set, AMF reports the UE's current location. If AMF does not have the current location at the requested granularity, it retrieves the information via New Generation-Radio Access Network (NG-RAN) Location reporting procedure
- **Continuous Location Reporting:** For continuous reporting, the serving node sends notifications every time a location change is detected, with the granularity depending on the accepted accuracy of location. In the context of the project, the granularity is per cell level.

The location request precision is lower or equal to cell level, thus Network App (i.e., AF) is notified for location updates triggered when a specific UE changes cell. Figure 1 Information flow for the Location reporting event below, is conducted by the overall information flow described in TS 23.502 [1]. The NFs being involved in the process are Application Function (AF), Network Exposure Function (NEF), Unified Data Management (UDM) and AMF. Moreover, the APIs for the interaction between NFs are identified (adapted from TS 29.522 [2]).

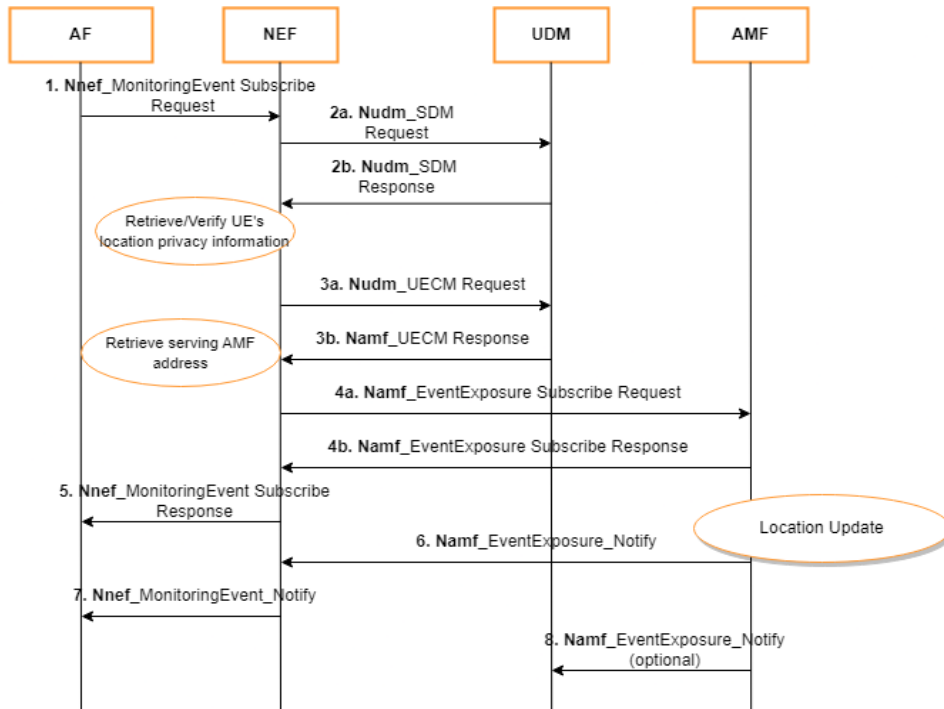


Figure 1 Information flow for the Location reporting event

Steps:

1. AF requests a subscription through Monitoring Event API provided by NEF for LOCATION_REPORTING events.
2. NEF retrieves the UE location privacy information from the UDM using the **Nudm_SDM** service (for more details see TS 29.503 [3]).
3. If the privacy setting is verified, then NEF interacts again with UDM to discover the serving AMF address by invoking the **Nudm_UECM** service.
4. After receiving the service AMF address, NEF interacts with the AMF and subscribes for the location_report event through **Namf_EventExposure** service.
5. NEF informs the AF that the subscription was successfully created
6. When the event occurs in the 5GS the AMF informs the NEF with a callback notification
7. NEF forwards the notification to the AF
8. (Optional) If the AMF detects a subscription change related to the LOCATION_REPORTING event, it sends the event report, by means of Namf_EventExposure_Notify message to the UDM.

In the Table below, a summary of the APIs used in the above procedure is presented.

Table 1 APIs used in the Location reporting event

NF provider	NF consumer	Service	API	3GPP Document / GitHub link
NEF	AF	MonitoringEvent API	POST /{scsAsID}/subscriptions	TS 29.122/29.522 – link
UDM	NEF	SDM API	GET /{ueId}/lcs-privacy-data	TS 29.503 – link
UDM	NEF	UECM API	GET /{ueId}/registrations/amf-3gpp-access	TS 29.503 – link

			or GET /{ueId}/registrations	
AMF	NEF	EventExposure API	POST /subscriptions	TS 29.518 – link

2.1.2 NEF/Loss of Connectivity Event

The network detects when the UE is no longer reachable for signaling or user plane communication. The AF can specify a Maximum Detection Time, indicating the maximum duration without communication with the UE after which the AF will be notified that the UE is considered unreachable. Figure 2 Information flow for the Monitoring event API below, is conducted by the overall information flow described in TS 23.502 [1] (see Information flows). The NFs being involved in the process are AF, NEF, UDM and AMF. Moreover, the APIs for the interaction between NFs are identified (adapted from TS 29.522 [2]).

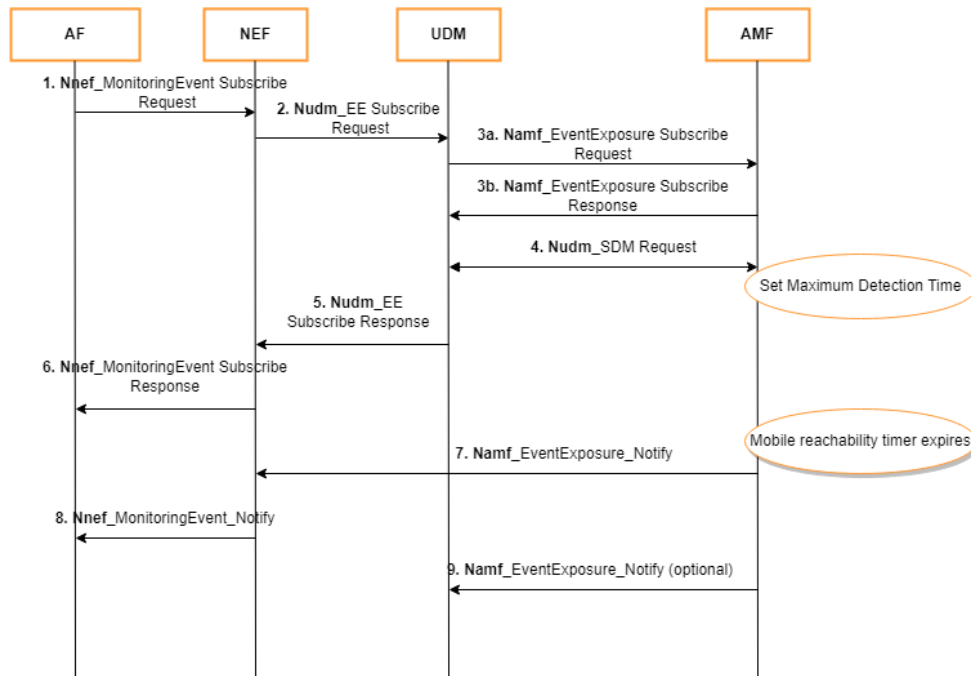


Figure 2 Information flow for the Monitoring event API

Steps:

1. AF requests a subscription through Monitoring Event API provided by NEF for LOSS_OF_CONNECTIVITY events.
2. NEF subscribes for Event exposure in UDM using the **Nudm_EE** service (for more details see TS 29.503 [4]).
3. Since the requested event (e.g., monitoring of Loss of Connectivity) requires AMF assistance, then the UDM creates the **Namf_EventExposure** subscription to the AMF serving the requested user.
4. The UDM shall set the subscribed periodic registration timer using the Maximum Detection Time (provided by AF in step 1). Then UDM notifies AMF about the new periodic registration timer (i.e., the assumption here is that AMF has already subscribed to the UDM using the SDM service).

5. UDM acknowledges NEF with response
6. NEF acknowledges AF with response
7. When the event occurs in the 5GS the AMF informs the NEF with a callback notification
8. NEF forwards the notification to the AF
9. (Optional) If the AMF detects a subscription change related to the LOSS_OF_CONNECTIVITY event, it sends the event report, by means of Namf_EventExposure_Notify message to the UDM.

In the Table below, a summary of the APIs used in the above procedure is presented.

Table 2 APIs used in the Loss of Connectivity event

NF provider	NF consumer	Service	API	Relevant 3GPP Document / GitHub link
NEF	AF	MonitoringEvent API	POST /{scsAsID}/subscriptions	TS 29.122/29.522 – link
UDM	NEF	EE API	POST /{ueIdentity}/ee-subscriptions	TS 29.503 – link
AMF	UDM	EventExposure API	POST /subscriptions	TS 29.518 – link
UDM	AMF	SDM API	GET /{ueId}/sdm-subscriptions	TS 29.503 – link

2.1.3 NEF/AsSessionWithQoS

Applications can utilize AsSessionWithQoS API to ensure better service experience and avoid service interruption which may be provoked due to unexpected QoS downgrades. 5GS comprises a complex QoS model where different networks function from the 5GC and the RAN interchange information to make the necessary adaptations for the best possible end-to-end connectivity (i.e., 5GS provides connectivity by establishing a session between UE and the data network).

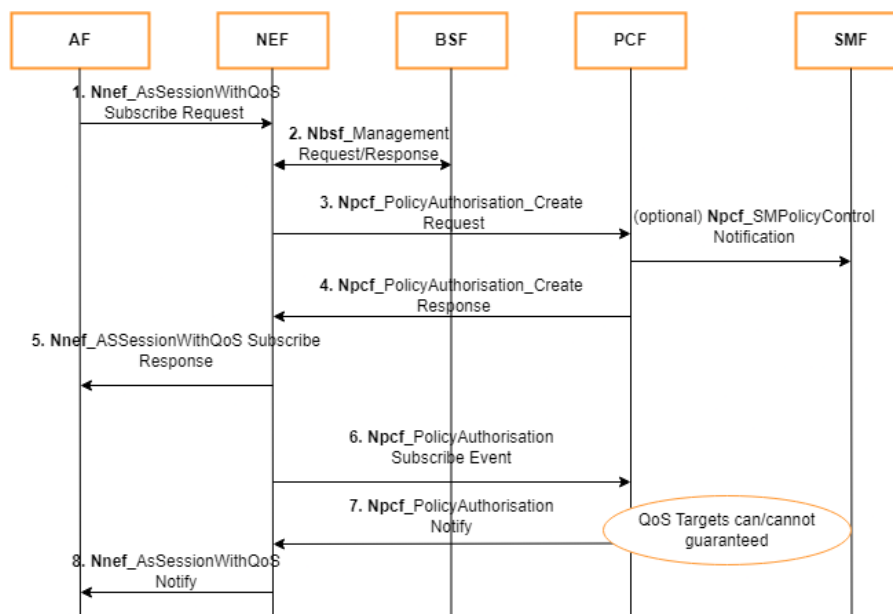


Figure 3 Information flow for the AsSessionwithQoS

1. The AF sends a request to subscribe for QoS Flow using the QoS Reference attribute through **Nnef_AsSessionWithQoS**.
2. NEF uses the UE address to discover the PCF from the BSF through **Nbsf_management**.
3. After discovering the PCF, NEF forwards the received parameters from the AF to PCF through **Npcf_PolicyAuthorisation**
4. The PCF derives the required QoS parameters of the PCC rule based on the information provided by the NEF and determines whether this QoS is allowed (according to the PCF configuration) and notifies the result to the NEF.

(Optional) If the PCF determines that the SMF needs updated policy information, the PCF issues a **Npcf_SMPolicyControl_Notification** request with updated policy information about the PDU Session.

5. The NEF sends a response message (Transaction Reference ID, Result) to the AF. The result indicates whether the request is granted or not.
6. NEF subscribes for the event (QOS_NOTIFY) (i.e., QoS targets can no longer/ can again be fulfilled) [23503] to the PCF.
 - When the event condition is met, e.g., that the establishment of the transmission resources corresponding to the QoS update succeeded or failed, the PCF sends **Npcf_PolicyAuthorization_Notify** message to the NEF notifying about the event.
8. NEF forwards the notification to the AF

In the Table below, a summary of the APIs used in the above procedure is presented.

<u>NF provider</u>	<u>NF consumer</u>	<u>Service</u>	<u>API</u>	<u>3GPP Document / GitHub link</u>
<u>NEF</u>	<u>AF</u>	<u>AsSessionWithQoS API</u>	<u>POST /{scsAsID}/subscriptions</u>	<u>TS 29.122/29.522 – link</u>
<u>BSF</u>	<u>NEF</u>	<u>Management API</u>	<u>GET /pcfBindings</u>	<u>TS 29.521 – link</u>
<u>PCF</u>	<u>NEF</u>	<u>PolicyAuthorization Service API</u>	<u>POST /app-sessions</u>	<u>TS 29.514 – link</u>
<u>PCF</u>	<u>SMF</u>	<u>SMPolicyControl API</u>	<u>POST /callback</u>	<u>TS 29.512 – link</u>
<u>PCF</u>	<u>NEF</u>	<u>PolicyAuthorization Service API</u>	<u>POST /app-sessions/{appSessionId}/events-subscription</u>	<u>TS 29.514 – link</u>

2.2 NWDAF/AF EVENT EXPOSURE SERVICE

NWDAF's implemented APIs follow 3GPP TS 29.517 [5] technical specification, which is pertinent to the "Application Function Event Exposure Service". The details of the API calls enabling the interaction between NWDAF and the AF, are illustrated in the below figure:

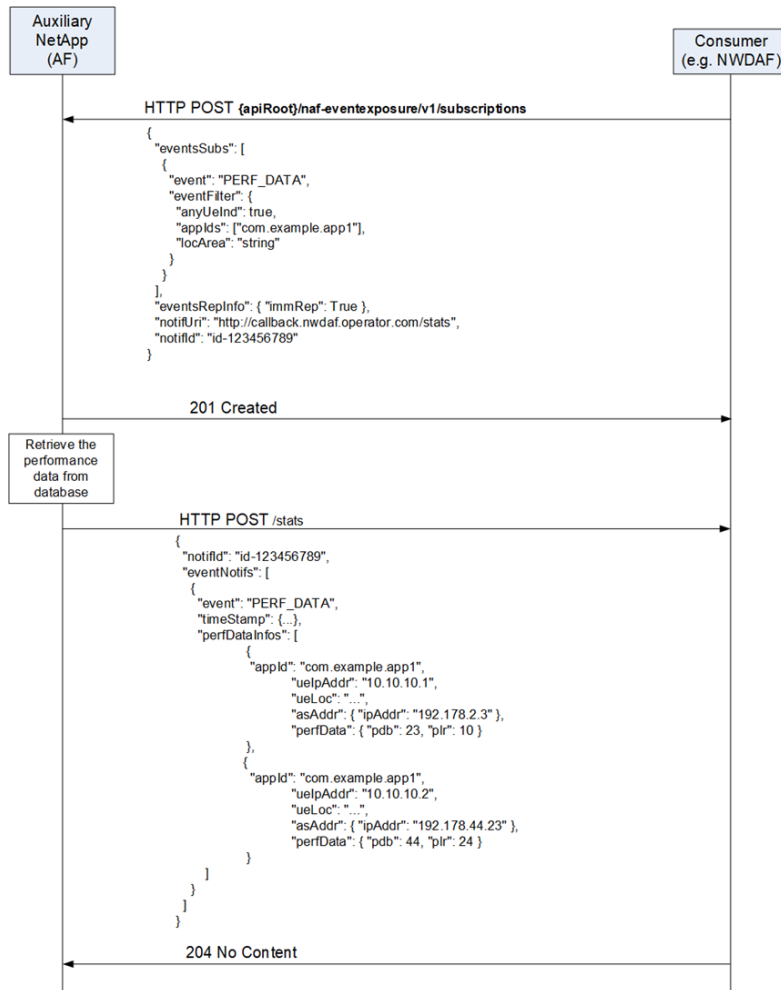


Figure 4 API calls enabling the interaction between NWDAF and the AF

Step A: NWDAF’s event exposure subscription request towards AF. Step B: AF’s notification callback request towards NWDAF.

As illustrated in the above figure, NWDAF-AF communication takes place in two steps.

- In the first POST request, NWDAF (API consumer) subscribes to AF (API producer), requesting performance data for a specific application (i.e., for fqdn: app=com.example.app1), and immediate reporting. In addition, NWDAF indicates a notification callback URI where it can be reached.
- In the second POST request, AF (API producer) provides performance data for two different UEs and two different Application Servers that host the application. (“pdb=23” indicates 23ms average packet delay and “plr=10” denotes 1.0% average packet loss rate.)

In this implementation, after those steps are completed, NWDAF (based on the statistics retrieved) calculates performance predictions using an AI engine. Finally, the derived performance analytics (statistics and predictions) are available to be consumed via an NWDAF’s exposing API. Beyond the official project commitments, this NWDAF’s preliminary implementation is considered a salient project outcome which serves as the basis for future extended development, in an attempt to assist vertical applications through the exploitation of network and application related analytics.

2.3 TSN APIs

Release 17 adds more enhancements to the 5G system's support of time synchronization services, moving it from simply supporting IEEE 802.1AS with Ethernet type PDU sessions to supporting the entire extent of IEEE Std 1588 with both Ethernet and IP type PDU sessions. To allow such time synchronization improvement and to assist an AF in requesting 5G to provide time synchronization services and/or to support non-TSN time sensitive communications, a new network entity Time Sensitive Communication and Time Synchronization Function (TSCTSF) is introduced. Depending on the network operator's deployment scenario, TSCTSF communicates with Network App directly when Network App is trusted or indirectly via NEF when Network App is not trusted.

The primary role of the TSCTSF is to deliver the following services to the Network App, either directly or indirectly through NEF, as stated in 3GPP TS29.565 and TS29.522:

- Providing Network App with 5GS synchronization capabilities and availability to provide time synchronization service via the Application Programming Interface (API):
 - Ntsctsf_TimeSynchronization_CapsSubscribe/CapsUnsubscribe/CapsNotify APIs;
 - Nnef_TimeSynchronization APIs "Time Synchronization Exposure Subscriptions" and "Individual Time Synchronization Exposure Subscriptions" resources of the Exposure API, as well as the API's "Time Synchronization Capability Notification"
- Enabling Network App to use the following APIs to configure the DS-TT and NW-TT of the 5GS to operate on an AF-selected synchronization method, such as one of the PTP instances compliant with IEEE Std 1588 or IEEE 802.1AS;
 - Ntsctsf_TimeSynchronization_ConfigCreate/ConfigUpdate/ConfigUpdateNotify/ConfigDelete APIs;
 - Nnef_TimeSyncExposure API
- Using the following APIs, a Network App can setup 5GS to spread the 5G access stratum time clock to outside of 5GS:
 - Ntsctsf_ASTI_Create/Update/Delete/Get APIs;
 - Nnef_TimeSync APIs "ASTI Configurations" and "Individual ASTI Configurations" API resources should be exposed;
- Enabling an AF to provide 5GS with the necessary QoS requirements, such as packet delay budget (PDB), maximum data burst volume (MDBV), and priority level, so that the 5G system may decide and apply the appropriate end-to-end QoS treatment to the relevant service data flow. Furthermore, for predictable periodic traffic, AF should give the traffic characteristics (including periodicity, burst arrival time, and service survival duration) to the 5G system to support adequate scheduling.
 - Ntsctsf_QoSandTSCAssistance_Create/Update/Delete/Notify/Subscribe/Unsubscribe APIs;
 - Nnef_AFSessionWithQoS API resources "AS Session with Required QoS subscriptions" and "Individual AS Session with Required QoS subscriptions"

2.4 SEAL APIS

Service Enabler Architecture Layer (SEAL) specifies the northbound APIs for its individual services so as to enable flexible integration with vertical applications. The SEAL client(s) communicates with the SEAL server(s) over the SEAL-UU reference points. SEAL-UU supports both unicast and multicast delivery modes. The SEAL client(s) provides the service enabler layer support functions to the VAL client(s) over SEAL-C reference points. The VAL server(s) communicate with the SEAL server(s) over the SEAL-S reference points. The SEAL server(s) may communicate with the underlying 3GPP network systems using the respective 3GPP interfaces specified by the 3GPP network system. The overall functionality is presented in the figure below.

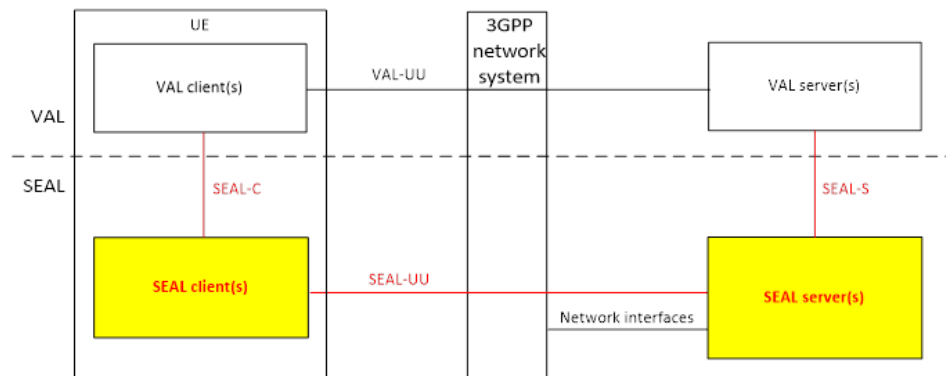


Figure 5 VAL and SEAL generic functional model

In the beginning of the project and more specifically during the requirements' collection phase, it had been identified that vertical applications would benefit from the location services offered by the 5G network. In an attempt to fulfill this common (for all vertical applications) need, EVOLVED-5G project selected to implement specific APIs (described in section 2.1) to be consumed by the developed Network Apps; the "Location API" was the one to be implemented and is exposed through NEF API for this purpose.

The exploration and the proposition of SEAL APIs within the project, were initially driven by the goal of offering services tailored for a massive number of devices, specifically centered around group management and configuration. However, as we delve into the diverse ecosystem of EVOLVED-5G and the discussions with SMEs and their distinct vertical application use-cases, it becomes evident that the aforementioned extensive management requirements were not universally essential. In addition, the location service, also available through SEAL, is already provided through the implementation and exposure by NEF APIs, which minimized further the necessity for SEAL usage within the scope of EVOLVED-5G project.

Fulfilling the various Network Apps' requirements while avoiding the unnecessary adaptations and complexity SEAL layer imposes on the vertical application domain, the project managed to safeguard a sustainable roadmap, focusing the efforts on implementing and exposing APIs, that could bridge the gap between the diversity of use cases, harmonizing them under a common and universal approach.

2.5 CAMARA APIs

One of the key benefits of NEF's API exposure is that it effectively hides the complexity of the underlying network from developers. By providing standardized interfaces, the NEF simplifies the interaction between applications and the network, allowing developers to focus on building their applications without needing in-depth knowledge of the mobile network's functionalities. However, the exposure of those specific network functionalities and the definition of interfaces is led by 3GPP standardization body, thus these APIs are not readily suitable for most application developers who lack in-depth knowledge of the telecom network. To bridge this gap, a translation layer has been introduced to convert service APIs into a format that can be exposed to enterprise app developers.

To facilitate the development and adoption of these APIs, a rapidly growing industry alliance called CAMARA has been established as an open-source project under the Linux Foundation. CAMARA collaborates with the GSMA Operator Platform Group to align API requirements and publish API definitions and documentation for developers. This collaborative effort aims to make applications globally available. Many leading service providers, vendors, and cloud providers have already become members of this alliance. CAMARA's main focus is to abstract APIs that are generally defined in SDOs like 3GPP, ETSI, and TMF, which are referred to as internal APIs, to allow the exposure of capabilities beyond the 5G Core network. The Transformation Function is responsible for abstracting internal APIs into Service APIs.

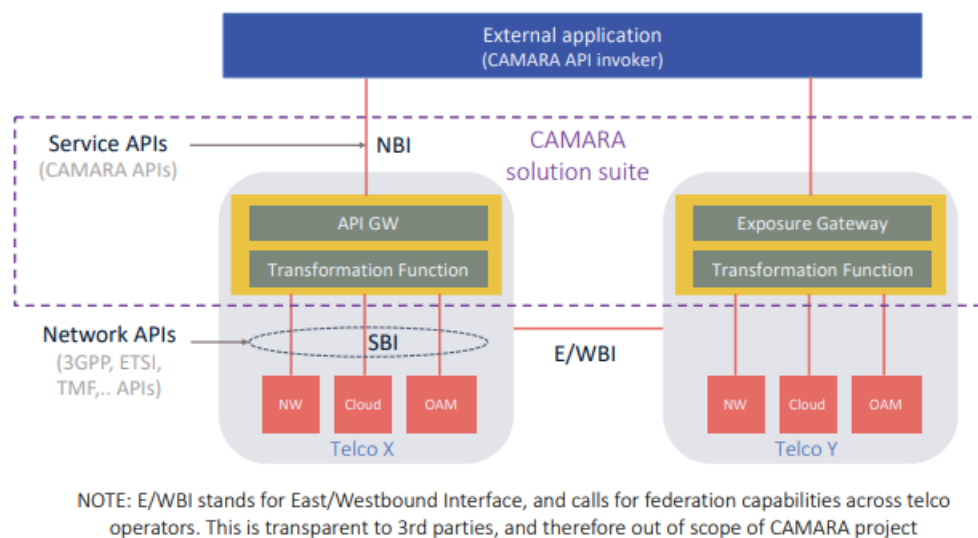


Figure 6 CAMARA architectural framework

Figure 6 depicts the reference architectural framework of CAMARA. As seen, it includes the following components:

- **Network APIs:** These are the APIs which are implemented in telco assets, including network resources (core, access, transport functions), cloud resources (virtualized and cloud-native workload hosting infrastructures) and IT resources (OSS and orchestration tools). These APIs are typically defined in standard bodies or industry

fora and tied to the underlying technology. Examples of these APIs include the ones defined by 3GPP¹, ETSI² or TMForum³, among others.

- Service APIs: These are the Network as a Service (NaaS) APIs, the ones to be made available for consumption to 3rd party applications, a.k.a. external applications. They are designed according to the principles listed in Section 1: open (API source code is Apache 2.0), global reach (they are offered by different telco operators) and user-friendly (service APIs result from the abstraction of network API semantics, hiding telco complexity).
- Transformation function: It keeps the information on correspondences between service APIs and network APIs and executes workflows to enforce these mappings. This component can be deployed as a microservice provisioned with a workflow engine.
- API Gateway: It provides all the capabilities that are needed to policy the interaction between the operator and the external applications, in relation to service API invocation. These capabilities include service API publication & discovery, access control (authentication & authorization of applications), auditing, accounting and logging. As seen, the focus of CAMARA work is on service APIs, and how they can build atop existing network APIs with the help of transformation function and API Gateway.

As seen in Figure 6, the architectural components building up the CAMARA solution suite are two: the transformation function and the API Gateway. The first component can be implemented using a workflow engine, while for the API Gateway the only mandate is that it needs to support OAuth2.0 with client credentials grant. There exists a wide variety of API Gateway solutions, both open-source and commercial, which are OAuth2.0 compliant. However, for the sake of consistency when offering service APIs, it is desirable to agree on a common, standards-based solution for this gateway. In this regard, CAMARA proposes to use 3GPP Common API Framework (CAPIF). CAMARA considers the CAPIF framework a convenient reference for managing the interaction between the API provider and consumer, as it is a standard solution accepted widely across the industry and not tied only to 3GPP APIs.

¹ <https://www.3gpp.org/>

² <https://www.etsi.org/>

³ <https://www.tmforum.org/>

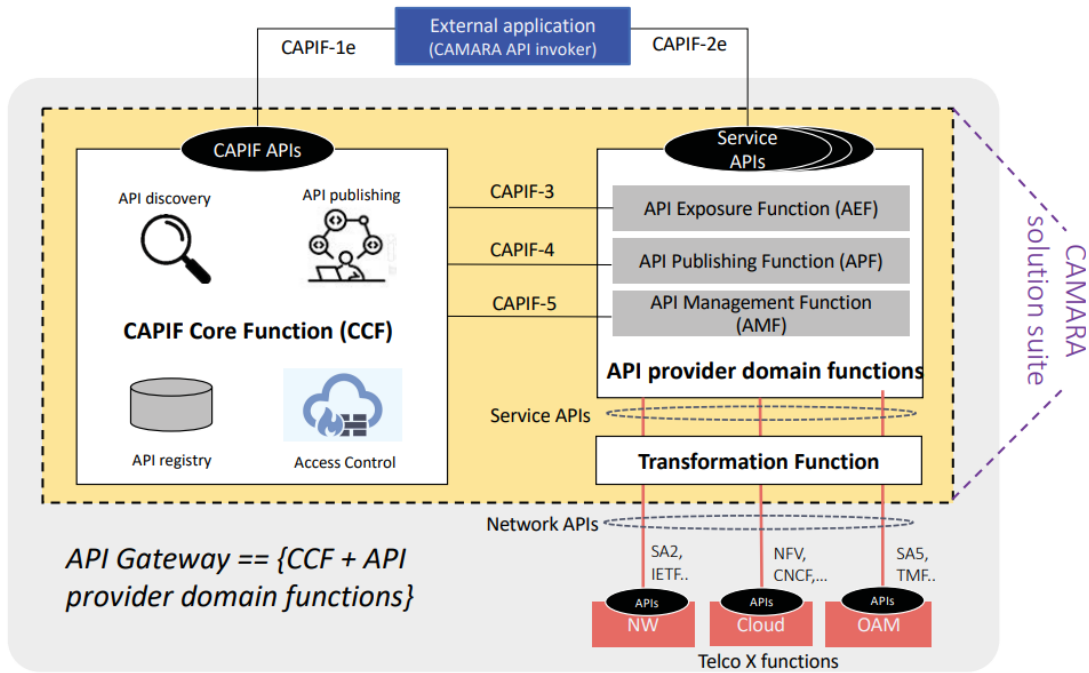


Figure 7 CAPIF as reference API GW for CAMARA

The reason for this proposal is twofold. On the one hand, CAPIF is a normative solution with wide acceptance at industry. The other main advantage of CAPIF is that though specified by the 3GPP, it is not tied to 3GPP APIs; indeed, CAPIF can be used as an API Gateway for any API, regardless of their semantics. Therefore, it is an ideal candidate to publish CAMARA APIs, and policy their exposure to the 3rd party applications.

It's important to note that during the EVOLVED-5G WP4, a continuous effort was made to simplify the complex "language" of 3GPP specifications and make them more accessible to network application developers. A prime example of this is the Monitoring Event API, which was originally defined by a complicated YAML file. In the project and in the context of WP4 especially, the main objective was to simplify the process by focusing on the essential and fundamental attributes. This allowed for the publishing of the location reporting event in a manner that developers could understand, bridging the gap between the technicalities of the telecommunications industry and the developers' understanding.

Due to this close relation in scope between EVOLVED-5G and CAMARA, it is worth-mentioned that partners TID, NCSR and INF have become members of CAMARA in order to act as liaison between the two activities and communicate the EVOLVED-5G results and activities effectively in CAMARA.

3 IMPLEMENTATION OF 5G NORTHBOUND APIS IN EVOLVED-5G

3.1 NEF APIS

One crucial factor that EVOLVED-5G had to take into consideration is that currently there are no open commercial solutions available that fully implement the service-based architecture and the southbound interfaces necessary for NEF to expose standardized APIs. As described in D4.1, EVOLVED-5G addressed this challenge by developing a NEF emulator. The primary goal of this emulator was to overcome the limitations by creating simulated and emulated events. For more information on the implementation specifics of the NEF emulator, please refer to document D3.1 [6].

The implementation process for the various APIs of the NEF emulator has been prioritized based on the needs expressed by the EVOLVED-5G SMEs. The joint analysis of those needs during the first period of the project, in accordance with the relative tasks of WP4, has resulted in a set of target APIs for the first version of the NEF emulator. Thus, the commonly identified APIs for implementation were the *AsSessionWithQoS* and the *MonitoringEvent* API. Initially the *MonitoringEvent* API supported and implemented the location reporting event. Specifically, for the location reporting, when a UE handover takes place to a neighbor cell, NEF informs for this event, letting the Network App know the new id of the cell where the UE moved to. More details about the implementation of the location reporting event can be found in D4.1 [7].

However, throughout the lifetime of the project it was deemed necessary to enhance the already implemented APIs with features that would be highly valuable for the SMEs and their respective Network Apps. Consequently, the *MonitoringEvent* API has been enhanced with two additional events, namely the *UE reachability* and the *loss of connectivity*. The main characteristics for the two events are as follows:

- *UE reachability*: This option is primarily utilized when applications need to detect when a UE becomes reachable again after an extended period of power-saving sleep. In the implementation within EVOLVED-5G the 5G network (NEF Emulator) identifies the event when a UE becomes active and reachable, and subsequently, the NEF notifies the application about the UE's availability. This allows the application to send downlink data intended for transmission.

POST /api/v1/3gpp-monitoring-event/v1/{scsAsId}/subscriptions Create Subscription

Create new subscription.

Parameters	Callbacks
Name	Description
scsAsId * required string (path)	<input style="width: 100%;" type="text" value="myNetapp"/>

Request body required

```

{
  "externalId": "10003@domain.com",
  "notificationDestination": "http://localhost:80/api/v1/utls/monitoring/callback",
  "monitoringType": "UE_REACHABILITY",
  "maximumNumberOfReports": 1000,
  "monitorExpireTime": "2024-07-27T12:02:31.521Z",
  "reachabilityType": "DATA"
}
                    
```

Figure 8 MonitoringEvent API - Loss of connectivity subscription

- **Loss of connectivity:** As described in the previous section through the sequence diagram and the information flows, the main feature within this event is that an application can request notification if the UE loses connectivity, for example, if the UE detaches from the 3GPP Network or if the UE has not communicated with the 3GPP Network after a pre-defined time. The network detects that the UE is no longer reachable for either signaling (i.e., control plane) or user plane communication. The application may provide a maximum detection time to identify the maximum interval after which the UE is considered unreachable. The implementation of the specific functionality is depicted in Figure 9 below.

UE1

location: [37.995962, 23.818214]

Cell ID: -

External identifier: 10003@domain.com

Speed: LOW

POST /api/v1/3gpp-monitoring-event/v1/{scsAsId}/subscriptions Create Subscription

Create new subscription.

Parameters	Callbacks
Name	Description
scsAsId * required string (path)	<input style="width: 100%;" type="text" value="myNetapp"/>

Request body required

```

{
  "externalId": "10003@domain.com",
  "notificationDestination": "http://localhost:80/api/v1/utls/monitoring/callback",
  "monitoringType": "LOSS_OF_CONNECTIVITY",
  "maximumNumberOfReports": 100,
  "monitorExpireTime": "2024-07-27T11:58:53.019Z",
  "maximumDetectionTime": 1
}
                    
```

Figure 9 MonitoringEvent API - UE reachability subscription

Table 3 below presents the supported services and in the latest release of NEF (Release 2.2.2) for Monitoring event API and Table 4 for AsSessionwithQoS event respectively.

Table 3 Available services for Monitoring Event API

API prefix: /api/v1/3gpp-monitoring-event/v1		
Endpoint	Method	Description
/ {scsAsId} /subscriptions	GET	<p>Returns all the active subscriptions</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required)</p> <p><u>Response Body:</u></p> <p>200: Array of MonitoringEventSubscription objects 204: No Content 422: HTTPValidationError object (Validation Error) 401: Unauthorized</p>
/ {scsAsId} /subscriptions	POST	<p>Create Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required)</p> <p><u>Request Body:</u></p> <p>MonitoringEventSubscriptionCreate object (required)</p> <p>-MonitoringType (string) specifies the event to which the application wants to subscribe - Enumeration: "LOCATION_REPORTING", "LOSS_OF_CONNECTIVITY", "UE_REACHABILITY"</p> <p><u>Response Body:</u></p> <p>200: MonitoringEventReport object (Successful Response) [if maximum number of reports equals to 1] 201: MonitoringEventSubscription object (Created) 422: HTTPValidationError object (Validation Error) 409: Conflict "There is already an active subscription for UE with external id 'externalId'" 401: Unauthorized</p>
/ {scsAsId} /subscriptions / {subscriptionId}	GET	<p>Read Individual Subscription</p>

		<p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required) subscriptionId (path) - Identifier of the subscription resource (string, required)</p> <p><u>Response Body:</u></p> <p>200: MonitoringEventSubscription object (Successful Response) 422: HTTPValidationError object (Validation Error) 401: Unauthorized 404: Not Found</p>
<pre>/{scsAsId}/subscriptions /{subscriptionId}</pre>	<p>PUT</p>	<p>Update Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required) subscriptionId (path) - Identifier of the subscription resource (string, required)</p> <p><u>Request Body:</u></p> <p>MonitoringEventSubscriptionCreate object (required)</p> <p><u>Response Body:</u></p> <p>200: MonitoringEventSubscription object (Successful Response) 422: HTTPValidationError object (Validation Error) 401: Unauthorized 404: Not Found</p>
<pre>/{scsAsId}/subscriptions /{subscriptionId}</pre>	<p>DELETE</p>	<p>Delete Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required) subscriptionId (path) - Identifier of the subscription resource (string, required)</p> <p><u>Response Body:</u></p> <p>200: MonitoringEventSubscription object (Successful Response) 422: HTTPValidationError object (Validation Error)</p>

		401: Unauthorized 404: Not Found
--	--	-------------------------------------

Table 4 Available services for AsSessionWithQos API

API prefix: /api/v1/3gpp-as-session-with-qos/v1		
Endpoint	Method	Description
{scsAsId}/subscriptions	GET	<p>Returns all the active subscriptions</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required)</p> <p><u>Response Body:</u></p> <p>200: Array of AsSessionWithQoSSubscription objects 204: No Content 422: HTTPValidationError object (Validation Error) 401: Unauthorized</p>
{scsAsId}/subscriptions	POST	<p>Create Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required)</p> <p><u>Request Body:</u></p> <p>AsSessionWithQoSSubscriptionCreate object (required)</p> <p><u>Response Body:</u></p> <p>201: AsSessiowithQoSSubscription object (Created) 422: HTTPValidationError object (Validation Error) 409: Conflict "There is already an active subscription for UE with external id 'externalId'" 401: Unauthorized</p>
{scsAsId}/subscriptions/{subscriptionId}	GET	<p>Read Individual Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required)</p>

		<p>subscriptionId (path) - Identifier of the subscription resource (string, required)</p> <p><u>Response Body:</u></p> <p>200: AsSessiowithQoSSubscription object (Successful Response) 422: HTTPValidationError object (Validation Error) 401: Unauthorized 404: Not Found</p>
<pre>/{scsAsId}/subscriptions /{subscriptionId}</pre>	<p>PUT</p>	<p>Update Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required) subscriptionId (path) - Identifier of the subscription resource (string, required)</p> <p><u>Request Body:</u></p> <p>AsSessiowithQoSSubscriptionCreate object (required)</p> <p><u>Response Body:</u></p> <p>200: AsSessiowithQoSSubscription object (Successful Response) 422: HTTPValidationError object (Validation Error) 401: Unauthorized 404: Not Found</p>
<pre>/{scsAsId}/subscriptions /{subscriptionId}</pre>	<p>DELETE</p>	<p>Delete Subscription</p> <p><u>Request Parameters:</u></p> <p>scsAsId (path) - The ID of the Network App that creates a subscription (string, required) subscriptionId (path) - Identifier of the subscription resource (string, required)</p> <p><u>Response Body:</u></p> <p>200: AsSessiowithQoSSubscription object (Successful Response) 422: HTTPValidationError object (Validation Error) 401: Unauthorized 404: Not Found</p>

3.2 TSN APIs

The configuration of Time Sensitive Networking and deterministic communication in the platforms is exposed via the TSN FrontEnd component. This component implements a unified API that is independent from the actual functionality and capabilities of the underlying platform, thus making this component re-usable in different environments. The included API endpoints can be seen in Table 6.

Table 5 TSN Frontend API endpoints

API prefix: /tsn/api/v1		
Endpoint	Method	Description
/profile	GET	<p>When used without parameters the endpoint returns a list of available profiles, with the following format:</p> <pre>{"profiles": ["<profile1>", "<profile2>", ..., "best_effort"]}</pre> <p>When used with the <i>name=<profile_name></i> URL parameter, the endpoint returns the default configuration values of the selected profile, with the format:</p> <pre>{"<profile_name>": {"<parameter1>": <value1>, "<parameter2>": <value2>, ...}}</pre>
/apply	POST	<p>Applies the specified configuration to the selected traffic identifier. The endpoint expects to receive a payload with the format:</p> <pre>{"identifier": <identifier>, "profile": <profile>, "overrides": <overrides>}</pre> <p>Where:</p> <ul style="list-style-type: none"> - <i>Identifier</i> is a unique identifier defined by the user for the configuration. - <i>profile</i> is the name of the base profile to use. The values in this profile will be used as default, when not overridden. - <i>overrides</i> is a dictionary of values that will be overridden from the used profile. May be empty.
/clear	POST	<p>Disables the configuration applied by a previous usage of <i>/apply</i>. The endpoint expects to receive a payload with the format:</p> <pre>{"identifier": <identifier>, "token": <token>}</pre> <p>Where:</p> <ul style="list-style-type: none"> - <i>identifier</i> is the same unique identifier used when requesting the configuration through <i>/apply</i> <p><i>token</i> is a value returned by the TSN AF as part of the response to the <i>/apply</i> request.</p>

A more thorough description of the TSN FrontEnd’s usage and functionalities, as well as its relation with the associated TSN Controller can be found in Deliverable D3.3, particularly in section 6.1.4.

3.3 NWDAF

In the context of the project and as part of Lenovo’s work on the Auxiliary Network App proof-of-concept, a Network Data Analytics Function (NWDAF) prototype has been developed, deriving analytics (statistics and predictions) which can in turn be consumed by other Network Apps, assisting them ultimately in their required operations. NWDAF prototype has been developed according to 3GPP standards, exposing 3GPP conforming APIs. Within the Auxiliary Network App prototype, NWDAF operates as service consumer according to the 3GPP Service Based Architecture (SBA), subscribing to a Network Application (considered to be an Application Function - AF), in order to retrieve connectivity statistics, and based on them, derive predictions using an AI analytics engine. Such network connectivity related analytics are eventually exposed by NWDAF, so as to be exploited by vertical application clients, assisting them (in the case of our proof-of-concept) to select the best vertical application server to connect to.

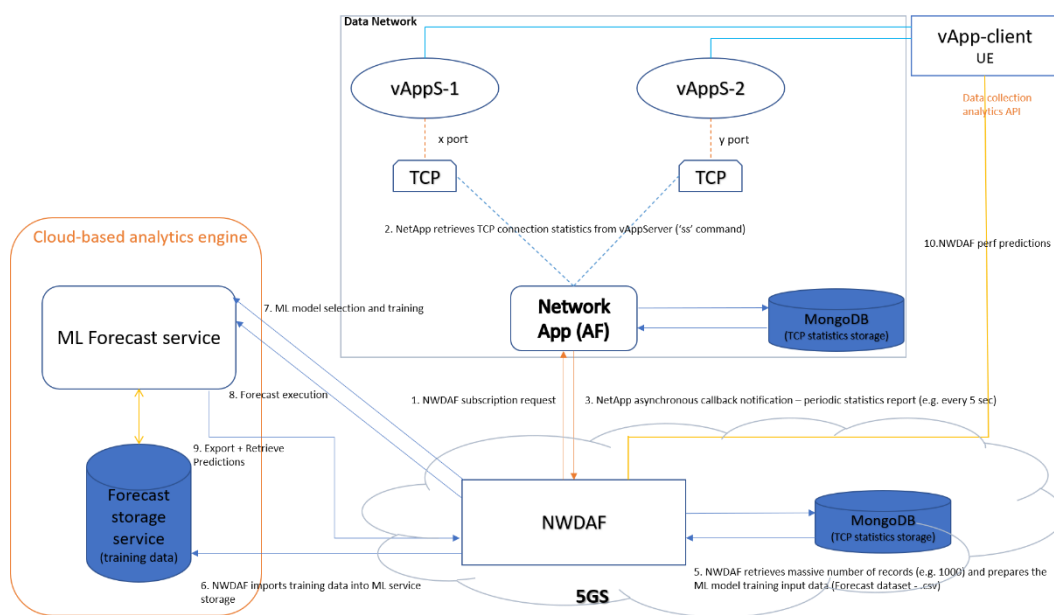


Figure 10 Network App acts as a TCP stats collector providing inputs to NWDAF

Figure 10 depicts the design concept of the procedure where the Network App acts as a TCP stats collector and provides input to NWDAF within the 5GS to support the calculation and extraction of application performance related analytics (stats or predictions).

Network App periodically collects TCP performance statistics from the platform (VAL servers or networking stack at the DN), assisting NWDAF to derive application latency statistics or predictions per indicated VAL server. Once NWDAF derives application layer performance analytics (in terms of latency), it can provide this information to the vApp client UE for the latter one to select the most appropriate VAL server to connect to in terms of latency. This way, application performance analytics utilization assists the vApp client in selecting the optimal vApp server and achieve low latency connectivity for a specific application service.

3.4 INTEGRATION OF NORTHBOUND 5GS APIs WITH CAPIF

The use of Application Programming Interfaces (APIs) has served for many years as a bridge between mobile operators and start-ups in emerging markets. Operators have begun to consider whether to open their APIs, starting from APIs related to mobile messaging, operator billing etc. In addition, the recently witnessed convergence of IT and Telecom worlds have contributed a lot on putting APIs in the epicentre of network programming and service provisioning. Representative examples that prove this statement are: the 5G Service Based Architecture (SBA), which has been designed based on the flexibility that HTTP/2 Restful APIs to provide interaction among 3GPP network functions; the intense work on API specification and development in open project and fora (e.g., the OpenAPI project of TM Forum); and the wide adoption of the microservice software pattern/architecture which is based on API interacting software pieces. In the same direction, Network Apps provide network- or vertical- oriented services, meaning that they can assist/enhance either the network operation/management⁴ or the vertical application⁵. As third-party applications, the Network Applications should interact with network functions/nodes through open and standardised interfaces/APIs that can reside at any plane (user, control, management) or any domain (core, radio, transport). Thus, a widely accepted and standardised API framework is needed, in order to guarantee interoperability and security in that interaction.

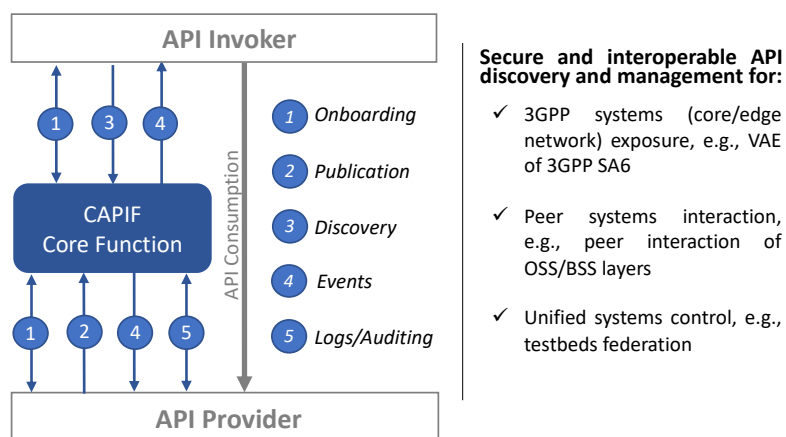


Figure 11 Abstract illustration of the CAPIF functionality, with representative examples of its applicability

The 3GPP Common API Framework (CAPIF) is adopted as the main candidate framework for that purpose. CAPIF has been an integral part of the 3GPP SA6 specifications since Rel. 15 and continues to evolve with new features and reference points, added in any following Release. Its

⁴ For instance, in the EVOLVED-5G project a related contribution to 3GPP SA6 work has emerged (3GPP/TSG SA2/eNA_Ph2 Rel.17: Contribution "Support of DN performance analytics by NWDAF" - S2-2101388) under the scope of extending the NWDAF analytics APIs so that Network Applications can retrieve data from vertical apps, and the NWDAF build performance analytics & predictions by using inputs from Network Applications.

⁵ The ICT-41 projects (5GPPP, phase 3, part 6 projects), work towards providing Network Applications that fulfil needs and requests from various vertical industries, e.g., automotive (5GIANA, 5GASP), Industry 4.0/manufacturing (5GINDUCE, EVOLVED5G, 5GERA), transport & logistics (VITAL5G, 5GERA), media (5GMediaHub), public protection and disaster relief (5G-EPICENTRE, 5GERA, 5GASP), healthcare (5GERA).

objective is to provide a unified and standardized northbound API framework across several 3GPP functions. Practically, it has been designed to facilitate the network core exposure, towards new application enablers of various vertical industries (including, Unmanned aerial systems, Edge data networks, Factories of the future, and Vehicular communication systems). Beyond this initial target, CAPIF has been used as a key standardized API-management framework for secure and interoperable interaction among any API providers and API consumers as depicted in Figure 11.

3.4.1 CAPIF APIs

Based on the 3GPP specifications, the main services that have been defined for the CAPIF Core Function (CCF) are listed in Table 5 below. In the current release of CAPIF (Release 3.1.2), all these services are available. To implement the API services of the CCF, we used the API definitions/signatures that are specified in 3GPP TS 29.222 [8] and are already published as YAML files [9]. The YAML files of those services, were used by a Swagger editor to inspect all information elements in JSON. Each of these YAML files defines one or more APIs and the supported methods to use them (POST, GET, DELETE, PUT).

Table 6 Available Northbound APIs for the CAPIF services

CAPIF_Discover_Service_API	
Service Operations	Consumer(s)
Discover_Service_API	API Invoker, other CCF
Event operations (CAPIF_Events_API)	API Invoker
CAPIF_Publish_Service_API	
Service Operation(s)	Consumer(s)
Publish_Service_API	API Provider (APF), other CCF
Unpublish_Service_API	API Provider (APF), other CCF
Update_Service_API	API Provider (APF), other CCF
Get_Service_API	API Provider (APF), other CCF
CAPIF_Events_API	
Service Operation(s)	Consumer(s)
Subscribe_Event	API Invoker, API Provider (APF, AEF, AMF)
Notify_Event	API Invoker, API Provider (APF, AEF, AMF)
Unsubscribe_Event	API Invoker, API Provider (APF, AEF, AMF)
CAPIF_API_Invoker_Management_API	
Service Operation(s)	Consumer(s)
Onboard_API_Invoker	API Invoker

Offboard_API_Invoker	API Invoker
Notify_Onboarding_Completion	API Invoker
Update_API_Invoker_Details	API Invoker
Notify_Update_Completion	API Invoker
CAPIF_Security_API	
Service Operation(s)	Consumer(s)
Obtain_Security_Method	API Invoker
Obtain_Authorization	API Invoker
Obtain_API_Invoker_Info	API Provider (AEF)
Revoke_Authorization	API Provider (AEF)
CAPIF_Monitoring_API	
Service Operation(s)	Consumer(s)
Event operations (CAPIF_Events_API)	API Provider (AMF)
CAPIF_Logging_API_Invocation_API	
Service Operation(s)	Consumer(s)
Log_API_Invocation	API Provider (AEF)
CAPIF_Auditing_API	
Service Operation(s)	Consumer(s)
Query_API_Invocation_Log	API Provider (AMF)
CAPIF_API_Provider_Management_API	
Service Operation(s)	Consumer(s)
Register_API_Provider	API Provider (AMF)
Update_API_Provider	API Provider (AMF)
Deregister_API_Provider	API Provider (AMF)

3.4.2 NEF INTEGRATION WITH CAPIF

In CAPIF, NEF Emulator acts as an API Provider taking the roles of publisher, exposer and management. Regarding implementation, the integration with CAPIF is realized through the EVOLVED-5G SDK libraries. In the API Provider Domain, the NEF authenticates with CCF and after the security negotiation it is onboard in the CCF. As part of the successful onboarding, NEF assumes the role of APF and publishes its APIs to the CCF. Finally, NEF is ready to expose the services to external applications (playing the role of AEF) under CCF's supervision. Access to the

API is securely managed through authentication tokens generated by CAPIF. A simplified workaround for exposing network capabilities to Networks apps through CAPIF is presented in Figure 12. Furthermore, the NEF Emulator makes use of the Logging API that allows all incoming requests received from network apps, together with the corresponding responses, to be logged into CCF. These logs can be later audited if necessary, providing a complete record of the interactions with the NEF Emulator APIs.

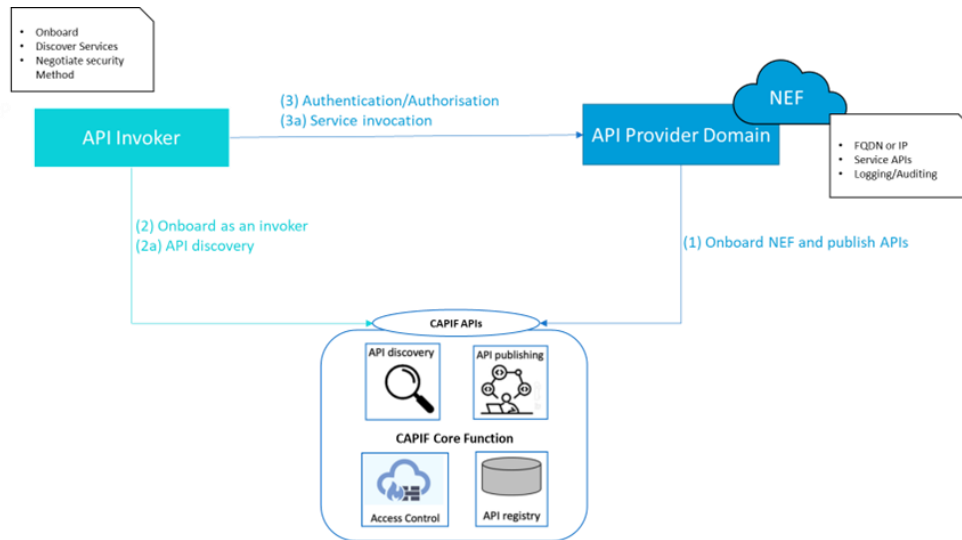


Figure 12 NEF exposure capabilities to Networks Apps through CAPIF

3.4.3 TSN INTEGRATION WITH CAPIF

The TSN FrontEnd is a critical component that seamlessly integrates with the EVOLVED-5G SDK libraries to facilitate smooth communication with the CAPIF instance of the platforms. This integration offers several essential functionalities as follows:

Firstly, the TSN FrontEnd serves as a Service Provider, exposing the TSN APIs to the allowed API Invokers within the system. By doing so, it enables discoverability of the TSN service, making it easily accessible to authorized users. To ensure security and proper access control, the API requires authentication tokens generated by CAPIF, guaranteeing that only authorized entities can interact with the TSN service.

Secondly, the TSN FrontEnd utilises on the Logging capabilities offered by CAPIF. Every incoming request through the APIs, along with information about the requester, is logged. Additionally, the FrontEnd captures the corresponding responses provided by the TSN service. This unified log serves as an auditing trail, which is beneficial in case of any necessity.

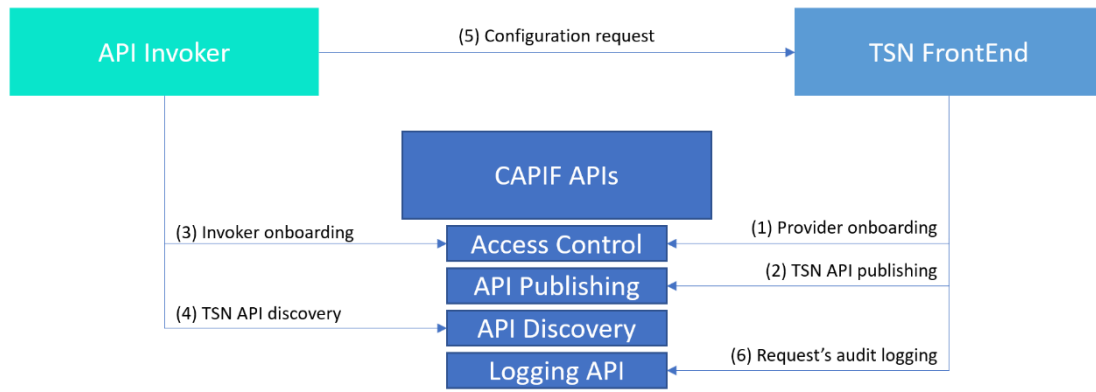


Figure 13 API Invoker, TSN FrontEnd and CAPIF interaction

4 DEVELOPMENT TOOLS AND OVERALL FRAMEWORK

4.1 DEVELOPMENT TOOLS

In EVOLVED-5G several tools to facilitate the developers towards the creation and development of their Network Apps within the EVOLVED-5G framework, have been created. These tools have been explained in detail in Deliverable D3.3 [10]. Below are the final updates of the developed tools along with a brief explanation about their purpose, are presented.

- **Network App template**
 - It has been developed to provide an example to developers on how a Network App in EVOLVED-5G must look like. This tool does provide a file and folder structure to developers therefore, once the Network App is created it is ready comply and go throughout every phase of the Network App lifecycle.
- **SDK**
 - CLI tool
 - The SDK CLI tool does serves twofold, i) create local and remote the repository for a new Network App with the help of the Network App template and ii) provide different commands to launch verification tests allowing the Network App to be verified prior going to validation phase.
 - Libraries
 - The SDK libraries are a set of Python classes providing abstraction towards the 5G APIs allowing developers to register and authenticate their Network App to CAPIF and “providers” (API creators wanting to make their APIs available to the Network Apps) to register their resources to CAPIF.
- **Open repositories**
 - EVOLVED-5G does manage two open repositories i) GitHub, as a code repository for Network Apps and ii) an artifact repository to store the Network App images
- **Dummy Network Application**
 - It is a non-functional piece of software designed acting as an implementation example for developers, reproducing the functionalities an actual Network App must have within the EVOLVED-5G project.

4.1.1 Updates on The Development Tools

This section provides a comprehensive overview of the various developments that have taken place on top of the different tools under the Workspace environment of EVOLVED-5G. The specific environment facilitates the initial phases within the EVOLVED 5G workflow, namely the NetApp development and verification. The development of a Network App marked the first phase, where developers utilized the array of tools provided by the EVOLVED-5G Facility. Throughout the lifetime of the project and compared to the information provided in D4.1, these tools have undergone evolution and enhancements, leading to improved functionality and efficiency that paved the way for the delivery of the final prototype of the Network Apps within each task of WP4. By detailing these advancements, this section serves as a valuable resource for understanding the evolution of the tools and their impact on the overall project workflow.

4.1.1.1 Network App template

The Network App template has undergone updates, resulting in a restructured "src" folder with new files and folders for improved functionality.

4.1.1.2 CLI tool

The SDK CLI tool serves two purposes: (i) creating the Network App locally and remotely on GitHub, and (ii) launching verification tests through the implemented pipelines in EVOLVED-5G. In the updated version of the tool, several changes have been implemented to enhance the developer's experience and simplify the input process. Instead of manually providing inputs via the terminal, the developer has given the option to compile a configuration file.

4.1.1.3 Dummy Network Application

The presence of the multiple entities/developments within the EVOLVED-5G ecosystem, such as CAPIF, NEF, and TSN, along with the functionalities of the APIs that Network Applications need to employ for communication with the 5G network, highlights the complexity and the demanding nature involved in the development of new Network Apps. To address this difficulty at the early stages of the project, a supplementary Network App named the Dummy Network App was created as a demonstrative model for application developers and its technical parameters and details are described thoroughly in D3.1 and D3.3.

The Dummy Network App, in alignment with the approach taken by all EVOLVED-5G Network Apps, is a container-based application developed in Python. It incorporates essential functions and employs SDK libraries to enable seamless communication with CAPIF, NEF, and TSN. It is important to note that throughout the project's duration, the Dummy Network App has been continuously enhanced to streamline developers' activities.

4.1.1.4 SDK Libraries

In order to save time on the developers' side and streamline communication setup, the SDK libraries have evolved to offer enhanced functionality. These libraries consist of a set of Python classes that served multiple purposes within the project.

In the initial version, the SDK libraries provided basic functionality for developers to interact with the NEF emulator. However, in response to the need for improved efficiency and reduced complexity, the SDK libraries have undergone significant enhancements and evolved to offer advanced functionality. One key delta from the initial version is the integration of registration and onboarding functionalities into the CAPIF tool for Network app developers. With this class, developers can effectively register, onboard, and publish endpoints to a CAPIF server.

In the current stage of the project, the evolved SDK now supports nine additional libraries compared to the initial version. This expansion in library support broadens the range of functionalities and resources available to developers, enabling them to leverage a more comprehensive set of tools and features for their development tasks. The EVOLVED-5G SDK at its current status implements the following six libraries:

- **LocationSubscriber:** allows for the tracking of devices and retrieves updates about their location. LocationSubscriber to create subscriptions, where each one of them can be used to track a device.
- **QoSAwareness:** allows for the request of QoS from a set of standardized values for better service experience (Ex. TCP_BASED / LIVE Streaming / CONVERSATIONAL_VOICE etc). The subscriptions can be created based on specific QoS parameters. A notification is sent back to the Network App if the QoS targets can no longer be fulfilled.
- **ConnectionMonitor:** allows you to monitor devices in the 5G Network. This class is used to retrieve notifications by the 5G Network for individual devices when connection is lost (for example the user device has not been connected to the 5G network for the past 10 seconds) or when connection is alive (for example the user device has been connected to the 5G network for the past 10 seconds).

- **CAPIFInvokerConnector** a low-level class, that is used by the evolved-5G CLI in order to register Network Apps to CAPIF
- **CAPIFExposerConnector** a low-level class, that allows an Exposer (like the NEF emulator) to register to CAPIF
- **ServiceDiscover** allows Network App developers to connect to CAPIF in order to discover services
- **TSNManager**: This Python class allows the Network App developer to interact with the TSN FrontEnd in order to apply Time-Sensitive Networking (TSN) standards to timesensitive Network Apps
- **CAPIFLogger**: This Python class allows a Provider (like the NEF emulator) to capture Log information to CAPIF.
- **CAPIFAuditor**: This Python class allows a Provider (like the NEF Emulator) to query the Log and retrieve information that was saved via the CAPIFLogger class.

4.1.1.5 SDK and EVOLVED-5G Services (NEF-CAPIF) mapping

Throughout the lifetime of the project, there have been several notable developments in the main software tools and services, namely the SDK and NEF and CAPIF. These advancements have contributed to the growth and improvement of the outcome regarding WP4 which is the final prototype of the Network Apps. The following table serves as a valuable resource for understanding the evolution of software components by presenting a mapping of their different versions. However, rather than delving into specific details, the table emphasizes the major updates that took place during each version, providing a more streamlined and concise representation of the software’s development history. By focusing on the major updates, the table highlights the significant enhancements, modifications, and new features introduced each time, serving as a quick reference point for identifying key milestones and advancements achieved over time. The overall release history of each of the components can be found in the GitHub repository of EVOLVED-5G⁶.

Table 7 Versioning of the EVOLVED-5G components

SDK version	NEF	CAPIF	TSN	Major Functionalities
v0.5.1	v1.0	-	-	Class LocationSubscriber
v0.6.0	v1.2	-	-	Class QoSAwareness
v0.8.0	v1.4	v1.0	-	New verification tests have been implemented related to NetApp code and NetApp container image analysis
v0.8.1	v1.5	v2.0	-	New class CAPIFConnector. New class ServiceDiscoverer.
v0.8.3	v1.6.1	v2.0		New SDK Library CAPIFExposerConnector,
v0.8.9	v1.6.2	v2.1		Added new verification tests CAPIF and NEF
v1.0	v2.0	v3.0		All Libraries classes (LocationSubscriber,

⁶ <https://github.com/EVOLVED-5G/>

				ConnectionMonitor, QosAwareness) communicate with CAPIF server to get authorization tokens. TSN library has been added in the SDK
v1.0.5	v2.1	v3.0	v1.0	TSN verification pipeline added.
v1.1	v2.2.2	v3.1.2	v1.2.1	Only CAPIF token is used. Offboard and deregister option has been added in the SDK.

4.2 UPDATES ON THE VERIFICATION TESTS

The verification process of a Network App checks several aspects of the code of the Network App that are relevant with its containerization, security aspects and communication with the 5G network. For the two first, standard tools are used and deployed via a proper pipeline in the Evolved-5G framework CI/CD platform. For the latter, extensive tests have been implemented to ensure proper interoperation of the Network App with the CAPIF Core Function tool, the NEF emulator and the TSN API.

The major updates of the verification process of the Network App are relevant with (i) the additional libraries included in the SDK, which are discussed in Section 3.1.14 as well as (ii) with added support for TSN. At the same time, the tests regarding CAPIF have significantly changed in terms of security and authorization, and in their nature as CAPIF is now involved in the whole Network App – NEF emulator, and Network App – TSN API communication channels. As a result, the following updates have been applied to the verification tests in comparison to the initial design reported in D4.1:

- **CAPIF and NEF:** Several changes were performed following the updates of the SDK and the relevant adjustments of the dummy Network App, which played the role of the reference application for verification. In particular:
 - Direct testing of CAPIF has been removed, with NEF bearing now the responsibility of registering with CAPIF during NEF deployment.
 - For NEF, tests regarding the request for appropriate authentication token were removed and were in practice replaced by the relevant SDK functionality. The tests now verify that the authentication part is performed using certificates, which are issued by the Network App during registration to CAPIF. In the eventuality of errors during registration, the verification fails.
- **TSN:** The TSN verification tests try to emulate a full working environment with both CAPIF and TSN components deployed and TSN fully registered to CAPIF. The tests are checking the communication between a registered Network App to CAPIF (similarly to NEF tests), as well as to the TSN exposed API leveraging the relevant SDK libraries.
- **General improvements:** The deployment stages for the Network applications and the Robot framework were generalized to support the execution of the pipelines, initially targeting the dummy Network App, so that any Network Application can be verified. Tests have also been refined in order to comply to the new file structure of the Network Apps, as a result to changes in SDK libraries.

For more details, please, refer to the deliverable D3.3 “Implementations and integrations towards EVOLVED-5G framework realisation” that was submitted in April 2023.

4.3 EVOLUTION OF THE NETWORK APPS

Section 3.2.2 of D4.1 [7] provides a detailed description of the specific development process that was implemented to ensure a consistent level of maturity across all Network Apps at the initial stage of the project. This process was carefully designed to promote homogeneity and uniformity in the results achieved by the various Network Apps. This approach involved establishing guidelines, best practices that all SMEs were required to adhere towards a consistent and unified framework for development, fostering interoperability and seamless integration among the different Network Apps.

Building upon this foundation and given the involvement of 11 SMEs and the development of a diverse range of Network Apps, it became crucial to establish a similar method for monitoring the overall development process. With multiple SMEs working on different Network Apps, it was essential to ensure consistency, track progress, and maintain quality during the lifetime of WP4. Monitoring the development process allowed for effective oversight and evaluation of each SME's activities, ensuring adherence to the defined development process and the timely delivery of results. Apart from the monitoring, the versioning process has been diligently followed for the Network Apps, ensuring a structured approach for managing and tracking their evolution. This process has played a pivotal role in supporting various aspects such as change management, compatibility management among the different components, quality control, and effective communication between developers among the technical WPs.

By adopting a robust versioning system, WP4 has successfully maintained a clear and organized record of the progression of the Network Apps. This approach has facilitated efficient development, seamless maintenance, and responsive support as it has enabled the core technical team to track and manage changes effectively, ensuring that updates and enhancements are implemented in a structured manner. This systematic approach has contributed to the overall effectiveness and productivity of WP4's activities. The following table presents the evolution of the Network Apps, focusing on the main functionalities that differentiate each version. The final version of the Network Apps within each pillar has been thoroughly described in the relevant technical deliverables, namely D4.4 “Network Apps for pillar Interaction of Employees and Machines”, D4.5 “Network Apps for FoF Operations”, D4.6 “Network Apps for Security Guarantees and Risk Analysis”, and D4.7 “Network Apps for Production Line Infrastructure”, all submitted concurrently with the current document.

Table 8 Evolution and versions of the Network Apps

Releases	Versions of Network Apps	Development process and functionalities
	Version 1	Local Development without SDK, which focused on splitting the functionality of an application between the vApp and the Network App
Release A	Version 2	Utilization of the First version of the SDK/CLI and first release of the EVOLVED-5G Network Apps communicating directly with NEF.
Intermediate release	Version 3	NEF-CAPIF integration added and therefore the Network Apps had first to be authorized via CAPIF and then discover NEF for further communication.

		Additional improvements on the vApp side were added.
Release B	Version 4	NEF-CAPIF-TSN integration was performed, and the Network Apps were capable of discovering via CAPIF both NEF and TSN APIs. Additionally, the utilization of the companion application was added, facilitating the validation of the Network Apps in indoor environments (on demand for the fourth pillar needs)
	Version 4.1 (Final Prototype)	The Network Apps were updated to the latest/final version of the EVOLVED-5G SDK and final versions of NEF-CAPIF-TSN APIs, as well as the latest version of the companion application. These final prototypes were the second and final release of the Network Apps (Rel. B)

As has been described in D4.1 the EVOLVED-5G project targeted two types of Network Apps: Stand Alone (SA) and Non-Stand-Alone (NSA) NetApp. These types are determined by how services are provided to verticals. It has been decided that EVOLVED-5G will support both modes to offer flexibility and engage with vApp developers.

In the NSA mode, SMEs delivering this mode have chosen their NetApp to act as a wrapper for Northbound APIs. The goal is to expose services through Business APIs. In this scenario, the Network App functions as an auxiliary software module that becomes operational when its business APIs are consumed by the vApp. By utilizing the 5G exposure capabilities through the business APIs, vApps can be upgraded and retrieve requested information successfully. On the other hand, SMEs opting for the SA mode integrate the NetApp directly into their vApp to leverage the 5G exposure capabilities. This approach allows for seamless integration between the Network App and the vApp. The table below serves as a reminder of the preferences of EVOLVED-5G developers regarding SA or NSA implementation and the commonly identified APIs that have been described in section 1. These preferences take into consideration the distinct characteristics and benefits offered by each use case.

Table 9 Type of Network Apps and API(s) used per-SME

Pillar	SME	NetApp Name	Network App Mode (SA/NSA)	API
IEM	IMM	<i>Remote assistance in AR Network App</i>	NSA	Monitoring_Event API
			NSA	QoS API
	INF	<i>Chatbot assistant Network App</i>	SA	Monitoring_Event API

	GMI-Aero	<i>Digital/physical twin Network App</i>	NSA	Monitoring_Event/QoS API
FoF	CAF	<i>NetMapper Network App</i>	NSA	QoS API/Moniroting Event API
	ININ	<i>Industrial grade 5G connectivity Network App</i>	SA	Moniroting Event/QoS API
	UMA	<i>Smart irrigation 5G Agriculture Network App</i>	NSA	Monitoring_Event API
	ZORTENET	<i>Anomaly Detection Network App</i>	SA	Monitoring_Event API
SEC	8BELLS	<i>Traffic Management Network App</i>	NSA	Moniroting Event/QoS API
	IQB	<i>ID Management and Access Control Network App</i>	SA	Moniroting Event API
	FOGUS	<i>5G SIEM Network App</i>	NSA	Monitoring_Event API
PLI	PAL	<i>TeleopNetwork App</i>	NSA	QoS API
	PAL/UMS	<i>Localisation Network App</i>	NSA	Monitoring_Event API

The 5G-PPP Software Network Working Group published a white paper introducing the concept of the Network App and the respective classifications of it. According to the 5G-PPP and the specific white paper, where EVOLVED-5G has actively contributed, reflecting the project innovations and concepts in its contents, the Network App ecosystem is more than the introduction of new vertical applications that have interaction capabilities. It refers to the need for a separate middleware layer to simplify the implementation and deployment of vertical systems on a large scale. Specifically, third parties or network operators can contribute to Network Applications, depending on the level of interaction and trust.

Different implementations have been conducted by the different projects considering different API types and different levels of trust between the verticals and the owner of 5G platforms.

Considering the level of interaction and trust, the Network Applications could be classified following their architectural position:

- Network Applications as part of 5G/B5G System
- Network Applications adjacent to the 5G/B5G System,
 - o still in the CSP domain, typically as part of a Network Operator network slice
 - o in interconnected (CSP / Service Provider) Domain, typically as part of a tenant / application slice

Following the level of Network Applications integration, it results three categories, as depicted in Figure 14:

- **aaS Model:** it is the model where the vertical application consumes the Network Applications as a service. The API is offered by a (Mobile / Communication) Service Provider (CSP) or a Vertical (Sector) specific Digital SP (DSP). The vertical application deployed in the vertical service provider domain. It connects with the 3GPP network systems in one or more PLMN operator domain. This type is the respective NSA Network App type according to the EVOLVED-5G definition.
- **Hybrid:** it is the model where the vertical instantiates a part of its Vertical Application in the operator domain like the EDGE. The other part remains in the vertical domain. A similar approach has been followed in TS 23.286 related to the deployment of V2X server. This is an intermediate case between the SA and the NSA Network Apps types that EVOLVED-5G defines.
- **Coupled/delegated:** it is the model where the vertical delegates its application (in short app) to the operator. The Network Applications will be composed and managed by the CSP. This approach is the SA Network App type defined by EVOLVED-5G.

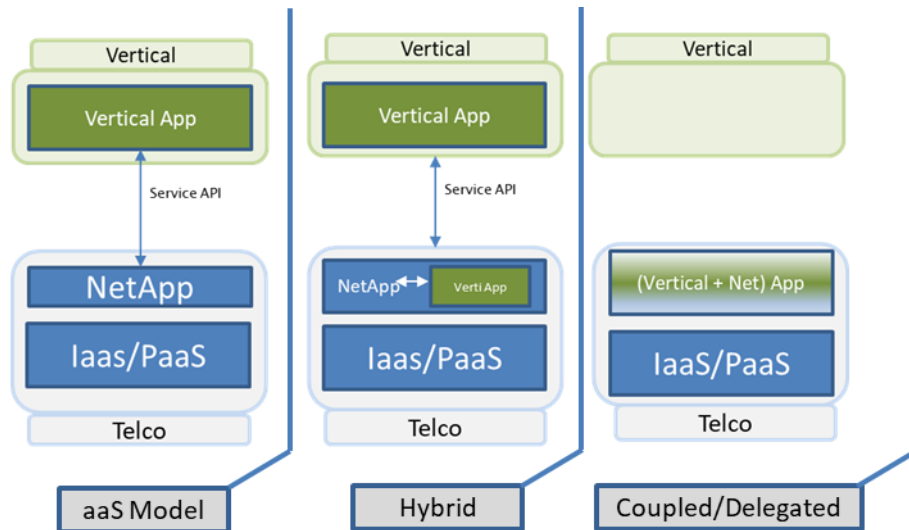


Figure 14 Network Applications classification

5 VERTICAL APPLICATION AND NETWORK APPLICATION INTEGRATION TRIALS

While Network applications have undergone thorough verification, validation, and certification within the EVOLVED-5G ecosystem as standalone components, it is important to highlight that these activities do not examine the vApp side. In this sense, recognizing the need for a holistic approach, two rounds of integration activities in the EVOLVED-5G platforms (Athens and Malaga) were performed to demonstrate the functionality of the Network app when seamlessly integrated with the vApp and to test the use case for each Network App overall. These integration activities were completed in April 2023 [M28] and after the progress of the developments the second round was completed in July 2023 [M31]. A detailed time-plan for the first and second round of integration activities combined with the other main activities of WP4 and WP5 is presented in the Figure 15 below.

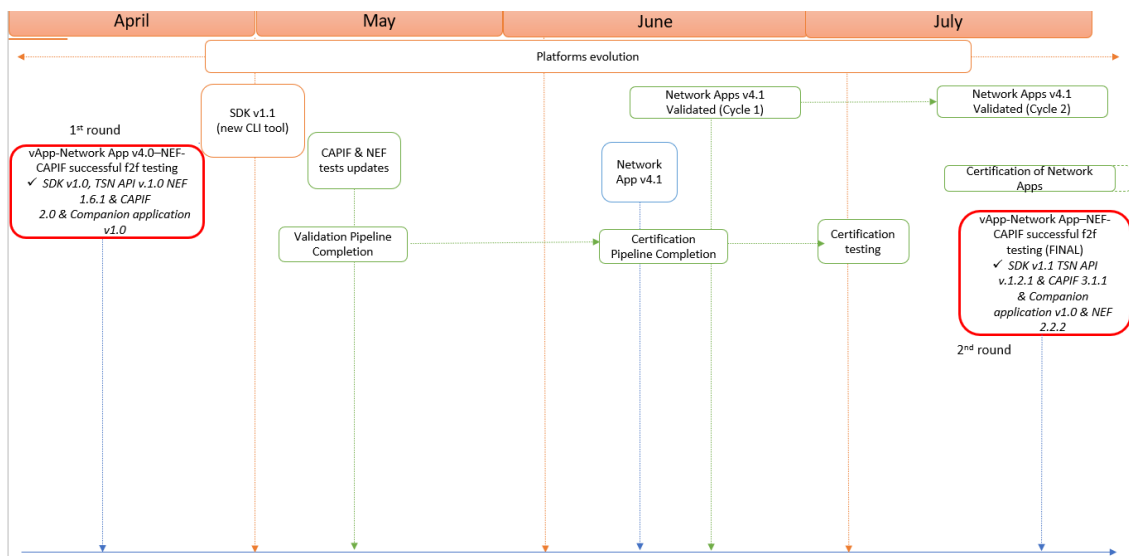


Figure 15 Timeline of first and second round integration activities between vApp and Networks Apps.

By considering the broader scope, the integration activities focus not only on evaluating and validating 5G connectivity but also on ensuring effective communication between the components of the EVOLVED-5G ecosystem. More specifically the first aim was to test the performance and reliability of the 5G network in the different scenarios that each Network App target as well as to identify any obstacles or limitations in implementing 5G connectivity and devising solutions to overcome them. Secondly the compatibility and interoperability of different the different components of EVOLVED-5G ecosystem (Network App, NEF, CAPIF, and TSN) has been tested in order to ensure that they can work together effectively without any communication gaps or conflicts. The initial round of testing utilized the Intermediate version of the Network apps, while the subsequent round was conducted using the versions included in Release B.

The following table presents the details for each of the two iterations. Apart from the different versions that have been used during the first and second round, an important difference is the fact that during the first round the cloud infrastructure of the two platforms (Athens, Malaga) has been utilised, while during the second round the K8s cluster of each platform served as the testing environment. Consequently, additional developments and configurations were applied to the Network Apps to ensure compatibility within the K8s cluster environment. The comprehensive technical details encompassing the two rounds of the integration activities, have

been extensively documented within dedicated sections in the relevant deliverables of each task, namely D4.4, D4.5, D4.6, and D4.7, submitted concurrently with the current document.



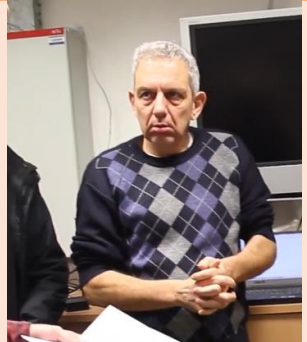


Table 10 Differences between the 2 rounds of integration testing

Specific Details related to the integration	First Round-Deployment to Open stack [12/2022-04/2023]	Second Round-Deployment to K8s [04/2023-08/2023]
EVOLVED-5G components versioning	CAPIF v2.1 – NEF v1.6.2 – SDK v0.8.9	CAPIF v3.1.2 – NEF v2.2.2 – SDK v1.1.1 – TSN 1.2.1
Deployment Options	Network App, NEF, CAPIF are deployed in separate VMs within the cloud infrastructure	<ul style="list-style-type: none"> • Network App, NEF, CAPIF are deployed within the Kubernetes cluster • Docker images for each Network Application to a public docker registry • Each Network App has created the manifest .yaml files for the deployment in Kubernetes
Network App	Intermediate release (version 3) and Final Release (version 4) of Network Apps	Final release of Network Apps (Version 4.1)
Deployment Considerations (vApp)	Depends on the use case, client connected to 5G and/or server deployed alongside other components	Same as the first round

The involvement of the SMEs in the two rounds of Network Apps and vApps integration activities proved highly beneficial for both the EVOLVED-5G ecosystem, since it resulted in evolved and enhanced versions of the various tools, and the SME itself, since improved versions of the vApp and Network App integration led to more efficient 5G engagement in the use-cases under study. Moreover, the overall successful outcome of these tests not only showcased and leveraged technical expertise of the participants, but also highlighted the practical benefits of integrating all the components of the EVOLVED-5G ecosystem so as to the various use cases.

Firstly, the initial round paved the way for improvements and further developments on both the vApp and Network App sides in order to establish connectivity with the 5G infrastructure. Secondly, the SMEs gained a clear understanding of the potential and benefits offered by utilizing the provided APIs in their use cases. This enabled them to leverage the capabilities of their already existing products and solutions, resulting in improved user experiences by harnessing the 5G network exposure capabilities.

During the onsite trials both for the first and second round, a comprehensive and informative video was crafted by the majority of the participating SMEs, aiming to elucidate the core objectives, specific aspects the process, and ultimately, the achieved outcomes. These produced videos serve as a useful tool, ready to be employed for dissemination purposes as well.

Video Link	Video Snapshot/SME Representative
<u>Internet Institute Network App deployment tests at the NCSR Athens 5G platform</u>	
<u>INFOLYSIS Intent-driven chatbot deployment tests at NCSR Athens 5G platform</u>	
<u>GMI Anita 4.0 deployment tests at NCSR Athens 5G platform</u>	
<u>IMMERSSION tests Mixed Reality for remote assistance under 5G in Malaga platform</u>	
<u>Deployment tests for the 5G enabled Security Information and Event Management system of FOGUS in Athens Platform</u>	

<u>PAL Robotics deployment tests in Malaga Platform</u>	
<u>CAFATECH deployment tests in Malaga Platform</u>	

5.1 K8S CLUSTER DEPLOYMENT OF THE SECOND INTEGRATION ROUND

For the purpose of the second integration round a new Kubernetes cluster has been designed in the Athens platform to facilitate the deployment of the components that are essential to conduct the tests. To ensure the smooth functioning of the cluster, various aspects have been taken into consideration through the creation of the cluster. The first step includes the preparation of the VMs that needed to be instantiated in the cloud infrastructure based on specific requirements and workloads. Since the workload required is negligible, a simple Kubernetes cluster with one master and one worker is adequate for proof-of-concept testing, therefore, 2 VMs with 4 vCPUs, 16 GiB RAM and 80 GB storage have been instantiated. However, more workers can be easily joined depending on the requirements.

The K8s cluster was deployed using an orchestration script, which automated the process to ensure consistency and efficiency. Additionally, the cluster was integrated with Cilium container network interface to enable seamless networking and programmability. To improve communication between services, we deployed a service mesh that includes an ingress controller based on NGINX solution that configures the NGINX load balancer according to Ingress resources. The K8s ingress resource supports content-based routing (host-based routing) and TLS termination for each hostname. Figure 16 depicts a simplified diagram of the architecture that includes the aforementioned aspects.

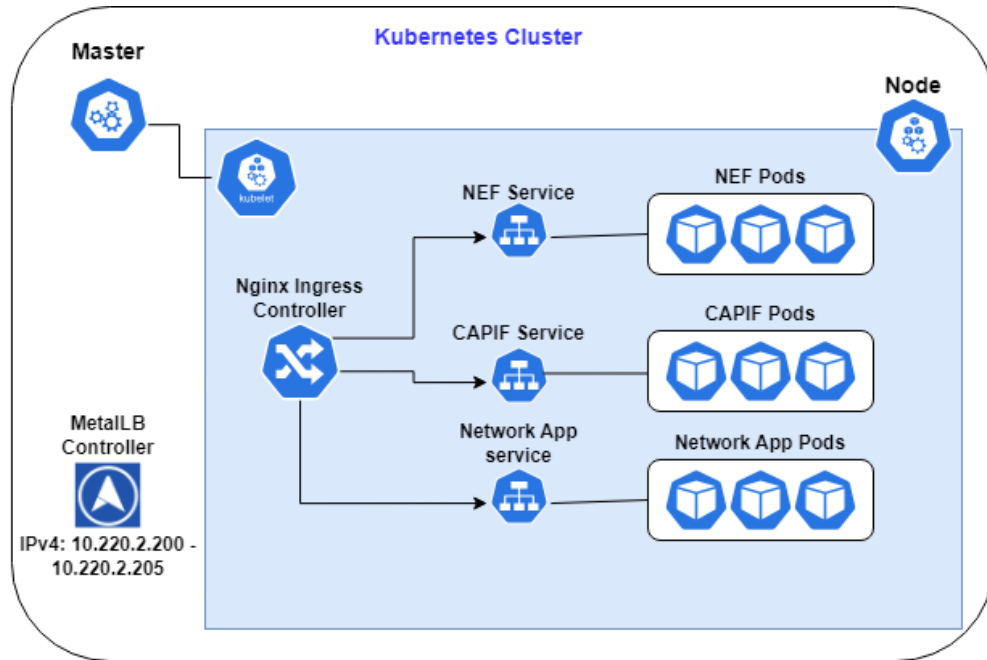


Figure 16 K8s cluster Architecture in Athens Platform

Another important aspect is the implementation of MetalLB in the K8s cluster. Kubernetes lacks an inherent implementation of network load balancers (i.e., services of type LoadBalancer in K8s require an external plugin in order to be publicly exposed under a load balancing service) for bare-metal setups, and usually the load balancing is supported by IaaS platforms (e.g., AWS, Azure). This leaves bare-metal cluster operators (i.e., this applies in both Athens and Malaga Platforms) with only “NodePort” and “externalIPs” services as alternatives to handle user traffic, both of which come with significant downsides for production use. MetalLB offers an independent network load balancer implementation that integrates with standard network equipment operating on the platforms. With MetalLB, our cluster can provide the same level of load balancer support as those running on supported IaaS platforms, enabling a more consistent and user-friendly experience for our applications and services in a bare-metal environment. As presented in Figure 16, MetalLB Controller assigns an external IP from range 10.220.2.200-205 to the NGINX ingress controller, handling all the networking configurations via L2.

For the case of Malaga’s platform, the K8s deployment didn’t undergo any major updates compared to the initial setup, in order to support the integration activities. In more detail, the Kubernetes deployment in the Málaga platform is based on a multi-master architecture configured for high availability. The deployment is composed by three master nodes and three worker nodes, with an additional node dedicated to storage. In the Málaga cluster, the nodes are accessed by means of a Load Balancer implemented with MetalLB, which allows balancing the traffic load between the services exposed in the different nodes. There is also an Ingress controller implemented with contour that allows a more elaborate way of exposing the services. A role-based policy (RBAC) is used to allow the isolation of users using the cluster. Containerd is used as container runtime, and for networking between the different elements of the system, Calico is used. In addition, KubeVirt is used for the virtualization of VMs in the system over the containerized infrastructure. Finally, for system monitoring, Prometheus is used together with Grafana for the visualization of the measurements taken by the system.

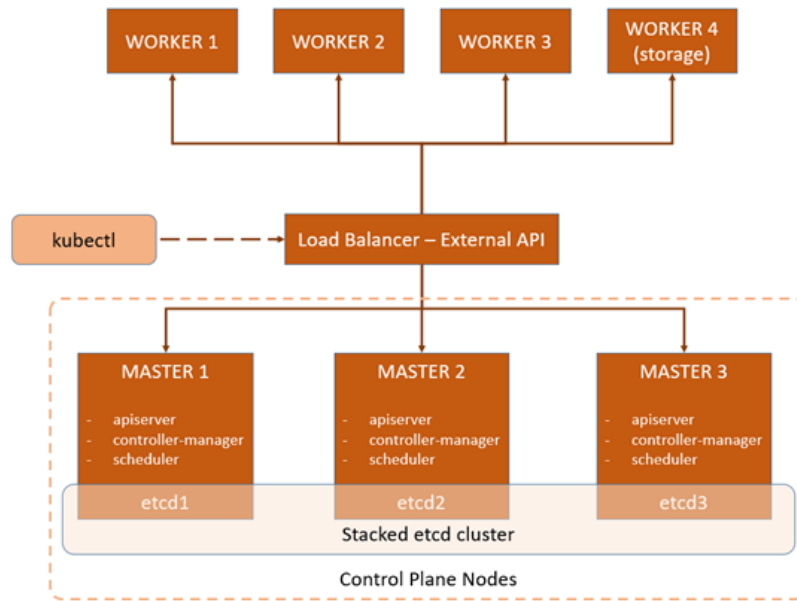


Figure 17 K8s cluster Architecture in Malaga Platform

5.2 CONSULTATION AND GUIDELINES TO THE SMES

To follow a consistent and coherent approach, a preparation phase was held to provide guidelines for the SMEs prior to the integration activities. The complete guide, which was common for both platforms, is provided below:

Step 1: Push Docker Images to a Public Docker Registry

To begin, the latest Docker images of the Network Applications must be pushed to a public Docker registry. The recommended approach is to use Docker Hub repository. <https://hub.docker.com/>.

Pushing the Docker images:

1. Tag the Docker image using the following command:
`docker tag <image-id> <your-dockerhub-username>/nef-emulator-backend:v1` Replace <image-id> with the ID of your Docker image and <your-dockerhub-username> with your Docker Hub username.
2. Log in to Docker Hub using the `docker login` command.
3. Push the tagged Docker image to Docker Hub:
`docker push <your-dockerhub-username>/nef-emulator-backend:v1` Ensure to replace <your-dockerhub-username> with your Docker Hub username.

Step 2: Creation of Manifest .yaml files for Kubernetes Deployment

Next, the creation of the manifest (.yaml) files is required for deploying Network Application in Kubernetes, including:



- **Deployment Resource File:** This file defines the deployment configuration for the Network Apps.
- **Service Resource File:** This file specifies the service configuration to expose Network Apps within the Kubernetes cluster.
- **ConfigMap Resource File (Optional):** If the Network Apps require environmental variables, config maps can be used by creating a separate file to define the configmap resource.

Step 3: Update hostnames and callback URLs in Network App's Code

To ensure seamless integration with NEF and CAPIF reverse proxies, the hostnames must be updated in the Network Application's code. The optimal way to handle this is through environment variables. Additionally, the callback URL of the Network Application must match the service name created in the manifest `service.yaml` file.

- NEF: Access NEF using the hostname `nefemu`.
- CAPIF: Access CAPIF using the hostname `capifcore`.

After successfully deploying the services and setting up the ingress controller along with ingress rules in K8s, service monitoring becomes possible. However, before proceeding with monitoring, it is essential to ensure that the following prerequisites are fulfilled. VPN access should be provided by the platform owners to establish a connection to the platforms. Moreover, in each environment where the services will be accessed, the provided hostnames should be mapped with the external IP of the ingress controller in the `/etc/hosts` file. Once these prerequisites are met, the services can be monitored using the following URLs:

- For monitoring NEF's UI: <https://validation-athens.com>
- To monitor CAPIF's database: <https://mongocapif.com>
- To monitor NEF's database: <https://mongonef.com>.

6 CONCLUSION

This deliverable, the concluding for WP4, presents in detail the work performed in the context of WP4 during the second period of the project and more specifically from M15-M32. Task 4.1 is driving the current deliverable and one of the primary purposes of the specific task, during the lifetime of the project, was to initially identify, design and ultimately improve the APIs that the Network Apps of each pillar utilised in order to reach the final prototype. The final prototype of the Network Apps within each pillar has been thoroughly described in the relevant technical deliverables, namely D4.4 “Network Apps for pillar Interaction of Employees and Machines”, D4.5 “Network Apps for FoF Operations”, D4.6 “Network Apps for Security Guarantees and Risk Analysis”, and D4.7 “Network Apps for Production Line Infrastructure”, all submitted concurrently with the current document.

Task 4.1 focuses on the 5G Exposure Capabilities for Network Apps development. In this regard, Section 2 provides a detailed description of the data flows of the implemented APIs related to the NEF. Additionally, it presents the enhancements made to those two APIs compared to the initial implementation described in D4.1. As part of T4.1, research work has been conducted to explore additional APIs and functionalities, which are also presented in the same section. Section 3 summarizes the specific tools provided to support the development process and highlights the changes compared to the first period, thus reaching the final release of these tools. In the context of WP4, integration activities focusing on evaluating and validating 5G connectivity, as well as ensuring effective communication between the components of the EVOLVED-5G ecosystem, have been performed. Section 4 presents the specific details and planning of these activities.

7 REFERENCES

- [1] 3GPP, "TS 23.502 "Procedures for the 5G System (5GS)", v17.0.0 , 2021".
- [2] 3GPP, "TS 29.522 "5G System; Network Exposure Function Northbound APIs"; Stage 3; v18.1.0, 2023".
- [3] 3GPP, "TS 29.503 "Unified Data Management Services" version 18.1.0, 2023".
- [4] 3GPP, "TS 29.503 "Unified Data Management Services", version 18.1.0, 2023".
- [5] 3GPP, "TS 29.517 "5G System; Application Function Event Exposure Service; Stage 3; v18.1.0, 2023".
- [6] EVOLVED-5G, " "D3.1 Implementations and integrations towards EVOLVED-5G framework realisation," <https://evolved-5g.eu/wp-content/uploads/2022/01/EVOLVED-5G-D3.1-v1.0.pdf>".
- [7] EVOLVED-5G, " "D4.1 5G Exposure Capabilities for Vertical Applications (Intermediate)" https://evolved-5g.eu/wp-content/uploads/2022/03/EVOLVED-5G-D4.1_v2.0-final.pdf".
- [8] 3GPP, "S 29.222 "Common API Framework for 3GPP Northbound APIs (Release 17), v17.1.0 (2021-06)".
- [9] "https://forge.3gpp.org/rep/all/5G_APIs".
- [10] EVOLVED-5G, "D3.3 "Implementations and integrations towards EVOLVED-5G framework realisation (final)" https://evolved-5g.eu/wp-content/uploads/2023/05/EVOLVED-5G-D3.3_FV.pdf".
- [11] EVOLVED-5G, "D3.4 "NetApp Certification Tools and Marketplace development (final)".