NBI APIs Realisation in the SBI
Version 4.0
28 July 2025

Security Classification: Non-Confidential

Copyright Notice

Disclaimer

Compliance Notice

The information contain herein is in full compliance with the GSMA Antitrust Compliance Policy.

This Permanent Reference Document is classified by GSMA as an Industry Specification, as such it has been developed and is maintained by GSMA in accordance with the provisions set out GSMA AA.35 - Procedures for Industry Specifications.

Table of Contents

1 Introduction

1.1 Overview

The idea of this document is creating a baseline for the realisation of Service APIs at lower levels in the OP architecture. This way partners and operators can use it as a reference implementation covering aspects such as charging, federation, gaps and recommended enablers to fill those service gaps.

1.2 Scope

This document provides guidelines to operators on how Service APIs should be implemented in the network identifying missing interfaces or enablers and aligning SDOs and the wider industry to the needs for service development.

1.3 Definitions

Term	Description
Application Profile	A set of application's thresholds for network quality (latency, jitter, loss, throughput), used by Application Providers to share the information about their application which would be relevant for network/CAMARA APIs related decision making.
CAMARA	CAMARA is an open-source project within Linux Foundation to define, develop and test the APIs
Packet Delay Budget	Maximum allowable one-way latency between the customer's device and the gateway from the operator's network to other networks (i.e., PDU Session Anchor - PSA).

1.4 Abbreviations

Term	Description
3GPP	3rd Generation Partnership Project
AAA	Authentication, Authorization, Accounting
AMF	Access and Mobility Management
AS	Application Server
API	Application Programming Interface
BSS	Business Support System
CSP	Communication Service Provider
DNN	Data Network Name
DNS	Domain Name System
EAS	Edge Application Server
EASDF	Edge Application Server Discovery Function
EWBI	East/Westbound Interface
GMLC	Gateway Mobile Location Centre
gNodeB	next generation Node B
GPS	Global Positioning System
HR	Home Routed

Term	Description
HSS	Home Subscriber Server
ICMP	Internet Control Message Protocol
ID	IDentifier
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
IPX	Internetwork Packet Exchange
KPI	Key Performance Indicator
KYC	KnowYourCustomer
MME	Mobility Management Entity
MSISDN	Mobile Station International Subscriber Directory Number
NEF	Network Exposure Function
NF	Network Function
NTF	Network Transformation Functions
NWDAF	Network Data Analytics Function
OB	Operating Business
OP	Operator Platform
OTP	One-Time Password
OWD	One-Way Delay
PDN	Packet Data Network
PDU	Packet Data Unit
PGW-C	PDN Gateway Control plane function
PRD	Permanent Reference Document
PSA	PDU Session Anchor
QoD	Quality on Demand
QoS	Quality of Service
RTT	Round-Trip Time
SBI	SouthBound Interface
SBI-NR	SouthBound Interface – Network Resources
SCEF	Service Capability Exposure Function
SDO	Standard Developing Organisation
SED	Simple Edge Discovery
SLR	Service Level Requirements
SIM	Subscriber Identity Module
SMF	Session Management Function
SMS	Short Message Service
TCP	Transmission Control Protocol
TI	Traffic Influence
UDM	Unified Data Management

Term	Description
UDP	User Datagram Protocol
UE	User Equipment
UNI	User to Network Interface
UPF	User Plane Function
URL	Uniform Resource Locator

1.5 References

Ref	Doc Number	Title
[1]	RFC 2119	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt
[2]	RFC 8174	Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words https://www.rfc-editor.org/info/rfc8174
[3]		CAMARA UE Identification https://github.com/camaraproject/Commonalities/blob/main/documentation/SupportingDocuments/UE-Identification.md
[4]		CAMARA API "CarrierBilling" https://github.com/camaraproject/CarrierBillingCheckOut
[5]	GSMA PRD OPG.07	Southbound Interface Charging Function APIs
[6]		CAMARA APIs "Device Location" https://github.com/camaraproject/DeviceLocation
[7]		CAMARA API "Location Retrieval" https://github.com/camaraproject/DeviceLocation/blob/main/code/API_definitions/location-retrieval.yaml
[8]		CAMARA API "Location Verification" https://github.com/camaraproject/DeviceLocation/blob/r1.2/code/API_definitions/location-verification.yaml
[9]	GSMA PRD OPG.03	Southbound Interface Network Resources APIs, Version 4.0
[10]		CAMARA Device Geofencing API https://github.com/camaraproject/DeviceLocation/blob/main/code/API_definitions/geofencing.yaml
[11]		CAMARA API "Number Verification" https://github.com/camaraproject/NumberVerification/blob/main/code/API_definitions/number_verification.yaml
[12]		CAMARA API "Quality on Demand" https://github.com/camaraproject/QualityOnDemand/tree/release-0.10.0
[13]	3GPP TS 29.214	Policy and Charging Control over Rx reference point

Ref	Doc Number	Title
[14]		CAMARA Edge Cloud Subproject https://github.com/camaraproject/EdgeCloud
[15]		CAMARA API “Traffic Influence” https://github.com/camaraproject/EdgeCloud/blob/v0.9.3-rc/code/API_definitions/Traffic%20Influence/Traffic_Influence.yaml
[16]		CAMARA API “KYC Match” https://github.com/camaraproject/KnowYourCustomer/blob/main/code/API_definitions/kyc-match.yaml
[17]		CAMARA API “KYC Fill-in” https://github.com/camaraproject/KnowYourCustomer/blob/main/code/API_definitions/kyc-fill-in.yaml
[18]		CAMARA API “OTP Validation” https://github.com/camaraproject/OTPvalidationAPI/blob/main/code/API_definitions/one-time-password-sms.yaml
[19]		CAMARA Device Reachability Status API https://github.com/camaraproject/DeviceStatus/blob/main/code/API_definitions/device-reachability-status.yaml
[20]		CAMARA Device Roaming Status API https://github.com/camaraproject/DeviceStatus/blob/main/code/API_definitions/device-roaming-status.yaml
[21]		CAMARA API “Simple Edge Discovery” https://github.com/camaraproject/SimpleEdgeDiscovery/blob/main/code/API_definitions/simple-edge-discovery.yaml
[22]	3GPP TS 23.501	5GS, System architecture for the 5G System, Release 18.
[23]	3GPP TS 23.502	5GS, Procedures for the 5G System, Release 18
[24]	3GPP TS 23.273	5G System (5GS) Location Services (LCS); Stage 2, Release 18
[25]		CAMARA Security and Interoperability Profile https://github.com/camaraproject/IdentityAndConsentManagement/releases/tag/r0.2.1
[26]		CAMARA Call Forwarding Signal API https://github.com/camaraproject/CallForwardingSignal/blob/r2.2/code/API_definitions/call-forwarding-signal.yaml
[27]		CAMARA API “Connectivity Insights - Application Profiles” https://github.com/camaraproject/ConnectivityInsights/blob/main/code/API_definitions/application-profiles.yaml
[28]		CAMARA API “Connectivity Insights” https://github.com/camaraproject/ConnectivityInsights/blob/main/code/API_definitions/connectivity-insights.yaml
[29]		CAMARA API “Connectivity Insights Subscriptions”

Ref	Doc Number	Title
		https://github.com/camaraproject/ConnectivityInsights/blob/main/code/API_definitions/connectivity-insights-subscriptions.yaml
[30]	3GPP TS 23.288	Architecture enhancements for 5G System (5GS) to support network data analytics services, Release 18
[31]	3GPP TS 23.503	Policy and charging control framework for the 5G System (5GS); Stage 2
[32]	GSMA PRD OPG.02	Operator Platform: Requirements and Architecture
[33]	RFC 7519	JSON Web Token (JWT) https://datatracker.ietf.org/doc/html/rfc7519
[34]	GSMA PRD OPG.10	Open Gateway Technical Realisation Guidelines
[35]	3GPP TS 29.522	5G System; Network Exposure Function Northbound APIs; Stage 3
[36]	3GPP TS 29.520	5G System; Network Data Analytics Services; Stage 3
[37]	GSMA PRD TS.43	Service Entitlement Configuration
[38]	GSMA PRD ASAC.01	Seamless Authenticator subsystem enhancement for TS.43 Operator Token
[39]		CAMARA APIs Access and User Consent Management https://github.com/camaraproject/IdentityAndConsentManagement/blob/main/documentation/CAMARA-API-access-and-user-consent.md
[40]		CAMARA QoS Profiles Mapping Table. Available at https://github.com/camaraproject/QualityOnDemand/blob/r2.2/documentation/API_documentation/QoSProfile_Mapping_Table.md
[41]		CAMARA API “SIM Swap” https://github.com/camaraproject/SimSwap/blob/main/code/API_definitions/sim-swap.yaml
[42]		CAMARA API “SIM Swap Subscriptions” https://github.com/camaraproject/SimSwap/blob/main/code/API_definitions/sim-swap-subscriptions.yaml

1.6 Conventions

“The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119 [1] and clarified by RFC8174 [2], when, and only when, they appear in all capitals, as shown here.”

2 Northbound CAMARA APIs Analysis

2.1 UE identification

Different solutions needed by several CAMARA APIs for the API invoker to identify the UE are described by CAMARA Commonalities group in [3].

To identify the UE addressed by a CAMARA API, an application provider should be able to use:

- a GPSI in any of the possible existing formats
- Optionally using the UE IP address or Ethernet MAC address (together with the application provider Identifier)

If an API invoker is providing an IP address or GPSI when invoking an API, there might be a need to translate that into a MSISDN.

In the case of using UE origin Public IP Address and Port, there is a need to map these to a subscriber identifier (Private IP, MSISDN). GSMA OPG.03 [9] defines how to properly retrieve UE Identifiers based on the Origin Public IP and Port tuple, including optional parameters like Protocol (TCP, STCP, UDP).

The alternatives proposed in [9] are:

- Operator’s uses Deterministic NAT, so each private UE IP is mapped to a specific port range of a Public IP Address of the CGNAT. This allows to use this mapping also in the reverse direction and map a public IP address and port combination back to the Private IP address and thus to the UE for which it would then be possible to determine the MSISDN.
- Network provides a User Directory where IP Address and Port tuple mappings will be stored along with the corresponding user MSISDN.

2.2 Authorization and authentication

The choice of the specific authorization flows to be used will be determined during the onboarding process, happening between the API Client and the Telco Operator exposing the API, taking into account the declared purpose for accessing the API, while also being subject to the prevailing legal framework dictated by local legislation.

It is important to remark that in cases where personal user data is processed by the API, and users can exercise their rights through mechanisms such as opt-in and/or opt-out, the use of 3-legged access tokens becomes mandatory. This measure ensures that the API remains in strict compliance with user privacy preferences and regulatory obligations, upholding the principles of transparency and user-centric data control.

The "Camara Security and Interoperability Profile" provides details on how a client requests an access token [25].

2.3 Generic error codes

The following errors codes apply to all CAMARA APIs currently in v1.0 (Number Verification, OTP SMS validation, location verification and Simple Edge Discovery).

Error code	Description
500	Server error
503	Service unavailable. Typically, the server is down
504	Request time exceeded

Table 1: Generic error codes

2.4 CarrierBilling API

The version of CAMARA CarrierBilling API which is considered in this document is ver. 0.2.0 [4].

2.4.1 Description

As defined in the CAMARA Carrier Billing Check Out sub project, this API allows application providers to request the payment of a (set of) good(s)/service(s), as well as to retrieve information about a specific payment or a list of payments.

In summary, this API provides the application provider with the ability to:

- Trigger carrier billing payment request (in one or two steps)
- Follow up of payment processing using as Payment Method Carrier Billing, i.e.: the operator performs the billing of the goods.

NOTE: The scope of this API family is limited to 4G and 5G at a first stage.

Required information for using the CarrierBilling API [4]:

- **Carrier Billing**: An online payment process which allows end-users to make purchases by charging payments against an Operating Business's (OB) Billing Systems, accordingly to the end-user's configuration in the OB. In a common usage in the industry, the payment is processed on current account balance or charged on next bill generated for the line.
- **Payment**: The process of paying for a (set of) good(s)/service(s).
- **1-STEP Payment**: Payment process performed in one phase (i.e. one action), that involves all the OB Carrier Billing Systems checking and trigger the charging request against billing systems.
- **2-STEP Payment**: Payment process performed in two phases (i.e. two actions). First action deals with payment preparation request to guarantee the reservation of the involved amount. Second action is an explicit confirmation or cancellation of the payment by the end-user. Any payment not confirmed/cancelled by a given end-user is discarded after some time to avoid inconsistency in the billing systems.

API functionality

This API allows application providers to request the payment of a (set of) good(s)/service(s), as well as to retrieve information about a specific payment or a list of payments.

The API provides several endpoints/operations:

- `/payments`: endpoint (POST) to request the payment in one step.

- A set of endpoints (POST) to request the payment in two steps:
  - `/payments/prepare`: endpoint to setup the payment reservation.
  - `/payments/{paymentId}/validate`: endpoint to validate such payment reservation.
  - /payments/{paymentId}/confirm: endpoint to confirm such payment reservation.
  - /payments/{paymentId}/cancel: : endpoint to cancel such payment reservation.
- A set of endpoints (GET) to retrieve information about a list of payments or a specific payment, identified by its specific `paymentId`:
  - `/payments: for list of payments`
  - `/payments/{paymentId}: for specific payment`
- an endpoint where the API Server can send notifications about a payment procedure, towards the `webhook.notificationUrl` when provided by API invoker.

2.4.2 SBI Realisation

No SBI-NR is available for the OP to serve the Carrier Billing API. The OP shall interpret the Carrier Billing API requests and transform them to proprietary request(s) to the operator's Business Support System (BSS).

2.4.3 Charging Aspects

The API can be classified as a "network capabilities exposure services: without impact on device’s data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.4.4 Roaming (Home Routed scenario) Aspect

The API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.4.5 Federation Aspects

It is assumed that the application provider has a relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the Leading Operator.

2.5 Device Location APIs

The Device Location APIs are [6]:

- Location Verification API
- Location Retrieval API
- Geofencing API

2.5.1 Location Retrieval API

The version of CAMARA Location Retrieval API which is considered in this document is ver. 0.3.0 [7].

2.5.1.1 Description

With this API, application client can retrieve the area where a certain end-user device is localised. The area provided in the response could be described:

- by a circle determined by coordinates (latitude and longitude) and a radius.
- by a simple polygon delimited by segments connecting consecutively an array of coordinates (points). The last point connects to the first point to delimit a closed shape bounded with straight sides.

The retrieved shape depends on the network conditions at the end-user's location and any of the supported shapes could be received.

The application client could optionally ask for a freshness of the localisation information by providing a maxAge ("I want a location not older than 600 seconds").

The result accuracy depends on the network's ability and accuracy to locate the device.

Additionally, to location information, the answer will also provide indication about the location time.

Note: Location is in most jurisdictions considered to be sensitive data and thereby consent by device owner/user must be verified before providing it to the developer.

Relevant terms and definitions

- **Device**: A device refers to any physical entity that can connect to a network and participate in network communication.
- **Area**: It specifies the geographical surface where a device may be physically located.
- **Max Age**: Maximum age of the location information which is accepted for the location retrieval (in seconds).
    - Absence of maxAge means that "any age" is acceptable for the application client. In other words, this is like maxAge=infinite. The system will return lastLocationTime in the response. If the system is not able to provide a location, an error 404 with code LOCATION_RETRIEVAL.DEVICE_NOT_FOUND is sent back.
    - maxAge=0 means that a fresh calculation is requested by the application client. If the system is not able to provide the fresh location, an error 422 with code LOCATION_RETRIEVAL.UNABLE_TO_FULFILL_MAX_AGE is sent back.
- **Last Location Time**: Last date and time when the device was localised.

API Functionality

The API exposes a single endpoint/operation:

- `/retrieve`: Retrieve where the device is localised. The operation returns:
    - a localisation defined with a circle with centre specified by the latitude and longitude, and radius for answer accuracy,
    - a timestamp about location information freshness.

2.5.1.2 SBI Realisation

The CAMARA Location Retrieval API leverages 3GPP Monitoring Event (monitoringType: LOCATION_REPORTING) as mentioned in section 2.5 of OPG.03 [9].

The flow sequence given below in Figure 1 details this functionality from a high-level point of view.

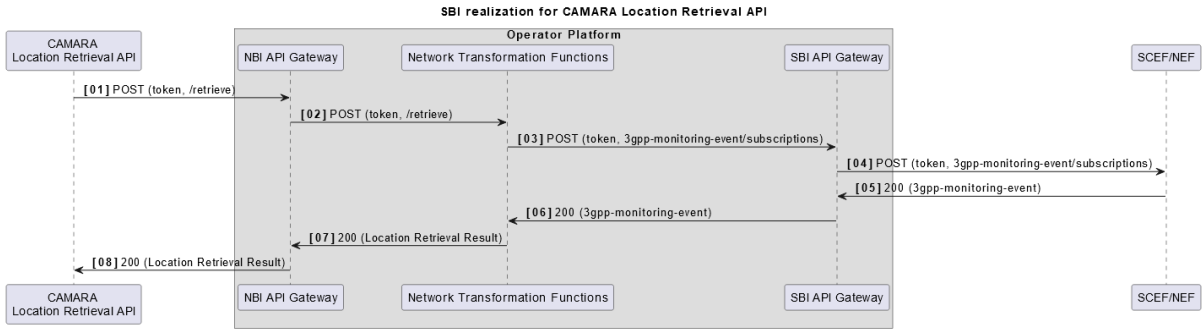

Figure 1: High level flow sequence of Location Retrieval API

Request:

CAMARA POST /retrieve	→ **NTF**	3GPP Monitoring Event POST //subscriptions

<pre>{
  "device": {
  "phoneNumber":
"33699901036"
  },
  "maxAge": 60
}</pre>
		<pre>{
		  "supportedFeatures": "4FF",
		  "msisdn": "33699901036",
		  "notificationDestination":
		"https://webhook.site/10a41b89-0d66-4bd0-
		8950-
		cfe8cd357eb9?cbInternet=true&cbEIN=false",
		  "monitoringType": "LOCATION_REPORTING",
		  "accuracy": "ENODEB",
		  "minimumReportInterval": 60,
		  "maximumNumberOfReports": 1,
		  "maximumResponseTime": 2,
		  "locationAccuracyGranularity" : "1",
		  "locationType": "LAST_KNOWN_LOCATION"
		}</pre>

Response:

The Monitoring event response is the following:

```
{
  "msisdn": "33699901036",
  "locationInfo": {
    "ageOfLocationInfo": 1,
    "cellId": "208-85-77000",
    "enodeBId": "208-85-1-300-1000",
    "trackingAreaId": "208-85-1000"
  },
  "monitoringType": "LOCATION_REPORTING",
  "locationType": "LAST_KNOWN_LOCATION"
}
```

With this answer the CAMARA service API must:

1. Check if the ageOfLocationInfo is smaller than the maxAge provided in the request (if the location is too old a specific error must be sent back)
2. Map cellId or enodeBid or trackingAreaId to an Operator's own geographic network reference to get either a circle or a polygon.

Once done, the CAMARA response API will be:

```
  "lastLocationTime": "2023-10-17T13:18:23.682Z",
  "area": {
    "areaType": "CIRCLE",
    "center": {
      "latitude": 48.801009,
      "longitude": 2.29498
    },
    "radius": 1000.0
  }
}
```

2.5.1.3 Charging Aspects

The Location Retrieval API is classified as a "network capabilities exposure services: without impact on device's data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.5.1.4 Roaming (Home Routed scenario) Aspect

Editor's Note: Roaming considerations are FFS.

2.5.1.5 Federation Aspects

Editor's Note: Federation considerations are FFS.

2.5.2 Location Verification API

The version of the Location Verification API which is mentioned in this document is version v2.0.0 in CAMARA [8].

2.5.2.1 Description

As defined in [8], the Location Verification API provides the developer with the ability to verify the location of a certain end-user device within the area specified. Currently the only area supported as input is a circle determined by a set of coordinates (latitude and longitude) and some expected accuracy (radius).

The verification result depends on the network's ability and accuracy to locate the device at the requested area.

- If the network's estimation of the device's location is fully contained within the requested area, the verification result is TRUE.
- If the network's estimation of the device's location does not overlap with the requested area at all, the verification result is FALSE.
- If the network's estimation of the device's location partially overlaps with the requested area, or it fully contains the requested area (because it is larger), the result

is 'PARTIAL'. In this case, a match_rate is included in the response, indicating an estimation of the likelihood of the match in percent.
- Lastly, the network may not be able to locate the device. In this case, the verification result is UNKNOWN.

The client may optionally include a maxAge indication. If the location information known to the server is older than the specified maxAge, an error 422 with code LOCATION_VERIFICATION.UNABLE_TO_FULFILL_MAX_AGE is sent back.

In addition:

- Absence of maxAge means "any age" is acceptable for the application client. In other words, this is like maxAge=infinite. In this case the system will still return lastLocationTime, if available.
- maxAge=0 means a fresh calculation is requested by the application client. If the system is not able to provide the fresh location, an error 422 with code LOCATION_VERIFICATION.UNABLE_TO_FULFILL_MAX_AGE is sent back.

lastLocationTime will be always included in the response unless there is no historical location information available for the device. In this case, UNKNOWN will be returned without lastLocationTime.

Location Verification could be useful in scenarios such as:

- Fraud protection, to ensure a given end-user is located in the location area claimed for financial transactions.
- Verification of GPS coordinates reported by the application on a device, to ensure the GPS was not faked, e.g. for content delivery with regional restrictions.
- Contextual-based advertising, to trigger advertising after verifying the device is in the area of interest.
- Smart mobility (vehicle/bikes renting), to confirm the location of the device and the location of the vehicle/bike to guarantee they are rented correctly.

API functionality

The API provides a single endpoint/operation:

- `/verify`: Verify whether the device location is within a requested area, currently a circle with center specified by the latitude and longitude, and radius specified by the accuracy. The operation returns a verification result and, optionally, a match rate estimation for the location verification in percent.

Identifying a device from the access token [8] (In case of difference, the CAMARA reference take precedence)

The API specification defines the device object field as optional in API requests, specifically in cases where the API is accessed using a 3-legged access token, and the device can be uniquely identified by the token. This approach simplifies API usage for API consumers by relying on the device information associated with the access token used to invoke the API.

- Handling of device information:

- Optional device object for 3-legged tokens:

  When using a 3-legged access token, the device associated with the access token must be considered as the device for the API request. This means that the device object is not required in the request, and if included it must identify the same device, therefore it is recommended NOT to include it in these scenarios to simplify the API usage and avoid additional validations.

- Validation mechanism:

  The server will extract the device identification from the access token, if available. If the API request additionally includes a device object when using a 3-legged access token, the API will validate that the device identifier provided matches the one associated with the access token.
  If there is a mismatch, the API will respond with a 403 - INVALID_TOKEN_CONTEXT error, indicating that the device information in the request does not match the token.

- Error handling for unidentifiable devices:

  If the device object is not included in the request and the device information cannot be derived from the 3-legged access token, the server will return a 422 UNIDENTIFIABLE_DEVICE error.

- Restrictions for tokens without an associated authenticated identifier:

  For scenarios which do not have a single device identifier associated to the token during the authentication flow, e.g. 2-legged access tokens, the device object MUST be provided in the API request. This ensures that the device identification is explicit and valid for each API call made with these tokens.

2.5.2.2 SBI Realization

The CAMARA Service API implementation could leverage 3GPP Monitoring Event (monitoringType : LOCATION_REPORTING)

The flow sequence given below in Figure 2 details this functionality from a high-level point of view.

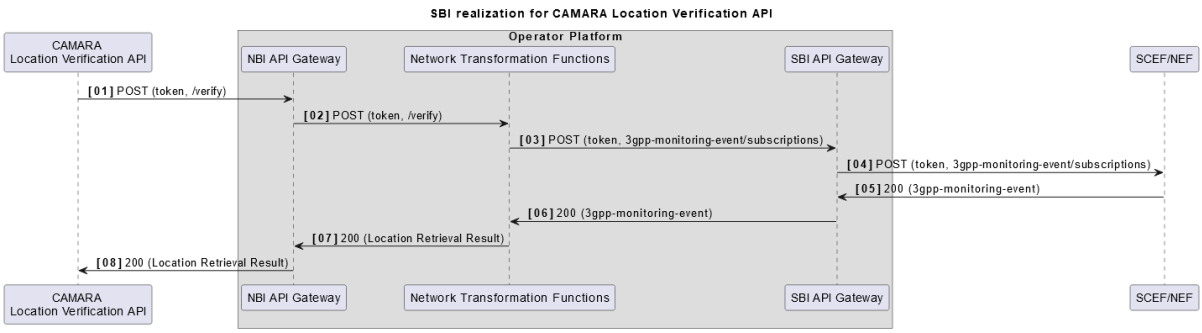

Figure 2: High level flow sequence of Location Verification API

Request:

CAMARA POST /verify	→ NTF	3GPP Monitoring Event POST /subscriptions

<pre>{
  "device": {
  "phoneNumber": "+123456789"
  },
  "area": {
  "areaType": "CIRCLE",
  "center": {
  "latitude": 50.735851,
  "longitude": 7.10066
  },
  "radius": 50000
  },
  "maxAge": 120
}</pre>
		<pre>{
		  "supportedFeatures": "4FF",
		  "msisdn": "+123456789",
		  "monitoringType":
		"LOCATION_REPORTING",
		  "locationType": "LAST_KNOWN_LOCATION"
		  "accuracy": "CGI_ECGI",
		}</pre>

Response:

3GPP Monitoring Event 200	→ NTF	CAMARA 200

<pre>{
  "msisdn": "+123456789",
  "locationInfo": {
  "ageOfLocationInfo": 1,
  "cellId": "208-85-77000",
  "enodeBId": "208-85-1-300-1000",
  "trackingAreaId": "208-85-1000"
  },
  "monitoringType":
"LOCATION_REPORTING",
  "locationType": "LAST_KNOWN_LOCATION"
}</pre>
		<pre>{
		  "verificationResult": "TRUE",
		  "lastLocationTime": "2023-09-
		07T10:40:52.000Z"
		}</pre>

For the answer to the CAMARA location verification API, the Network Transformation Functions (NTF) may involve a request to an internal operator enabler that can get the longitude and latitude coordinates of a specific cell.

2.5.2.2 Charging Aspects

The Location Verification API is classified as a "network capabilities exposure services: without impact on device's data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.5.2.3 Roaming (Home Routed scenario) Aspect

Editor's Note: Roaming considerations are FFS.

2.5.2.4 Federation Aspects

Editor's Note: Federation considerations are FFS.

2.5.2.5 Error codes

The following errors apply to Location Verification API

Error code	Description
400	This error code is returned when the application client specified an invalid argument, request body or query param (ex: MSISDN value does not comply with the schema).
401	This error code is returned when there is an authentication problem with the client request: a request without an access token, an invalid access token, an expired access token, no Authorization header in the request
403	This error code is returned when the application does not have sufficient permission (access token does not have the required scope).
404	This error code is returned when the resource is not found (ex: some identifier cannot be matched to a device).
422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code: “UNPROCESSABLE_ENTITY”: The request was well-formed but was unable to be processed due to semantic errors or not applicable values. This is the generic error code for 422 responses. "LOCATION_VERIFICATION.UNABLE_TO_FULFILL_MAX_AGE”: The system is not able to provide the fresh location required by the application client. "DEVICE_NOT_APPLICABLE": The provided device is not compatible with the requested operation, according to the service provider rules. "DEVICE_IDENTIFIERS_MISMATCH": Several device identifiers are provided but do not match the same device. “UNSUPPORTED_DEVICE_IDENTIFIERS”: Some type of device identifiers are not supported by the implementation. "UNIDENTIFIABLE_DEVICE": No device identifier provided for the device to be located. “LOCATION_VERIFICATION.AREA_NOT_COVERED”: the provided area is out of the operator's coverage or it is not supported for any reason “LOCATION_VERIFICATION.INVALID_AREA”: Legal restrictions regarding privacy, or other regulatory or implementation issues, may force the operator to set restrictions to the provided area, such as setting a minimum value to the accepted radius. In these cases
429	This error code is returned when the application client is out of resource quota or reaching rate limiting. The following response code will be added to the returned error code: "QUOTA_EXCEEDED": Request is rejected due to exceeding a business quota limit. "TOO_MANY_REQUESTS": API Server request limit is overpassed.

Table 2: Specific error codes

2.5.3 Device Geofencing API

The version of the Device Geofencing API which is mentioned in this document is version 0.3.0 in CAMARA [10].

2.5.3.1 Description

As defined in [10], developers can create applications that can receive notifications when a device enters or exits a specified area. The area provided in the request is described by a circle determined by coordinates (latitude and longitude) and an accuracy defined by the radius.

Upon successfully creating a subscription, the API will provide an Event Subscription ID, and it will indicate the subscription's expiration date.

If the geofencing-state of a device changes, the event subscriber will be notified back to the provided Notification-Url given by the subscription-request.

Device geofencing API could be useful in scenarios such as:

- Tracking devices for Presetting of Home-Settings
- Tracking of logistics

Required information for using the Device Geofencing API :

API functionality

The API provides one endpoint/operation:

- `/subscriptions`: operations to manage event subscription on geofencing events for leaving and entering an area

2.5.3.2 SBI Realization

The CAMARA Geofencing API leverages 3GPP Monitoring Event (monitoringType: LOCATION_REPORTING) as mentioned in section 2.5 of OPG.03 [9].

The flow sequence given below in Figure 3 details this functionality from a high-level point of view.

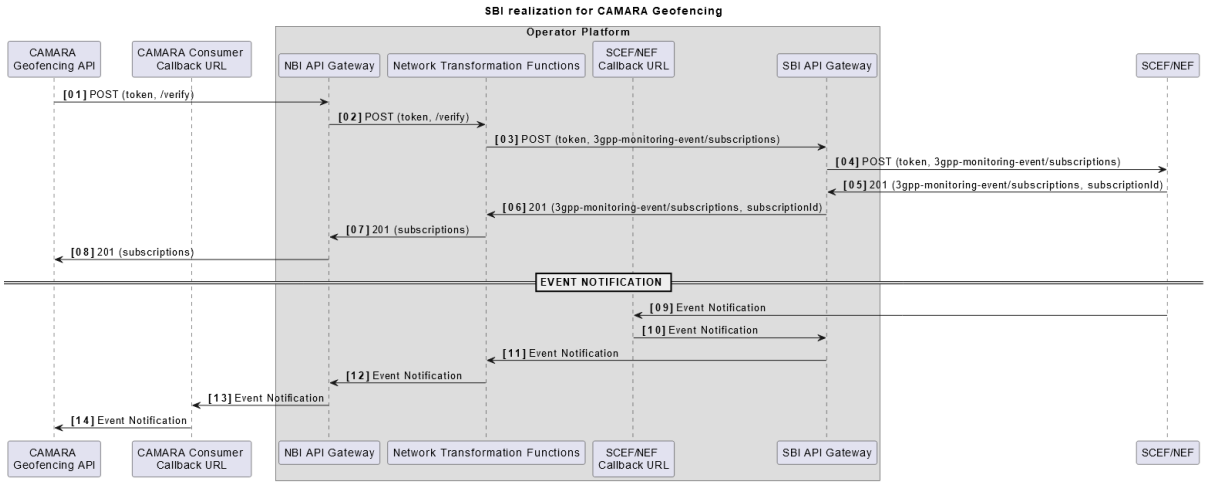

Figure 3: High level flow sequence of Geofencing API

Subscription Request:

CAMARA POST /subscription	→ **NTF**	3GPP Monitoring Event POST /subscriptions

<pre>{
  "protocol": "HTTP",
  "sink":
"https://notificationSendServer12
.supertelco.com",
  "types": [
 	
"org.camaraproject.geofencing-
subscriptions.v0.area-entered"
  ],
  "config": {
  "subscriptionDetail": {
  "device": {
  "phoneNumber":
"+12345678912"
  },
  "area": {
  "areaType": "CIRCLE",
  "center": {
  "latitude": 50.735851,
  "longitude": 7.10066
  },
  "radius": 2000
  }
  },
  "initialEvent": true,
  "subscriptionMaxEvents": 10,
  "subscriptionExpireTime":
"2024-03-22T05:40:58.469Z"
  }
}</pre>
		<pre>{
		  "supportedFeatures": "4FF",
		  "msisdn": "+12345678912",
		  "notificationDestination":
		"https:
		https://notificationSendServer12.
		supertelco.com",
		  "monitoringType":
		"LOCATION_REPORTING",
		  "accuracy": "CGI_ECGI",
		  "locationType":
		"CURRENT_LOCATION",
		"monitorExpireTime": "2024-03-
		22T05:40:58.469Z ",
		"maximumNumberOfReports": 10
		}</pre>

Based on field “types”:

- If it is “`org.camaraproject.geofencing-subscriptions.v0.area-entered`”
  - In case of `"initialEvent": true` : if the UE is already in the specified area when the subscription is requested, the answer will be sent
  - In case of `"initialEvent": false` : the answer will be sent only when the UE enter the area.
- If it is “`org.camaraproject.geofencing-subscriptions.v0.area-left`”
  - In case of `"initialEvent": true` : if the UE is already in the specified area when the subscription is requested, the answer will be sent

- In case of `initialEvent": false` : the answer will be sent only when the UE leave the area.

Response to the subscription

3GPP Monitoring Event 201 (or 202 if the answer to the request subscription is delayed i.e. to check the possibility)	→ **NTF**	CAMARA 201

<pre>{
"msisdn": "123456789",
"monitoringType":
"LOCATION_REPORTING",
"accuracy": "CGI_ECGI",
"locationType":
"CURRENT_LOCATION",
"maximumNumberOfReports": 10,
"monitorExpireTime": "2024-03-
22T05:40:58.469Z",
"notificationDestination": "https:
https://notificationSendServer12.su
pertelco.com"
}</pre>
		<pre>{
		"sink": "https:
		//notificationSendServer12.supe
		rtelco.com"
		"sinkCredential": {},
		"types": [
		"string"
		],
		"config": {
		"subscriptionDetail": {
		"device": {
		"phoneNumber":
		"+123456789",
		
		"networkAccessIdentifier":
		"123456789@domain.com",
		"ipv4Address": {
		"publicAddress":
		"84.125.93.10",
		"publicPort": 59765
		},
		"ipv6Address":
		"2001:db8:85a3:8d3:1319:8a2e:37
		0:7344"
		},
		"area": {}
		},
		"subscriptionExpireTime":
		"2024-03-22T05:40:58.469Z",
		"subscriptionMaxEvents":
		10,
		"initialEvent": true
		},
		"id": "1119920371",
		"startsAt": "2024-03-
		02T19:31:17.317Z",
		"expiresAt": "2024-03-
		22T05:40:58.469Z ",
		"status":
		"ACTIVATION_REQUESTED"
		}</pre>

For the answer to the CAMARA Geofencing API, the NTF may involve a request to an internal operator enabler that can get the longitude and latitude coordinates of a specific cell.

When an event occurs (UE entering or leaving the area), the notification event is sent from the SCEF/NEF to a notification server that forward it to the NTF.

2.5.3.3 Charging Aspects

The Geofencing API is classified as a "network capabilities exposure services: without impact on device's data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.5.3.4 Roaming (Home Routed scenario) Aspect

Editor's Note: Roaming considerations are FFS.

2.5.3.5 Federation Aspects

Editor's Note: Federation considerations are FFS.

2.6 Number Verification API

The version of the CAMARA Number Verification API which is considered in this document is ver. 2.0.0 [11].

2.6.1 Description

As defined in CAMARA Number Verification subproject, this API allows application clients to verify that a provided "mobile phone number" is the one used in the device. It verifies that the end-user is using a device with the same "mobile phone number" as is associated with the SIM used in the device connected to the mobile data network.

It also makes it possible for an application provider to verify the number whereby the phone number associated to the authenticated user's phone number is returned (enabling verification to be done by the invoker of the API/application service provider).

In summary, this API provides the application client with the ability to:

- verify in real time the validity of the phone number of the mobile device being used to access their service. The service is offered without requiring end-user interaction (e.g. no OTPs (One-Time Passwords) received by SMS nor authenticator application downloads etc).
- enhance their security checks (verified number increases the likelihood that a number is not spoofed / cloned).

NOTE: The scope of this API family is limited to 4G and 5G at a first stage.

Required information for using the Number Verification API [11]:

- **End-User Authentication:** Number Verification API requires a 3-legged token and authentication of the end user
- **Network-Based Authentication**: authentication mechanisms (of the device / UE) are based on the identification of the endpoint of a network connection (e.g. the mobile phone number associated to a specific mobile network connection / IP address).

Network-Based Authentication is implementation specific, however one possible example of an implementations is an “IP translation mechanism” whereby the received IP address is translated into information known to and stored by the MNO (related to the UE’s SIM.

- **SIM-Based Authentication**: authentication mechanism based on the identification of the subscriber's SIM installed in the user's device. This mechanism relies on temporary tokens provided by the operator, as defined by GSMA TS.43 [37] and GSMA ASAC [38].
- **PDU/PDN session:** An active PDU/PDN session is required to access this service when Network-based authentication is implemented. The mobile device requesting the service shall do so over a mobile network. SIM-based authentication is also supported over an active PDU/PDN session.
- **Wi-Fi session:** If the mobile device is connected over a Wi-Fi session to the internet, the SIM-based authentication mechanism supports access to this service.

The authentication request is supported depending on the availability of the TS.43 temporary token:

- **With a temporary token**: the API service is supported on both the PDU/PDN and Wi-Fi connections. The API consumer sends the temporary token to their backend which sends a CIBA authentication request with a parameter “`login_hint= operatortoken:`”. How the TS.43 temporary token is acquired and how this token is sent to the API consumer backend, is out-of-scope of the API definition.
- **Without a temporary token**: the mobile device must be connected to the mobile network to avail the API service. The API consumer must use OpenId Connect Authorization Code Flow as described in CAMARA APIs Access and User Consent Management [39].

API functionality

The API provides two endpoints/operations:

- `/verify`: checks if the user mobile phone number matches the phone number associated with the mobile device. It can receive either a hashed or a plain text phone number as input and it compares the received input with the authenticated end-user's phone number associated to the access token to respond true/false.
- `/device-phone-number:` retrieves the phone number associated to the end-user's token and returns it so the verification can be made by the API invoker.

2.6.2 SBI Realisation

The OP shall interpret the CAMARA Number Verification API requests and redirects it to SBI when possible. For the network side, relevant interfaces are implementation specific and subject to the network implementation of the Network-based or SIM-based authentication mechanism.

2.6.3 Charging Aspects

The Number Verify API is classified as a "network capabilities exposure services: without impact on device's data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.6.4 Roaming (Home Routed scenario) Aspect

The API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.6.5 Federation Aspects

It is assumed that the API invoker has a relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the Leading Operator.

2.6.6 Error codes

The following errors apply to Number Verification API

Error code	Description
400	This error code is returned when the application client specified an invalid argument, request body or query param (ex: MSISDN value does not comply with the schema)
401	This error code is returned when the application uses a request without an access token, an invalid access token or an expired access token
403	This error code is returned when the application client does not have sufficient permission (access token does not have the required scope). This error code will also be returned if the end-user was not authenticated via the mobile network or the MSISDN cannot be deduced from access token

Table 3: Specific error codes

2.7 QualityOnDemand API

The Service APIs for QualityOnDemand are [6]:

- Quality-On-Demand API
- QoS Profiles API
- QoD Provisioning API

2.7.1 Quality-On-Demand API

The version of the Quality-On-Demand API which is mentioned in this document is version 1.0.0 in CAMARA [18].

2.7.1.1 Description

As defined in CAMARA Quality-On-Demand (QoD), this API allows application developers and other entities consuming this capability) to request a specific packet delay budget[1] or throughput from the mobile networks without the necessity to have an in-depth knowledge of the 4G/5G system or the overall complexity of the Telecom Systems.

In summary, this API provides the application client with the ability to:

- Set quality for a mobile connection (e.g. required packet delay (latency), throughput).
- Get a notification if the network cannot fulfil the request.

The application client has a pre-defined set of Quality of Service (QoS) profiles using the QoS profile API (see § 2.7.2) which they could choose from depending on their latency or throughput requirements.

Required information for using the QoD API [12]:

- **QoD service endpoint**: The Uniform Resource Locator (URL) pointing to the RESTful resource of the QoD API.
- **Authentication**: security access keys such as OAuth 2.0 client credentials used by application clients to invoke the QoD API.
- **QoS profiles and QoS profile labels:** Latency or throughput requirements of the application mapped to relevant QoS profile values. The set of QoS Profiles that an API provider is offering may be retrieved via the QoS profile API (ref §) or will be agreed during the onboarding with the API provider.
- **Identifier for the User Equipment (UE):** At least one identifier for the user equipment out of four options: IPv4 address (public address and port), IPv6 address, MSISDN, or Network Access Identifier assigned by the mobile network operator for the user equipment (Network Access Identifier is defined for future use and will not be supported with this version of the API).
- **Identifier for the Application Server (AS):** IPv4 and/or IPv6 address of the application server (application backend)
- **Application data Flow** (between the application client and application server): The precise application data flow the application provider wants to prioritise and have stable latency or throughput for. This flow is in the current API version determined by the identifiers used for the user equipment and the application server. And it can be further elaborated with details such as ports or port ranges. Future versions of the API might allow more detailed flow identification features.
- **Duration:** Duration (in seconds) for which the QoS session (between application client and application server) should be created. Limits for session duration can be set by the implementation for the QoS profile. The end-user may request a termination before its expiration.
- **Notification URL and token:** Application client may provide a callback URL (`sink`) on which notifications about all status change events (e.g. session termination) regarding the session can be received from the MNO. This is an optional parameter.

[1] By limiting the delay, the network can provide an acceptable level of performance for various services, such as voice calls, video streaming, and data. The end-to-end or round trip latency will be about two times this value plus the latency not controlled by the operator

If `sink` is included, it is RECOMMENDED for the application client to provide as well the `sinkCredential` property to protect the notification endpoint. In the current version, `sinkCredential.credentialType` MUST be set to `ACCESSTOKEN` if provided.

API functionality

The usage of the CAMARA QoD API is based on QoS profile classes and parameters which define application data flows.

The QoS session resources can be created, queried, and deleted. The expectation is that once the QoS profile class is requested, application clients get a prioritised service with requested latency or throughput, even in the case of congestion.

The QoD API has the following characteristics:

- A specified Application data Flow (App-Flow) is prioritised to ensure stable latency or throughput for that flow.
- The prioritized App-Flow is described by providing information such as device IP address (or other device identifier) & application server IP addresses and port/port-ranges.
- The developer specifies the duration for which they need the prioritised App-flow.
- Stable latency or throughput is requested by selecting from the list of QoS profiles made available by the service provider (e.g. QOS_E) to map latency and throughput requirements.
- The API consumer can optionally also specify callback URL (`sink` parameter) on which notifications for the session can be sent.

The API provides several operations:

- `/sessions`: manages QoS session (using POST)

  When creating a session (using POST):

  - If the qosStatus in the API response is "AVAILABLE" and a notification callback is provided the API consumer will receive in addition to the response a QOS_STATUS_CHANGED event notification with qosStatus as AVAILABLE.
  - If the qosStatus in the API response is REQUESTED, the client will receive either:

    - a QOS_STATUS_CHANGED event notification with qosStatus as AVAILABLE after the network notifies that it has created the requested session, or
    - a QOS_STATUS_CHANGED event notification with qosStatus as UNAVAILABLE and statusInfo as NETWORK_TERMINATED after the network notifies that it has failed to provide the requested session.
    - A QOS_STATUS_CHANGED event notification with qosStatus as UNAVAILABLE will also be send if the network terminates the session before the requested duration expired

NOTES:

- In case of a QOS_STATUS_CHANGED event with qosStatus as UNAVAILABLE and statusInfo as NETWORK_TERMINATED the resources of the QoS session are not directly released, but will get deleted automatically at earliest 360 seconds after the event.
- This behavior allows API consumers which are not receiving notification events but are polling to get the session information with the qosStatus UNAVAILABLE and statusInfo NETWORK_TERMINATED. Before a API consumer can attempt to create a new QoD session for the same device and flow period they must release the session resources with an explicit delete operation if not yet automatically deleted.

- `/sessions/{sessionId}`: Get QoS session information (using GET), Release resources related to QoS session (using DELETE)

  When releasing resources (using DELETE):
  If the notification callback is provided and the qosStatus of the session was AVAILABLE the client will receive in addition to the response a QOS_STATUS_CHANGED event with: qosStatus as UNAVAILABLE and statusInfo as DELETE_REQUESTED.There will be no notification event if the qosStatus was already UNAVAILABLE.

- `/sessions/{sessionId}/extend`: Extend the duration of an active session
- `/retrieve-sessions`: Querying for QoS session resource information details for a device

Identifying the device from the access token

This API requires the API consumer to identify a device as the subject of the API as follows:

- When the API is invoked using a two-legged access token, the subject will be identified from the optional device object, which therefore MUST be provided.
- When a three-legged access token is used however, this optional identifier MUST NOT be provided, as the subject will be uniquely identified from the access token.

This approach simplifies API usage for API consumers using a three-legged access token to invoke the API by relying on the information that is associated with the access token and was identified during the authentication process.

2.7.1.2 SBI Realisation

The OP shall interpret the CAMARA QoD API requests and redirects it to the Service Capability Exposure Function (SCEF) or Network Exposure Function (NEF) via Southbound Interface – Network Resources (SBI-NR) as defined in GSMA PRD OPG.03 section 2.2 [9] to realise it when a request is accepted.

The below flow sequences in Figure 4 to Figure 9 detail this functionality from a high-level point of view.

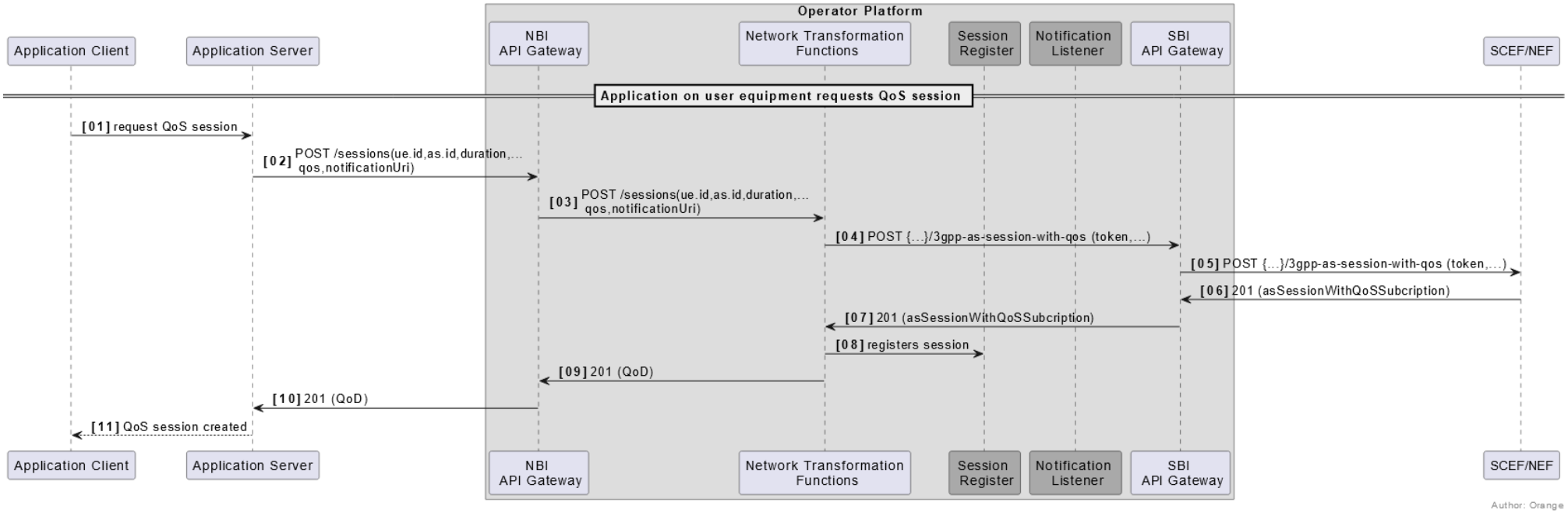

Figure 4: High level flow sequence of QoD API (QoS session request)

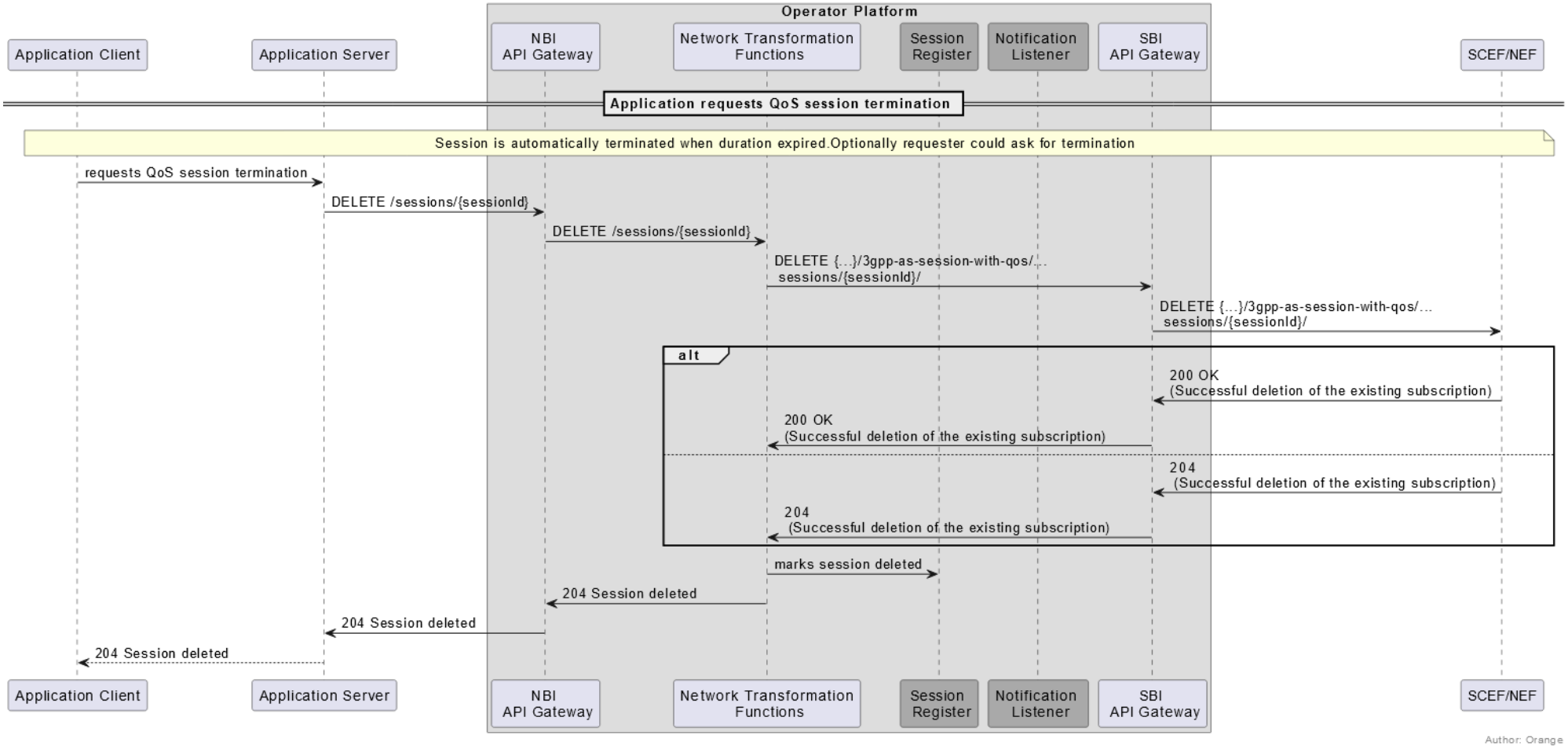

Figure 5: High level flow sequence of QoD API (QoS session termination)

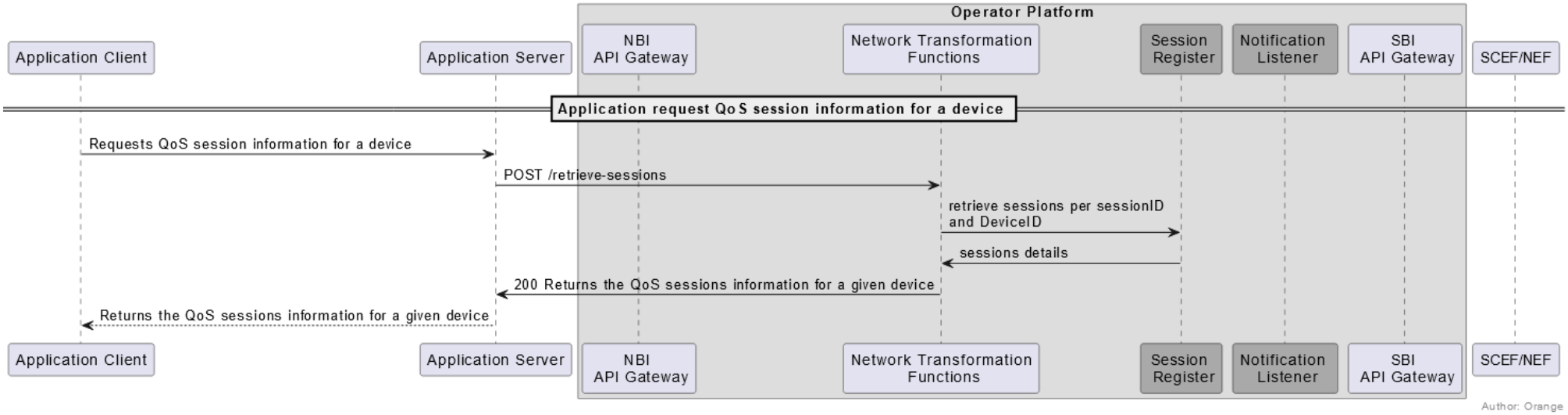

Figure 6: High level flow sequence of QoD API (QoS session information request)

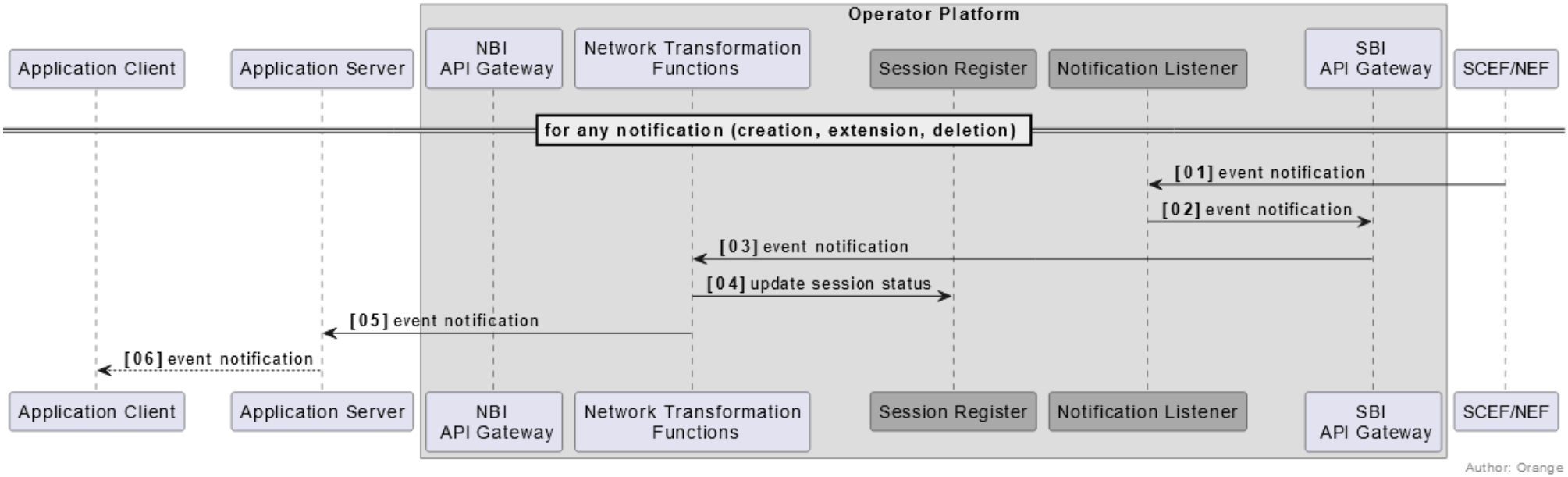

Figure 7: High level flow sequence of QoD API (notifications)

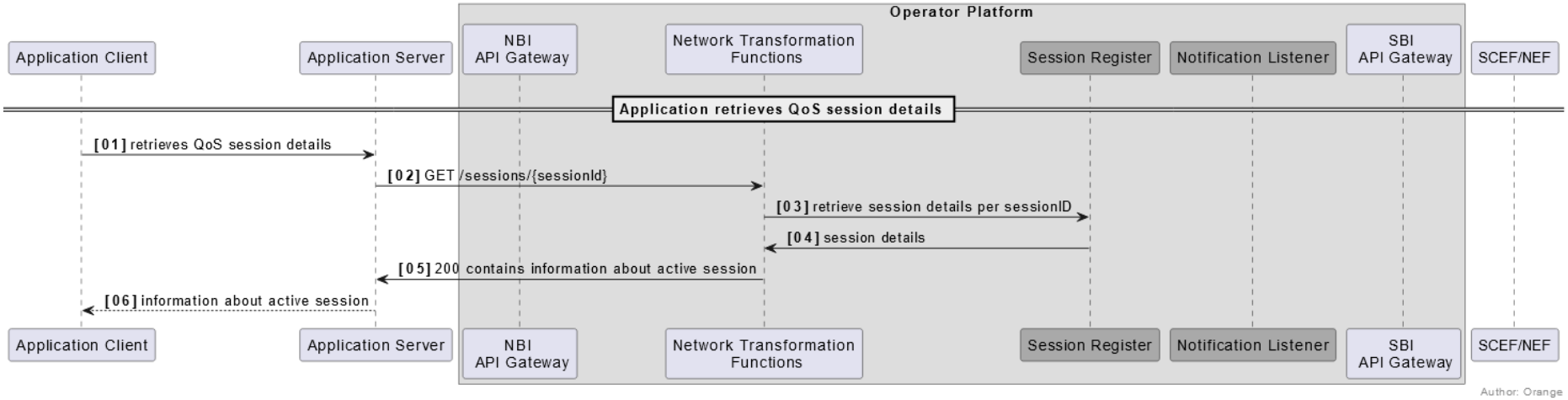

Figure 8: High level flow sequence of QoD API (QoS session retrieval)

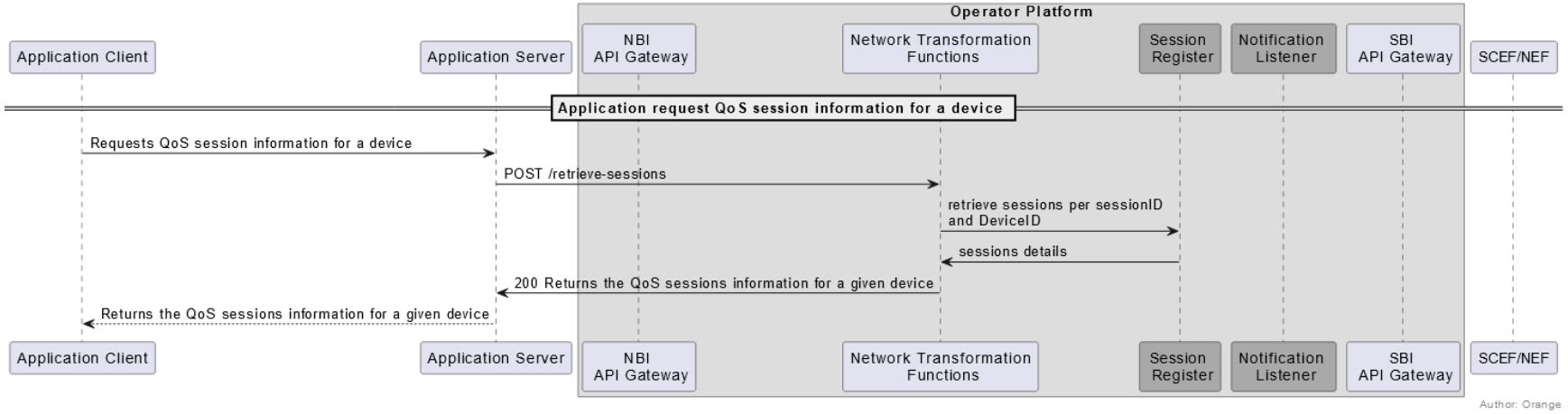

Figure 9: High level flow sequence of QoD API (QoS session retrieval for a device)

In Figure 4 to Figure 9, whether the CSP holds the information about the sessions inside or outside the OP is left as an implementation choice. Additionally, CRUD operations over the Session Register could be considered part of the Network Transformation Functions.

The realisation of the QoD request through the Operator Platform requires a transformation function. Hereafter, a simple case of mapping is detailed for the creation.

The simple case consists of a QoD API request where the IP addresses and ports represents scalar values.

- IP address with no mask or prefix
- Port represents scalar value, i.e. not a range or list

An example of this request (noted 1 in the flow in Figure 4) is the one below.

```
      {
         "duration": 60,
         "device": {
           "ipv4Address": {
             "publicAddress": "84.125.93.10",
             "publicPort": 59765

           }
         },
         "applicationServer": {
           "ipv4Address": "172.20.120.84"
         },
         "devicePorts": {
           "ports": [
             50984
           ]
         },
         "applicationServerPorts": {
           "ports": [
             10000
           ]
         },
         "qosProfile": "middle",
         "sink": "https://endpoint.example.com/sink"
         "sinkCredential": {
             "credentialType": "ACCESSTOKEN"
             "accessToken": 27284495-859a-48bc-ba2a-9a3cc0113b784
             "accessTokenExpiresUtc": 2025-07-03T12:27:08.312Z
             "accessTokenType"= "Bearer"
             },
         }
}
```

The corresponding SCEF/NEF compliant request is:

```
      {
         "supportedFeatures": "0",
         "notificationDestination":
      "https://endpoint.example.com/sink/<sinkCredential>",
         "flowInfo": [
           {
             "flowId": 17,
             "flowDescriptions": [
```

```
            "permit   in   ip   from   172.20.120.105   55289   to
      172.20.120.84 10000",
            "permit   out   ip   from   172.20.120.84   10000   to
      172.20.120.105 55289"
          ]
        }
      ],
      "qosReference": "b55e2cc8-b386-4d90-9f95-b98ba20be050",
      "ueIpv4Addr": "172.20.120.105",
      "usageThreshold": {
        "duration": 60,
        "totalVolume": 0,
        "downlinkVolume": 0,
        "uplinkVolume": 0
      },
    }
```

Comparing these two requests we can see that we have different mapping problems to solve:

- The CAMARA QoD API does not allow indicating the transport protocol. However, "ip" can be used as key word used in the protocol. In this case, the port(s) are used to describe the port(s) of any protocol if available ([13] section 5.3.8).
- qosReference must be pre-existent in the system
- the qosProfile field must be transformed into the qosReference that corresponds to the expected behaviour
- this qosReference must match the flowId injected in flowInfo according to the rules of T8 API
- supportedFeatures must be set to “0” as there is no input for it
- notificationDestination must be set to an url that contains original notificationUrl and notificationAuthToken to be able to transform and forward event coming from SCEF/NEF if necessary (if we have a notificationUrl in the request) or alternatively to a proxy

The reverse mapping could be used to map response from SCEF/NEF:

```
    {
      "self":     "https://{baseURL}/3gpp-as-session-with-qos/v1/
    /subscriptions/{subscriptionId}",
      "supportedFeatures": "0",
      "notificationDestination":
    "https://endpoint.example.com/sink",
      "flowInfo": [
        {
          "flowId": 1,
          "flowDescriptions": [
            "permit   in   ip   from   172.20.120.105   50984   to
    172.20.120.84 10000",
            "permit   out   ip   from   172.20.120.84   10000   to
    172.20.120.105 50984"
          ]
        }
      ],
      "qosReference": "173047be-d805-4df9-8cd7-9cb0b78cbc2b",
      "ueIpv4Addr": "172.20.120.105",
      ""usageThreshold": {
```

```
            "duration": 60,
            "totalVolume": 0,
            "downlinkVolume": 0,
            "uplinkVolume": 0
         },
      }
```

Into a CAMARA API compliant response:

```
      {
         "duration": 60,
         "device": {
           "ipv4Address": {
             "publicAddress": "84.125.93.10",
             "publicPort": 59765

           }
         },
         "applicationServer": {
             "ipv4Address": "172.20.120.84"
         },
         "devicePorts": {
             "ports": [
               57498
             ]
         },
         "applicationServerPorts": {
             "ports": [
               10000
             ]
         },
         "qosProfile": "low",
         "sink": "https://endpoint.example.com/sink",
         "sessionId": "16f6e90d-431a-46a1-99e4-1fee437689b8",
         "startedAt": 1698137659,
         "expiresAt": 1698137719,
         "qosStatus": "REQUESTED"
      }
```

The notification sent by the SCEF/NEF when the resource is available is as follows:

```
      {
         "transaction":     "https://{baseURL}/3gpp-as-session-with-
      qos/v1/ /subscriptions/{subscriptionId}",
         "eventReports": [
         {
             "event": "SUCCESSFUL_RESOURCES_ALLOCATION",
             "flowIds": [
               1
             ]
         }
      ]
      }
```

This notification is transformed to a CAMARA API as follows:

```
{
   "eventType": "QosStatusChangedEvent",
   "eventDetail": {
       "sessionId": "8e11ef7b-c7cd-4b8a-91cd-9948ada815e1",
       "qosStatus": "AVAILABLE"
   }
}
```

2.7.1.3 Charging Aspects

The QoD API is classified as a network capabilities exposure services with impact on the device's data usage. Refer to SBI-CHF API [5] specification for details on possible options for charging for this category.

2.7.1.4 Roaming (Home Routed scenario) Aspect

In the Home Routed (HR) roaming scenario, all end user traffic is routed back to the home network (Operator A) as depicted in the Figure 10 below.

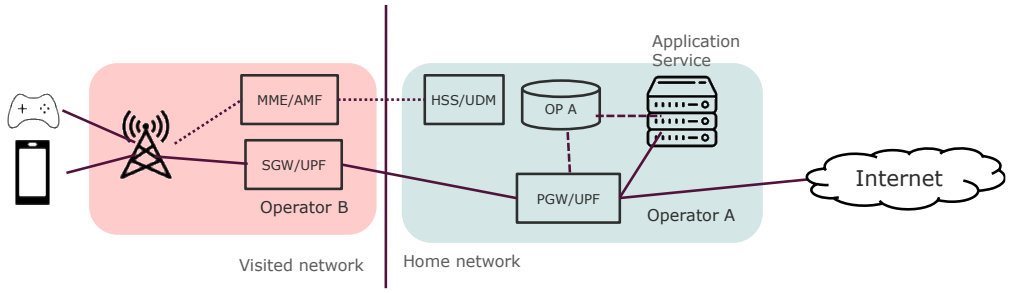

Figure 10: Simplified roaming architecture for home routed scenario

When the devices register into the visited network (Operator B), the visited network contacts home network (Operator A) and requests the service and QoS from Home Subscriber Server (HSS) in 4G or Unified Data Management (UDM) in 5G that the devices can use. Once the data sessions are established, all device data is routed back to the home network.

If an application provider or a developer requests the CAMARA QoD API, the home network is responsible for modifying the network. However, requesting a lower latency/packet delay budget may not result in better user experience. The highest latency gain is in the radio network that belongs to the Operator B, and Operator A cannot modify that. On top of it, there are Internetwork Packet Exchange (IPX) networks that introduce additional latency and are not managed by the Operator A.

2.7.1.5 Federation Aspects

In the HR roaming scenario as described in Section 2.7.1.4, there is no need to contact the visited network (Operator B) via East/Westbound Interface (EWBI).

2.7.1.6 Error codes

The following errors apply to Quality-on-Demand API

Error code	Description
400	Bad request when creating a session. The following response code will be added to the returned error code: • `/sessions`: "INVALID_ARGUMENT": the device value is an empty object, some device identifier value does not comply with the schema, invalid argument, generic syntax exception, missing request body, empty object as request body, error response for empty property in request body, error response for invalid qos profile in request body, error response when duration is not valid for selected qos profile "OUT_OF_RANGE" or "INVALID_ARGUMENT": out of range port "OUT_OF_RANGE": Specific Syntax Exception used when a given field has a pre-defined range or a invalid filter criteria combination is requested "INVALID_CREDENTIAL": invalid credential, "QUALITY_ON_DEMAND.DURATION_OUT_OF_RANGE": The requested duration is out of the allowed range for the specific QoS profile, "INVALID_TOKEN" or "INVALID_ARGUMENT": invalid token • `/sessions/{sessionId}` and `/retrieve-sessions`: "INVALID_ARGUMENT": Invalid Argument. Generic Syntax Exception ("`sessionId`" has not a UUID format), "OUT_OF_RANGE": Specific Syntax Exception used when a given field has a pre-defined range or a invalid filter criteria combination is requested, • `/sessions/{sessionId}/extend`: "INVALID_ARGUMENT": Invalid Argument. Generic Syntax Exception (request body is set to any value which is not compliant with the schema, "`sessionId`" has not a UUID format, Missing request body, Empty object as request body), "OUT_OF_RANGE": Specific Syntax Exception used when a given field has a pre-defined range or a invalid filter criteria combination is requested
401	This error code is returned when there is an authentication problem with the client request. The following response code will be added to the returned error code: "UNAUTHENTICATED": no header "Authorization", invalid access token "UNAUTHENTICATED" or "AUTHENTICATION_REQUIRED": expired access token
403	This error code is returned when the application client does not have sufficient permission. The following response code will be added to the returned error code: "PERMISSION_DENIED": Missing access token scope, QoS session not created by the API client given in the access token (for Get QoS session information, release resources related to QoS session and duration extension)
404	This error code is returned when resource is not found. The following response code will be added to the returned error code: • `/sessions` and `/retrieve-sessions`: "IDENTIFIER_NOT_FOUND": Some identifier cannot be matched to a device, • `/sessions/{sessionId}` and `/sessions/{sessionId}/extend`: "NOT_FOUND": `sessionId` of a no existing QoS session
409	This error code is returned when there is a conflict. The following response code will be added to the returned error code: • `/sessions`:

Error code	Description
	"CONFLICT": QoD session already exists for that device • `/sessions/{sessionId}/extend`: "QUALITY_ON_DEMAND.SESSION_EXTENSION_NOT_ALLOWED": Extending duration for session with qosStatus not available. Not defined for `/sessions/{sessionId}` and `/retrieve-sessions`
422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code: • `/sessions` and `/retrieve-sessions`: “MISSING_IDENTIFIER”: the subject cannot be identified from the access token and the optional device object is not included in the request, “UNNECESSARY_IDENTIFIER”: the subject can be identified from the access token and the optional device object is also included in the request. This will be the case even if the same device is identified by these two methods, as the server is unable to make this comparison. “"UNSUPPORTED_IDENTIFIER": None of the provided device identifiers is supported by the implementation, "SERVICE_NOT_APPLICABLE": Service not available for the device "IDENTIFIER_MISMATCH": Device identifiers mismatch Not defined for `/sessions/{sessionId}` and `/sessions/{sessionId}/extend`
429	This error code is returned when the application client is out of resource quota or reaching rate limiting. The following response code will be added to the returned error code: "QUOTA_EXCEEDED": Request is rejected due to exceeding a business quota limit. "TOO_MANY_REQUESTS": API Server request limit is overpassed.

Table 4: Specific error codes

2.7.2 QoS Profiles API

The version of the QoS Profiles API which is mentioned in this document is version 1.0.0 in CAMARA [18].

2.7.2.1 Description

The QoS Profiles API provides the following functionalities:

- Discover all QoS profiles offered by the API provider
- Discover the available QoS profiles for a specific device
- Retrieve the characteristics of a specific QoS profile by name

How QoS profiles are mapped to connectivity characteristics are subject to agreements between the API provider and the API consumer. An example sample for such a mapping of QoS profiles can be found in CAMARA QoS Profiles Mapping Table [40].

The API provides two operations:

- `/retrieveQoSProfile`: Retrieve QoS profiles that match the given criteria

- `/qos-profiles/{name}`: Returns a QoS Profile that matches the given name

Identifying the device from the access token

The two operations defined by this API both allow the API consumer to use a three-legged access token to specify a target device as a query filter. When provided, only profiles available to the specified target device will be returned. Additionally, when a two-legged access token is used, the operation retrieveQoSProfiles allows the API consumer to optionally specify the target device in the request body.

Hence, for the retrieveQoSProfiles operation:

- When invoked using a two-legged access token, the optional input device object will be used as filter if present.
- When invoked using a three-legged access token, this optional identifier in the request body MUST NOT be provided, as the device to filter will be uniquely identified from the access token.

2.7.2.2 SBI Realisation

No SBI-NR is available for the OP to serve the QoS Profiles API. The OP shall interpret the OTP SMS API requests and transform them to proprietary request(s) to the dedicated endpoint of the resource server.

The following figure is the generic high-level flows of this API.

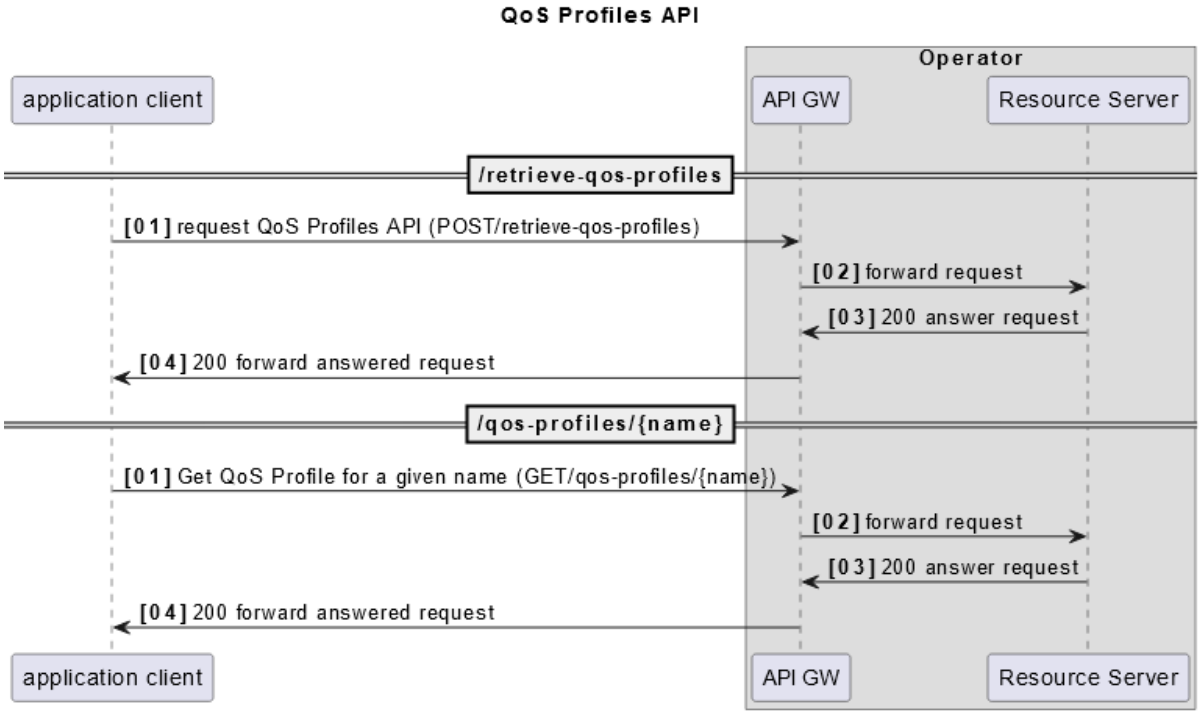

Figure 11: High level flow sequence of QoS Profiles API

2.7.2.3 Charging Aspects

The QoS Profiles API is classified as a “network capabilities exposure services: without impact on device’s data usage”. Refer to SBI-CHF API specification in GSMA PRD OPG.07 [5] for details on possible options for charging for this category.

2.7.2.4 Roaming (Home Routed scenario) Aspect

Editor’s Note: Roaming considerations are FFS.

2.7.2.5 Federation Aspects

Editor’s Note: Federation considerations are FFS.

2.7.2.6 Error codes

The following errors apply to QoS Profiles API

Error code	Description
400	Bad Request (Invalid Argument. Generic Syntax Exception, Out of Range. Specific Syntax Exception used when a given field has a pre-defined range or a invalid filter criteria combination is requested)
401	Unauthorized (request cannot be authenticated, new authentication is needed, authentication is no longer valid)
403	Forbidden (Permission denied. OAuth2 token access does not have the required scope or when the user fails operational security)
404	Resource is not found, Device identifier not found

422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code:
	• `/retrieveQoSProfile`: Unprocessable Content
	  • Inconsistency between identifiers not pointing to the same device: IDENTIFIER_MISMATCH error code
	  • Service not applicable for the provided identifier:
	  • None of the provided identifiers is supported by the implementation
	  • An explicit identifier is provided when a device or phone number has already been identified from the access token (UNNECESSARY_IDENTIFIER error code)
	• `/qos-profiles/{name}`: An explicit identifier is provided when a device or phone number has already been identified from the access token (UNNECESSARY_IDENTIFIER error code)

429	Request is rejected due to exceeding a business quota limit, Access to the API has been temporarily blocked due to rate or spike arrest limits being reached

Table 5: Specific error codes

2.7.3 QoD Provisioning API

This API is in initial state in CAMARA. It will be included when it reaches stable state.

2.8 Traffic Influence API

The version of the Traffic Influence API which is mentioned in this document is version v0.9.2 in CAMARA [15].

2.8.1 Description

The Traffic Influence API is a Service API developed in the CAMARA Edge Cloud subproject [14].

The reference scenario foresees a Service, composed by one or more Service Producers deployed in different geographical locations on Edge Cloud Zones (Edge datacentres of Telco Operator) or in Cloud. The Service Producer, deployed at the Edge, is referred as Edge Application Server (EAS).

An Edge Cloud Zone is a platform in the Telco Operator network, offering network, compute and storage resources to application providers. An application provider can deploy and run applications on an Edge Cloud Zone, meaning reduced latency to end-users that are nearby, as the network path is shorter. A network operator's EdgeCloud may comprise multiple Edge Cloud Zones, each in a discrete location to bring latency benefits to end-users across a country. The Operator can help application providers to know which of the Edge Cloud Zones will bring the best performance benefit for a given end-user and application.

The Traffic Influence API (TI API) provides the fastest routing from the end-user device (e.g. a Smartphone) to the optimal EAS instance in a specific geographical location, installed in an Edge Cloud Zone.

If a Service is offered by Cloud Instances and by Edge Instances, the TI API can be used get the optimal routing of the traffic to the Edge Instances, maybe for a set of end-users. Getting the optimal routing can be used to improve latency maybe in combination with other CAMARA APIs such as QoD (Quality On Demand). Providing the optimal routing is indeed an important step to get the optimal latency.

If the TI API is used to get the best routing at the Edge for a device in a geographical location and the device moves to another geographical location, the TI API can be invoked to get the optimal routing in the new geographical location for that device.

The TI API provides the capability to establish the best routing, in terms of latency, in a specific geographical area, between the end-user device, e.g. the end-user’s smartphone, and the optimal EAS instance nearby. If the device moves in a different geographical location where a more suitable EAS instance is available, the TI API can be invoked to influence the device connectivity to get the optimal routing to the that local instance. It is important to notice that it is a task of the Application invoking the TI API to detect the changes in the device location.

The generic architecture for the Service can foresee some Cloud instances of the Application, one or more Edge Instances of the Application. A component of the Service is the TI API Consumer. This logical component can be integrated in other components of the Service to invoke the TI API, creating a "TrafficInfluence" resource specifying the desired intent.

The TI API Producer implements the intent specified in the "TrafficInfluence" resource.

While the TI API can be invoked to activate the fastest routing for any end-user, it can also be used to request the best routing for a specific end-user. Invoking the TI API for each end-

user, many "TrafficInfluence" resources are created for each end-user to provide the requested routing for a set of end-users.

The same approach is used for the geographical locations where the influence of the traffic must be applied. Invoking the TI API without specifying a geographical area activates the optimal routing toward any EAS instance, while invoking the TI API specifying a geographical area activates the optimal routing only toward the EAS instance located closest to that geographical area. To activate the optimal routing in selected geographical areas, the TI API must be invoked for each geographical area.

The TI API can be used to:

- optimise the routing when devices need to consume the service provided by a local EAS Instances.
- effect an already established device routing when the device moves among different geographical locations.

When the TI API consumer detects a device has entered a geographical location where an EAS instance is available, it can invoke the TI API to get the optimal routing toward that EAS instance. If the device moves to another geographical location, served by another EAS instance, the routing might not be optimal anymore for the new EAS instance. In the case the Application detects a location change, it can invoke the TI API again to request a new routing optimization toward the new EAS instance.

In summary, TI API provides the capability to establish the best routing, in terms of latency, in a specific geographical area, between the end-user device, e.g. the smartphone, and the optimal EAS instance nearby. If the Device moves in a different geographical location where a more suitable EAS instance is available, the TI API can be invoked to influence the Device connectivity to get the optimal routing to the that local instance.

The usage of the TI API is based on the management of a "TrafficInfluence" resource, an object containing the intent requested invoking the TI API and that is implemented by the platform configuring the Mobile Network for the optimal routing toward the EAS instance.

Required information for using the Traffic Influence API:

- **Base-Url:** The RESTful TI API endpoint.
- **Authentication:** Security access keys such as OAuth 2.0 client credentials used by applications to invoke the Traffic Influence API.
- **trafficInfluenceID:** Identifier for the Traffic Influence resource. This parameter is returned by the TI API and can be used to update it.
- **apiConsumerId:** Unique identifier for the TI API Consumer.
- **region and zone:** The geographical area to which the developer requires the fastest routing toward the nearest application instance. A Region is a wide geographical area and it contains one or more Zones. A Zone is where the Telco Edge sites are located.
- **applicationId:** Unique Application identifier inside the Telco Operator Platform. This identifier is provided during the application onboarding process.
- **instanceId:** Unique identifier generated by the Operator Platform to identify a specific instance of the Application on a specific zone.

- **trafficFilters:** The Application can expose different service on different interfaces, with this parameter it is possible to enable just some of those services maybe for different sets of users.
- **device:** A end-user device for which the routing toward the selected Application instance is applied.
- **notification URL and token:** The call back URL on which notifications regarding the subscription can be received from the service provider.

2.8.2 SBI Realization

The OP shall interpret the CAMARA Traffic Influence API requests and redirect it to the Network Exposure Function (NEF) via Southbound Interface – Network Resources (SBI-NR) as defined in GSMA PRD OPG.03 section 2.2 [9] to realise it when a request is accepted.

The flow sequence given below in Figure 12 details this functionality from a high-level point of view.

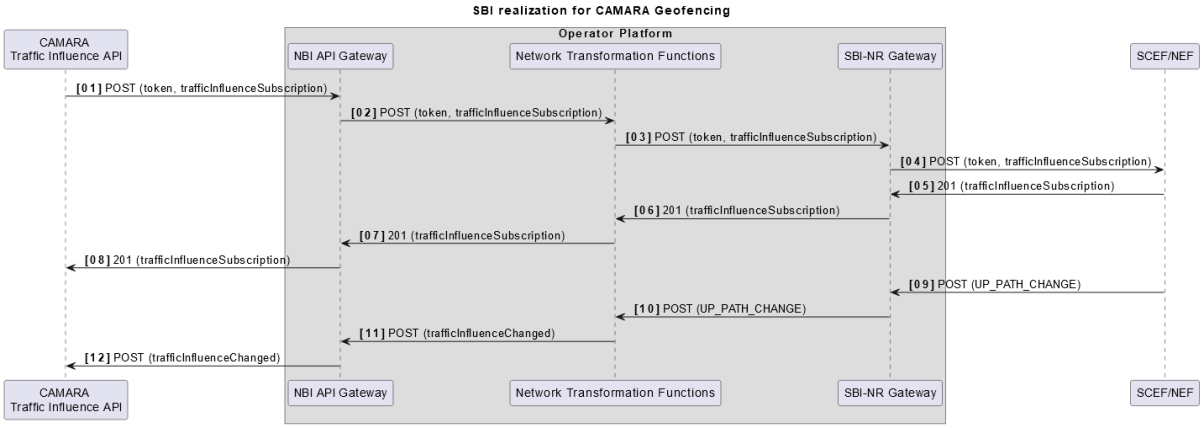

Figure 12: High level flow sequence of Traffic Influence API

The realisation of the Traffic Influence request through the Operator Platform requires a transformation function. A simple case of mapping in the transformation function is detailed below.

In the example used here, a Traffic Influence API request contains the device identifiers like MSISDN and IP Address, the required traffic filters and the information about the edge to which the traffic is routed using the zone and edge attributes.

An example of this request and the mapping to the corresponding NEF compliant request is given below.

CAMARA Request	NTF	3GPP Traffic Influence Request to NEF

<pre>{
"apiConsumerId": "Consumer123",
"applicationId": "ConsumerAppID001",
"instanceId": "Zi43ZVDN3Tm",
"region": "eu-west-3",
"zone": "eu-west-3 Local Zone1",
"device": {
"phoneNumber": "9876543219",
"networkAccessIdentifier": null,
"ipv4Address": "10.60.0.2",
"ipv6Address": null
},
"trafficFilters": [
"ip 80"
],
"notificationUri":
"https://8b4b1508-4815-4764-81db-999a08e51e39.mock.pstmn.io/event"
}</pre>
		<pre>{
		"afServiceId": "Consumer_AF_Service_001",
		"afAppId": "ConsumerAppID001",
		"afTransId": "1710477652865",
		"dnn": " dnn01",
		"gpsi": "9876543219",
		"ipv4Addr": "10.60.0.2",
		"notificationDestination": "https://{baseURL}/af/3gpp-traffic-influence/v1/uppathchg/notifications",
		"trafficFilters": [
		{
		"flowDescriptions": [
		"permit in ip from 10.60.0.2 to 192.168.1.100 80"
		],
		"flowId": 1
		}
		],
		"trafficRoutes": [
		{
		"dnai": "edge01.dnn01.com",
		"routeInfo": {
		"ipv4Addr": "100.128.10.1",
		"portNumber": 123
		}
		}
		],
		"suppFeat": "0"
		}</pre>

The above table shows the key mapping to be taken care of in the platform.

- The applicationId can be used to map the afAppId and the afServiceId.
- The applicationId may be used to map the DNN on which the application server is running.
- The traffic filters for NEF are created using the Protocol Type and the Destination Port parameters given in the request, IP Address of the device as the source IP Address, and application server IP address retrieved using the applicationId and InstanceIdentifier parameters as the destination IP Address.
- The traffic route details are mapped from the Region and Zone parameters given in the request.

When the NEF responds back, the response is reverse mapped to the CAMARA compliant response, as given below.

3GPP Traffic Influence Response from NEF	 NTF	CAMARA Response

<pre>{
"self":"https://afservice:8050/af/v1/subscriptions/61",
"afServiceId":"Consumer_AF_Service_001",
"afAppId": "ConsumerAppID001",
"afTransId": "1710477652865",
"dnn": "dnn01",
"gpsi": "9876543219",
"ipv4Addr": "10.60.0.2",
"trafficFilters": [
{
"flowDescriptions": [
"permit in ip from 10.60.0.2 to 192.168.1.100 80"
],
"flowId": 1
}
],
"trafficRoutes": [
{
"dnai": "edge01.dnn01.com",
"routeInfo": {
"ipv4Addr": "100.128.10.1",
"portNumber": 123</pre>
		<pre>{
		"trafficInfluenceID": "76",
		"apiConsumerId": "Consumer123",
		"applicationId":"ConsumerAppID001",
		"instanceId": "Zi43ZVDN3Tm",
		"region": "eu-west-3",
		"zone": "eu-west-3 Local Zone1",
		"device": {
		"phoneNumber": "9876543219",
		"networkAccessIdentifier": null,
		"ipv4Address": "10.60.0.2",
		"ipv6Address": null
		},
		"state": "created",
		"trafficFilters": [
		"ip 80"
		],
		"notificationUri": "https://8b4b1508-4815-4764-81db-999a08e51e39.mock.pstmn.io/event",
		"notificationAuthToken": null
		}</pre>

<table>
<tr>
<td>

<pre>
      }
    }
 ],
 "suppFeat": "0"
}
</pre>

</td>
<td></td>
<td></td>
</tr>
</table>

The Notification that NEF sends when the traffic route is modified (User Plane Path Changed) is reverse mapped to the CAMARA compliant response, is given below.

<table>
<tr>
<td>Notification from NEF when the UP_PATH_CHG event is triggered</td>
<td> 
NTF</td>
<td>CAMARA Notification</td>
</tr>
<tr>
<td>

<pre>
{
    "afTransId": "1710477652865",
    "dnaiChgType": "EARLY",
    "subscribedEvent":
"UP_PATH_CHANGE",
    "targetTrafficRoute": {
        "dnai": "edge01.dnn01.com
",
        "routeInfo": {
            "ipv4Addr":
"100.128.10.1",
            "portNumber": 123
        }
    },
    "gpsi": "9876543219",
    "srcUeIpv4Addr": "10.60.0.2",
    "tgtUeIpv4Addr": "10.60.0.2",
    "targetDnai":
"edge01.dnn01.com ",
}
</pre>

</td>
<td></td>
<td>

<pre>
{
  "trafficInfluenceChanged": {
    "trafficInfluenceID": "76",
    "apiConsumerId":
        "Consumer123",
    "applicationId":
        "ConsumerAppID001",
    "instanceId": "Zi43ZVDN3Tm",
    "region": "eu-west-3",
    "zone": "eu-west-3 Local Zone1",
    "device": {
        "phoneNumber": "9876543219",
        "networkAccessIdentifier":
         null,
        "ipv4Address": "10.60.0.2",
        "ipv6Address": null
    },
    "state": "active",
    "trafficFilters": [
       " ip 80"
    ],
    "notificationUri":
     "https://8b4b1508-4815-4764-
     81db-
     999a08e51e39.mock.pstmn.io/even
     t",
    "notificationAuthToken": null
   }
</pre>

</td>
</tr>
</table>

		}

2.8.3 Charging Aspects

The TI API is classified as “network capabilities exposure services with impact on the device’s data usage”. Refer to SBI-CHF API [5] specification for details on possible options for charging for this category.

2.8.4 Roaming (Home Routed scenario) Aspect

Editor’s Note: Roaming considerations are FFS.

2.8.5 Federation Aspects

Editor’s Note: Federation considerations are FFS.

2.9 KYC Match API

The version of CAMARA KYC Match API which is considered in this document is ver. 0.3.0 [16].

2.9.1 Description

As defined in CAMARA KnowYourCustomer (KYC) sub project, KYC Match API provides the API invoker with the ability to compare the information it has for a particular end-user with that on file (and verified) by the end-user's Operator in their own KYC records, in order for the API invoker to confirm the accuracy of the information and provide a specific service to the end-user. The information can include phone number, name, postal code, address, birthdate, email address etc. No Personal Identifiable Information (PII) is returned.

The API is intended to be used in the following scenarios, for example:

- To verify the end-user personal data during the digital registration of an account to a 3rd party service.
- To prevent fraud, wrong or imprecise information, and/or facilitate the onboarding of an end-user to a 3rd party service.

Required information for using this API:

- Authentication/Authorisation/End-User Consent capturing: use of the OpenID Connect (OIDC) as security scheme.

API functionality

The API exposes a single endpoint/operation:

- / `match`: Verify the matching of a number of attributes related to an end user identity against the account data bound to their phone number. The operation returns:
    - Match Result which is boolean values (True, False, not available) for each of the requested attributes.

- Match Score which is a numerical value that quantifies the similarity between the input value and the value stored in the operator's system for string-type attributes. The score will use the Jaro-Winkler distance algorithm [16] unless otherwise captured in the implementation guidelines published to API invokers by operator. This parameter, as optional, will be returned depending on the capability of operator. This shall only be returned when the Match Result is false.

2.9.2 SBI Realization

No SBI-NR is available for the OP to serve the KYC Match API. The OP shall interpret the KYC Match API requests and transform them to proprietary request(s) to the resource server.

The following figure is the generic high-level flows of this API.

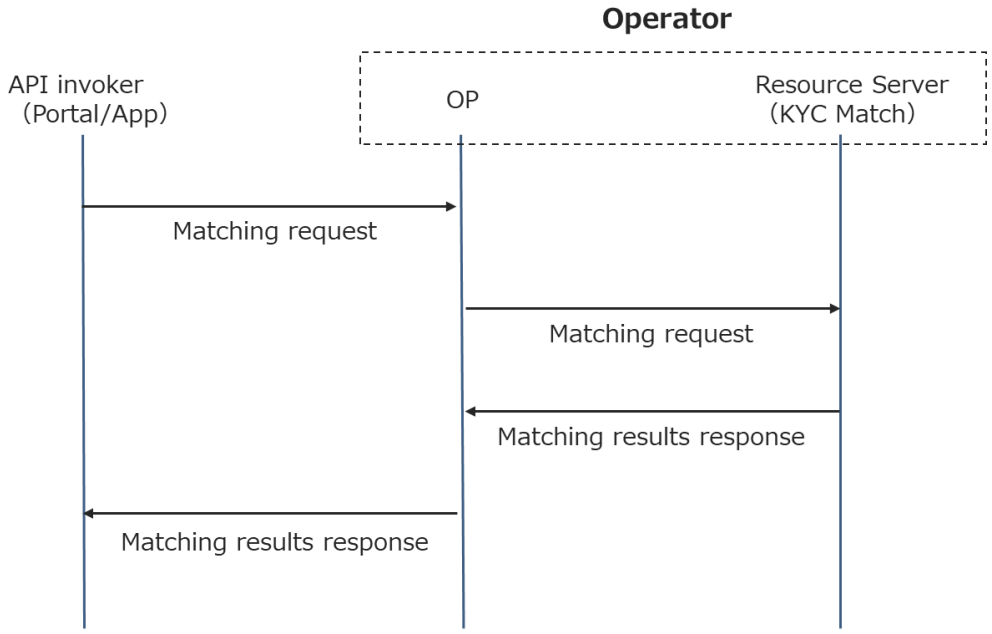

Figure 13: High level flow sequence of KYC Match API

2.9.3 Charging Aspects

The API can be classified as a "network capabilities exposure services: without impact on device's data usage" in GSMA PRD OPG. 07 [5] for charging.

2.9.4 Roaming (Home Routed scenario) Aspect

The API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.9.5 Federation Aspects

It is assumed that the API invoker has relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the operator which the Leading OP belongs to.

2.10 KYC Fill-in API

The version of CAMARA KYC Fill-in API which is considered in this document is ver. 0.3.0 [17].

2.10.1 Description

As defined in CAMARA KnowYourCustomer (KYC) sub project, KYC Fill-in API provides the API invoker with the ability to request and receive the information for a particular end-user, which on file (and verified) by the end-user's Operator in their own KYC records, in order for the API invoker to confirm the accuracy of the information and provide a specific service to the end-user.

The API is intended to be used in the following scenario, for example:

- To fill-in the end user personal data during the digital registration of an account to a 3rd party service.

Required information for using this API:

- Authentication/Authorisation/End-User Consent capturing: use of the OpenID Connect (OIDC) as security scheme.
- The request body of the Fill-in Request does not include any parameters/attributes.
- It is up to Operator to decide which information the API invoker will receive by calling the API, and also the Operator is responsible for security and privacy issues.

API functionality

The API exposes a single endpoint/operation:

- `/fill-in`: Providing information related to a 3rd party identity stored in the account data bound to the end-user's phone number. The operation returns:
    - Attributes available and agreed when making contract with API provider/Operator.

2.10.2 SBI Realization

No SBI-NR is available for the OP to serve the KYC Fill-in API. The OP shall interpret the KYC Fill-in API requests and transform them to proprietary request(s) to the resource server.

The following figure is the generic high-level flows of this API:

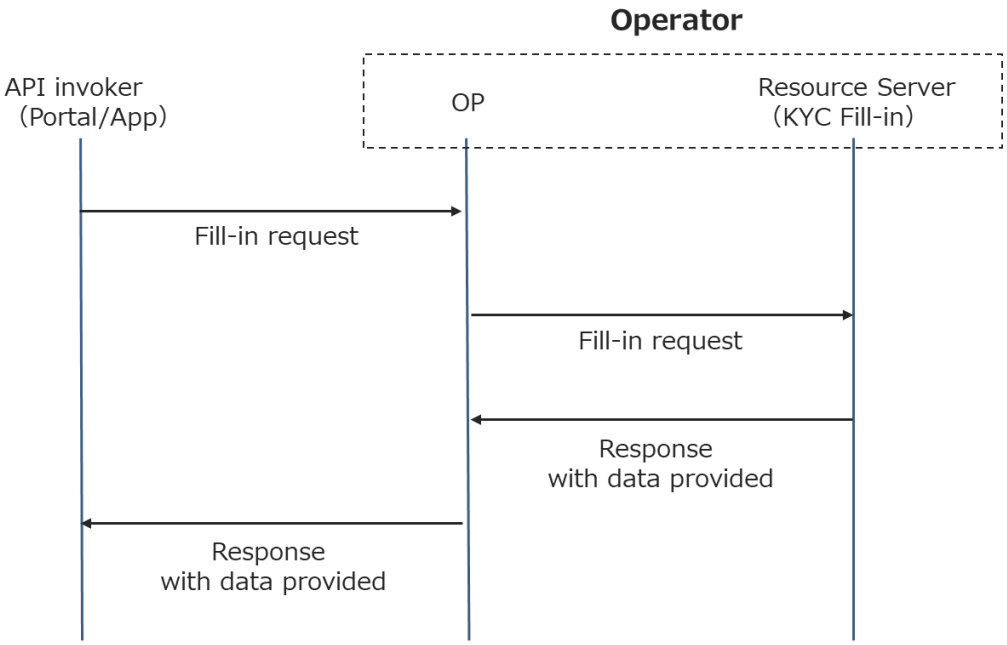

Figure 14: High level flow sequence of KYC Fill-in API

It is noted that:

- For this version of the KYC Fill-in API, the request body of the Fill-in Request does not include any parameters/attributes. An Authentication / Authorisation / End User Consent capturing processes before the Fill-in request can give the Operator the 3rd party identity (with the Access Token) and the end-user identity (with the ID Token), so the Operator can identify the 3rd party and the end-user. Then, it is up to Operator to decide which information the 3rd party will receive by calling the API, and also the Operator is responsible for security and privacy issues.
- For example, below is a potential operation:
    - when making contract with Operator, an API invoker receives its 3rd party identity and also decide which information (attributes) it will receive for its API call,
    - when calling this API, the API invoker put its 3rd party identity (only) in the request body,
    - then, Operator will provide information (attributes) which the API invoker is allowed to receive by the contract.

2.10.3 Charging Aspects

The API can be classified as a "network capabilities exposure services: without impact on device's data usage" in GSMA PRD OPG. 07 [5] for charging.

2.10.4 Roaming (Home Routed scenario) Aspect

The API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.10.5 Federation Aspects

It is assumed that the API invoker has relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1)

when the UE or subscriber to whom the request relates is not a subscriber to the operator which the Leading OP belongs to.

2.11 OTP SMS API

The version of the OTP (One Time Password) SMS API which is mentioned in this document is version 1.1.0 in CAMARA [18].

2.11.1 Description

As defined in CAMARA OTP Validation sub project [18], OTP SMS provides method to perform real time checks in order to verify possession of the device by delivering an OTP (one-time password) through SMS and validating it afterwards. This API allows application providers to verify that a provided “mobile phone number” is the one used in the device. It verifies that the end-user is using a device with the same “mobile phone number” as is associated with the SIM used in the device connected to the mobile data network.

OTP SMS is a secure method for providing one-time access to an application client or performing a single transaction.

API functionality

The API provides two endpoints/operations:

- `/send-code`: sends an SMS with the desired message and an OTP code to the received phone number
- `/validate-code`**:** verifies the received code as input is valid for the given authenticationId.

2.11.2 SBI Realization

No SBI-NR is available for the OP to serve the OTP SMS API. The OP shall interpret the OTP SMS API requests and transform them to proprietary request(s) to the dedicated endpoint of the resource server.

The following figure is the generic high-level flows of this API.

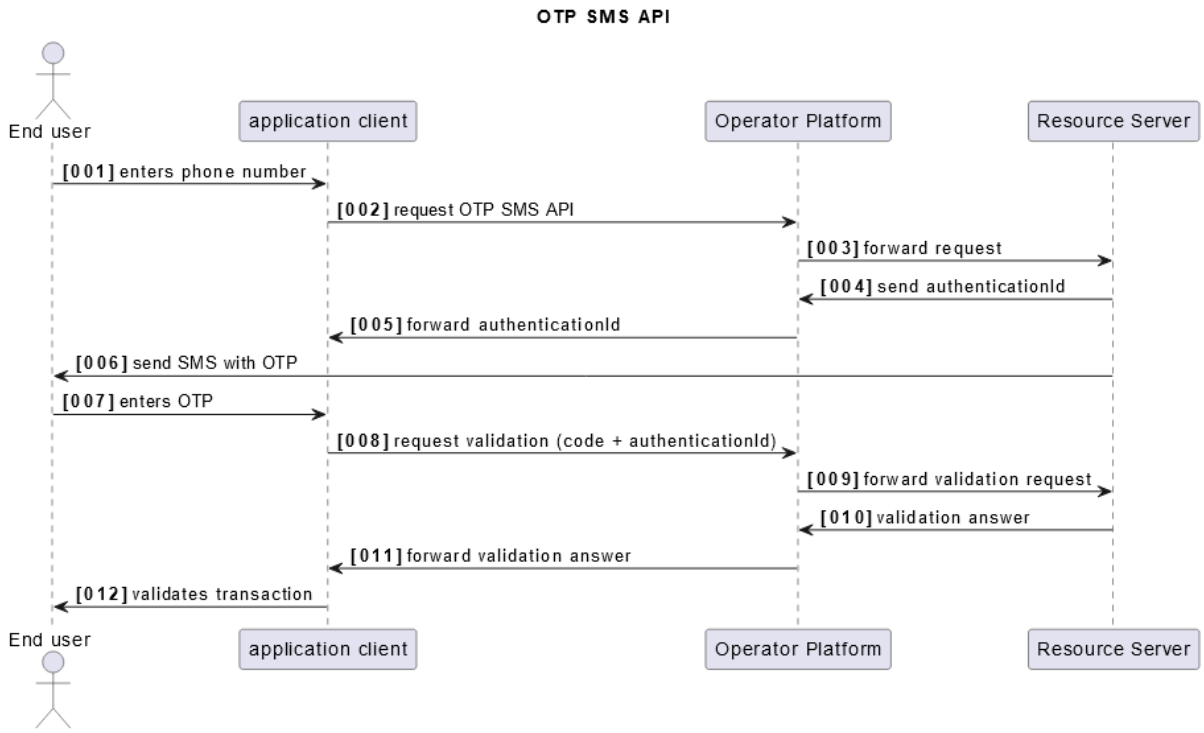

Figure 15: High level flow sequence of OTP SMS API

2.11.3 Charging Aspects

The OTP SMS API is classified as a “network capabilities exposure services: without impact on device’s data usage”. Refer to SBI-CHF API specification in GSMA PRD OPG.07 [5] for details on possible options for charging for this category.

2.11.4 Roaming (Home Routed scenario) Aspect

The API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.11.5 Federation Aspects

It is assumed that the API invoker has relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the Leading Operator.

2.11.6 Error codes

The following errors apply to OTP SMS API

Error code	Description
400	This error code is returned when the application client specified an invalid argument, request body or query param (ex: MSISDN value does not comply with the schema). For the different end-point, the error code will be returned for in the following cases: `/send-code`: Missing request body for send-code, empty object as request body for send-code, incorrect phone number in the request, missing message request

Error code	Description
	attribute, missing code in message request attribute, message attribute exceed maximum length authorized `/validate-code`: Missing request body, Empty object as request body for validate-code, missing authenticationId as request parameter, missing code as request parameter, exceed the maxLength for code, code value distinct from the code value received in the SMS, the time elapsed since the send-code exceed the allowed time, code expired because a new one has been requested for the same phone number, verification expired because authenticationId is no longer valid because it has already been used, the maximum number of attempts for this authenticationId was exceeded without providing a valid OTP
401	This error code is returned when there is an authentication problem with the client request: a request without an access token, an invalid access token, an expired access token, no Authorization header in the request
403	This error code is returned when the application does not have sufficient permission (access token does not have the required scope). This error code will also be returned by the /sent-code endpoint for the following reasons: too many OTPs have been requested for this MSISDN, the given MSISDN can't receive an SMS due to business reasons in the operator (fraud, receiving SMS is not supported,…), the given MSISDN is blocked to receive SMS due to any blocking business reason in the operator.
404	This error code is returned when the resource is not found. For the different end-point, the error code will be returned for in the following cases: `/send-code`: phone number that did not belong to the operator `/validate-code`: authenticationId is set to an unknown value
429	This error code is returned when the application client sends too many request to the server (Request is rejected due to exceeding a business quota limit, Access to the API has been temporarily blocked due to rate or spike arrest limits being reached)

Table 6: Specific error codes

2.12 Device Status API

The Device Status APIs provides the API consumer with the ability to:

- check if a device is reachable or is not connected to the network by using Device Reachability Status API
- check if a device is roaming, and in which country by using Device Roaming Status API
- query to which Mobile Communication Technology the device is connected to by using Connected Network Type API
- receive notifications if the connectivity or roaming status of the device changes by using Device Reachability Status Subscription APIs or Device Roaming Status Subscription API
- subscribe to notifications on changes to the connected network type of a device by using Connected Network Type Subscriptions

2.12.1 Device Reachability Status

The version of the Device Reachability Status API which is mentioned in this document is version v1.0.0 in CAMARA [20].

2.12.1.1 Description

As defined in [20], Device Reachability Status API provides a programmable interface for developers and other users (capabilities consumers) to verify whether a certain user device is reachable from the network via data- or sms-usage.

API functionality

The API provides one endpoint/operation:

- `/retrieve`**:** retrieves current connectivity status.

2.12.1.2 SBI Realization

The CAMARA Device Reachability Status API leverages 3GPP Monitoring Event as mentioned in section 2.1 of OPG.03 [9]. The following monitoringType can be used

- LOSS_OF_CONNECTIVITY: to be notified when the 3GPP network detects that the UE is no longer reachable for signalling or user plane communication
- UE_REACHABILITY: to be notified when the UE (User Equipment) becomes reachable for sending either SMS or downlink data to the UE. For this monitoring type, there are two reachability types:
    - SMS: The AS requests to be notified when the UE becomes reachable for sending SMS to the UE
    - DATA: The AS requests to be notified when the UE becomes reachable for sending downlink data to the UE

To get the device status concerning the reachability, a possible implementation is the following:

- First call with a request to get the status for data connectivity. If the call is successful, the device is connected to the data network. If it fails, perform a second call
- Second call with a request to get the status of the device for SMS connectivity. If the call is successful, the device is connected to the SMS network. If it fails, declare loss of connectivity

The below flow sequence in Figure 16 details this functionality from a high-level point of view.

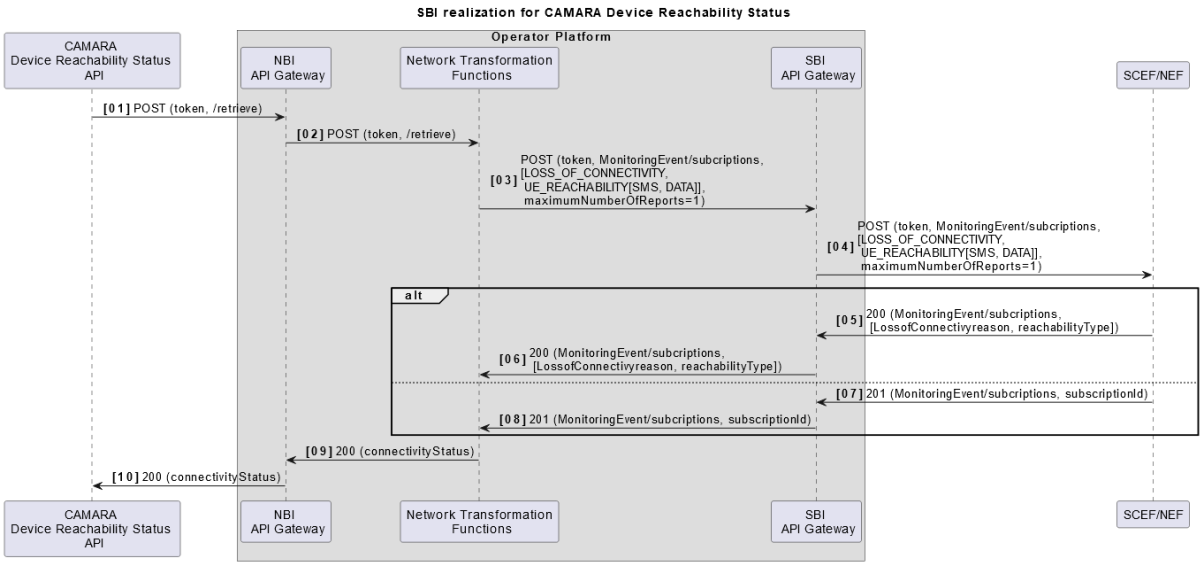

Figure 16: High level flow sequence of Device Reachability Status API

Below is an example for connectivity status for SMS:

Request

CAMARA POST /retrieve-connectivity-status	→ NTF	3GPP Monitoring Event POST /subscriptions

<pre>{
"device":{
"phoneNumber":
"123456789"
}
"webhook": {
"notificationUrl":
"https://application-server.com",
 	
"notificationAuthToken":
"c8974e592c2fa383d4a3960714"
}
}</pre>
		<pre>{
		"msisdn": "123456789",
		"monitoringType": "UE_REACHABILITY",
		"reachabilityType": "SMS",
		"maximumNumberOfReports": 1,
		"notificationDestination": "https://application-server.com/token=c8974e592c2fa383d4a3960714"
		}</pre>

Response:

3GPP Monitoring Event 200	→ NTF	CAMARA 200

<pre>{
"msisdn": "123456789"
"monitoringType": "UE_REACHABILITY",
"reachabilityType": "SMS",
"notificationDestination": "https://application-server.com/token=c8974e592c2fa383d4a3960714"
 	
}</pre>
		<pre>{
		"lastStatusTime": "2024-02-20T10:41:38.657Z",
		"connectivityStatus": [
		"SMS"
		]
		"webhook": {
		
		"notificationUrl": "https://application-server.com",
		
		"notificationAuthToken": "c8974e592c2fa383d4a3960714"
		
		}
		}</pre>

`lastStatusTime` claim is added by the NTF to the 3GPP Monitoring Event response.

Figure 17 details this functionality from a high-level point of view.

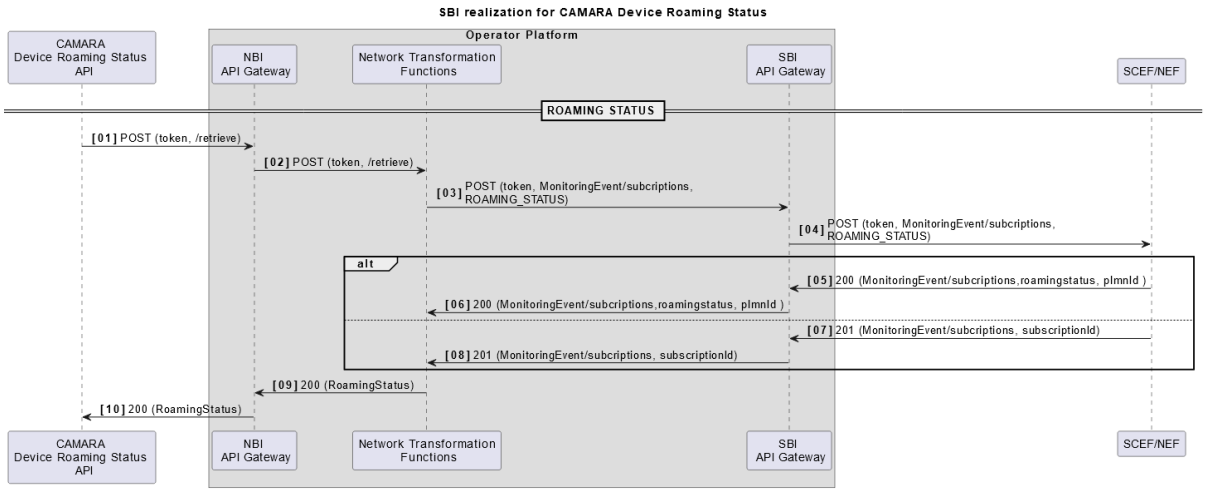

Figure 17: High level flow sequence of Device Roaming Status API

Below is an example for roaming status:

Request

CAMARA POST /retrieve-roaming-status	→ NTF	3GPP Monitoring Event POST /subscriptions

<pre>{
"device":{
"phoneNumber":
"123456789"
}
"webhook": {
"notificationUrl":
"https://application-
server.com",
 	
"notificationAuthToken":
"c8974e592c2fa383d4a396071
4"
}
}</pre>
		<pre>{
		"msisdn": "123456789",
		"monitoringType": "ROAMING_STATUS",
		"plmnIndication": true,
		"maximumNumberOfReports": 1,
		"notificationDestination": "https://
		application-
		server.com/token=c8974e592c2fa383d4a396
		0714"
		
		}</pre>

Response:

3GPP Monitoring Event 200	→ NTF	CAMARA 200

<pre>{
"msisdn": "123456789"
"monitoringType": "ROAMING_STATUS",
"plmnId":{
"MCC": 901,
"MNC": [],
}</pre>
		<pre>{
		"lastStatusTime": "2024-02-
		20T10:41:38.657Z",
		"roaming": true,
		"countryCode": 901,
		"countryName": [],
		"webhook": {
		
		"notificationUrl":
		"https://application-
		server.com",
		
		"notificationAuthToken":
		"c8974e592c2fa383d4a3960714"
		
		}
		}</pre>

`lastStatusTime` claim is added by the NTF to the 3GPP Monitoring Event response.

2.12.1.3 Error codes

The following errors apply to Device Roaming Status API

Error code	Description
400	This error code is returned when an invalid argument is used or when there is a generic syntax exception (e.g.; the device value is an empty object, some device identifier value does not comply with the schema)
401	This error code is returned when there is an authentication problem with the client request: a malfomed access token, an expired access token, no Authorization header in the request
403	This error code is returned when the application client does not have sufficient permission If the access token does not have the required scope or the user fails operational security, the response code is "PERMISSION_DENIED".
404	This error code is returned when some identifier cannot be matched to a device. The response code is "IDENTIFIER_NOT_FOUND".
422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code: "IDENTIFIER_MISMATCH": Inconsistency between device identifiers not pointing to the same device, "SERVICE_NOT_APPLICABLE": Service is not available for the provided identifier, “UNSUPPORTED_IDENTIFIERS”: None of the provided identifiers is supported by the implementation, “MISSING_IDENTIFIER”: f the subject cannot be identified from the access token and the optional device object is not included in the request, “UNNECESSARY_IDENTIFIER”: If the subject can be identified from the access token and the optional device object is also included in the request. This will be the case even if the same device is identified by these two methods, as the server is unable to make this comparison.
429	This error code is returned when the application client is out of resource quota or reaching rate limiting. The following response code will be added to the returned error code: "QUOTA_EXCEEDED": Request is rejected due to exceeding a business quota limit. "TOO_MANY_REQUESTS": API Server request limit is overpassed.

Table 7: Specific error codes

2.12.2 Device Reachability Status Subscription API

This API is in initial state in CAMARA. It will be included when it reaches stable state.

2.12.3 Device Roaming Status Subscription API

This API is in initial state in CAMARA. It will be included when it reaches stable state.

2.12.4 Connected Network Type API

This API is in initial state in CAMARA. It will be included when it reaches stable state.

2.12.5 Connected Network Type Subscription API

This API is in initial state in CAMARA. It will be included when it reaches stable state.

2.12.6 Charging Aspects

The Device Status APIs are classified as a "network capabilities exposure services: without impact on device's data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.12.7 Roaming (Home Routed scenario) Aspect

The APIs will always be addressed to the home network, as such no roaming aspects need to be considered.

2.12.8 Federation Aspects

It is assumed that the application provider has a relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the Leading Operator.

2.13 Edge Cloud Discovery APIs

2.13.1 Simple Edge Discovery API

2.13.1.1 Description

The Simple Edge Discovery (SED) API is a Service API developed in the CAMARA Edge Cloud subproject and defined according to the following YAML file [21]. This API allows a client application to discover the closest Edge Cloud Zone name to the UE hosting the client application. 'Closest' means 'shortest network path' as that will give the shortest propagation distance, which is a major factor in latency.

Network operators may host multiple Edge Cloud Zones in a given territory. Connecting the Application Client to a server on the closest Edge Cloud Zone means packets travel the shortest distance between endpoints, typically resulting in the lowest round-trip latency. Note, the physical (GPS) location of a user device is not a reliable way to determine the closest Edge Cloud Zone, due to the way operator networks are routed - so the operator will calculate the Edge Cloud Zone with the shortest network path to the network-attached device identified in the API request.

Once the SED API gives the name of the closest Edge Cloud Zone to the user device, can be used to:

- Connect the application client on the user device to the application server instance on that Edge Cloud Zone. Note: the address of that server instance is not part of the API response but should be known in advance.
- Or, if there is no instance on that Edge Cloud Zone, Application Provider may wish to deploy one there.

The API may be called either by an application client hosted on a device attached to the operator network (i.e. phone, tablet), or by a server.

There is a single API endpoint: /edge-cloud-zones?filter=closest. To call this endpoint, the API consumer must first obtain a valid OAuth2 token from the token endpoint and pass it as an Authorization header in the API request.

The API returns the closest Edge Cloud Zone to a given device, so that device needs to be identifiable by the network: Required information for using the SED API If the API call is from a server, it must explicitly pass one or more device identifiers in the HTTP request header:

- IP-Address: This is the public IP address of the user device: this can be obtained by an application hosted on that device calling a public IP address API (e.g. GET https://api.ipify.org?format=json).
- Phone-Number: The international E.164 format (starting with country code), e.g. +4407123123456.
- Network-Access-Identifier (where available from the API host operator): If the API is called from a device attached to the operator network, then Application Providers can attempt the request without passing device identifier(s) parameters in the request header. If that request fails with a 404 Not Found error, then retry the request but this time include a device identifier.

The provider of the Edge Cloud Zone may be an operator, or a cloud provider.

2.13.1.2 SBI Realization

The OP shall interpret the CAMARA Simple Edge Discovery API requests and properly redirect it to the Network Element in charge of handling the request(s) within the Operator's Network via the SouthBound Interface – Network Resources (SBI-NR) [9].

The objective of Simple Edge Discovery is to retrieve the closest Edge Cloud Zone Name for a specific UE ID, to obtain this information Operators can use Transformation Function mechanism:

To implement the SED API, the most appropriate UE identifier should be the IP address, but this will depend on each Operator's Network and the ease with which they can map the IP address to internal subscriber identifiers (e.g. MSISDN) as is described in section 2.1.

2.13.1.2.1 Transformation Function

This mechanism leverages the information from the Operator's Network. In order to meet the latency expectations of an Operator Platform, it is necessary to be able to identify the gateway through which a user is connecting to the Internet. In the Operator's domain the UPF is the element responsible for providing the internet gateway to Operator's Subscribers. This way, in general, the lowest latency will be achieved using the closest the Edge Cloud Zone to a UPF.

2.13.1.2.1.1 SMF Based Solution

Among other functionalities such as managing sessions (including their establishment, modification and termination) and overseeing IP address allocations to User Equipment (UE) and authorizing them, when necessary, the SMF, as defined in [22], is the Core Network element in charge of addressing the UE to the different UPFs in the operator domain. Therefore, the creation of a carbon copy within an SMF Backend will be pivotal in order to encompass a solution this type for Simple Edge Discovery as shown in Figure 18. The API Gateway shall query this SMF backend to retrieve the UPF that a user is connected to. In case there is no SMF backend this element can be addressed using the Radius/Diameter protocol.

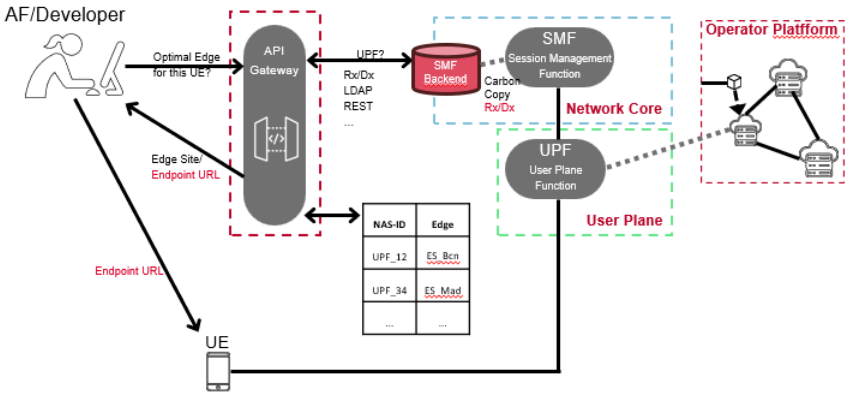

Figure 18: Architecture of the SMF Based Solution

Note: Querying Network Core in a massive way could lead to network outage. SMF Backend is highly recommended. In 4G architectures SMF is known as AAA or also as PGW-C and is addressed the same way.

Once the user is attached to the UPF that is discovered by the API logic, a mapping between this UPF and the Edge Application Server needs to be done. Therefore, a database or a dictionary should be included within the API logic to support this feature.

The way in which this element would be filled is during the deployment process of the Telco Edge Cloud Zones: Once a zone has been deployed, it would be advantageous to ascertain which UPF can demonstrate superior Key Performance Indicators (KPIs) using the new zones compared to the current ones and modify the mapping in accordance with this analysis.

Consequently, this component is of significant importance in the solution architecture and, as previously stated, could be characterized as either a database or a query file due to the limited number of zones.

2.13.1.2.1.2 NEF (Location) Based Solution

The operator network can infer which gNodeB a particular user is connected to at any time. In the 5G standard architectures the Access and Mobility Management function (AMF) is the element in charge of managing the UE mobility within the different gNodeBs of the operator (MME in 4G architectures). OP can access this information using NEF APIs, particularly 3GPP namf_location [23], getting to know which gNodeB is serving the user so to make the mapping between gNodeB and Edge Cloud Zone NEF is not the only way to get to know the gNodeB serving a user, but also GMLC [24] or even the UDM. Nevertheless, the gNodeB serving the user might not be related with the UPF serving them which may lead into not optimal latency. This is because the SMF, which is responsible for assigning the most suitable UPF for a certain connection, does not follow an intuitive criterion. The SMF typically assigns the UPF to a terminal based on load levels among the different UPFs that can serve this user (best effort) and in general terms is not even performing a mobility among UPFs. Therefore, a user may be connected to a radio cell in City A but access the internet through the UPF in City B instead of the physically closest UPF in City A. That said, to achieve a firm solution, it is needed mobility procedures in SMF regarding UPF.

From the developer/API logic point of view, this mobility may be forced using a NEF API such as Traffic Influence [see 2.8] or Traffic correlation. However, this solution may be very conditioned by the Network topology (distribution of the SMFs and UPFs) and may not be suitable for all the Operators.

Operator configuration can be done to enable User Plane re-anchoring after UE mobility as defined in [22]. Anyhow, this sort of configuration will lead to load balancing problems regarding user plane if the network is not correctly dimensioned.

Recapitulating, to enable this sort of solution for Simple Edge Discovery, Location procedures and optimal User Plane allocation would be needed which may be performed using NEF (traffic Influence) or through internal configuration. This solution may be key in Edge use cases such as Smart Edge Allocation.

2.13.1.2.1.3 NEF (Traffic Influence) Based Solution

The NEF Traffic Influence API describes the procedures between an Application Function and the SMF to maintain an efficient user plane, as is defined in 2.8. Therefore, within the API logic there is a point where it must be acknowledged the UPF serving a certain user, nevertheless this is not exposed as it is against the NEF design premises (no network topology should be exposed to an Application Function).

However, it is possible for NEF to act as a network function in conjunction with another Network Function (NF), such as 3GPP EASDF [23]. EASDF is a NF that comprises a Domain Name System (DNS) resolver, which is used to correlate Edge Cloud Zones IP addresses and DNS records, as well as to facilitate the discovery and selection of edge application servers (EAS). The use of NEF APIs such as Traffic Influence is being considered. The use of this API could offer the capabilities of Edge Discovery, although additional configurations would be required in the core.

The proposal is that the integration of the Operator Platform can be done jointly with the Core. In this way the Traffic Influence API whose SBI realization is defined in 2.8, would collaborate in obtaining step 3 of the flow in Figure 19 to expose it to the Operator Platform. In the same way that a notify can be done in the event of a UPF change, it should be possible to do a GET by what is known as a trusted entity, which in this case would be the Operator Platform which, in short, would be the AF shown in the first flow in Figure 19.

In order to facilitate the correlation between SMF-UPF-EASDF-EAS, it is necessary to implement a pre-configuration in the core as mentioned before. So, making a pre-configuration to make the correlation between SMF-UPF-EASDF-EAS is needed.

An alternative, more straightforward approach would be to define a GET Method within operation number 6 in Figure 19. This operation shows the Traffic Routing of the N6 interface (UPF to Domain Network/EAS). Anyhow, this is not defined in any standard because of the reasons stated above.

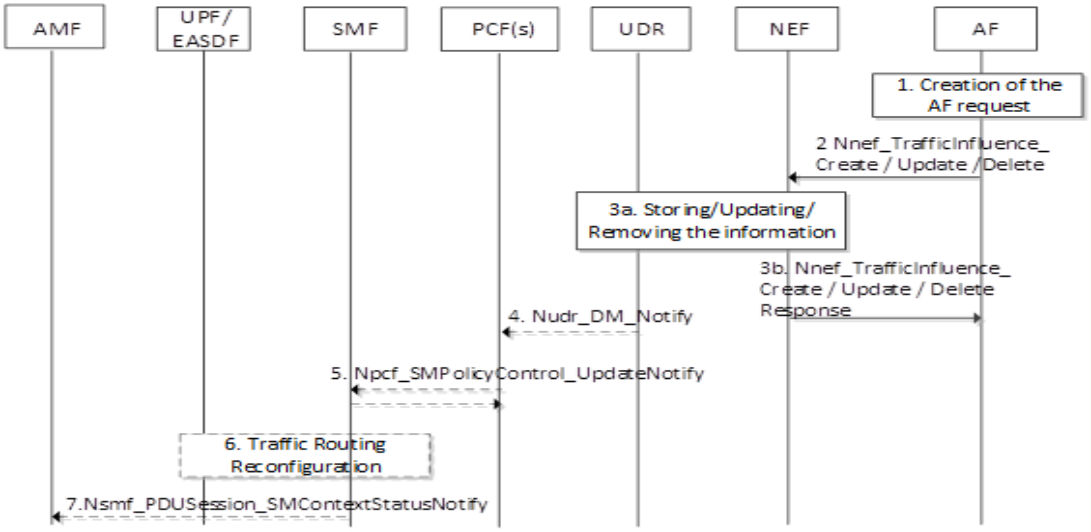

Figure 19: Application Function influence on traffic routing and service function chaining [23]

NEF-based solutions are quite complex to be a short-term reality. Vendors are not supporting most of the functionalities yet and the integration with all the elements of the Core is quite a challenge.

Using the gNodeB to determine the UPF/Edge Cloud Zone a user is/should be connecting to is quite risky since the UPF mobility procedures must be implemented within the operator architecture (SMF) so to provide the optimal latency expectations.

SMF based solution proves as the better option as the support for 4G and 5G can be ensured. Anyhow, direct SMF/AAA queries prove a high risk in terms of Network outage. Queries should be done using a backend.

2.13.1.2.2 Measurements/Probes

This mechanism consists in measuring the performance from the UE to the Edge Cloud Zones and choosing the Edge Cloud Zone Name of the best measurement. The probe in this document refers to the mechanism by which an application client on UE can send TCP/UDP/ICMP protocol packets or probes to Edge Cloud Zones and receive reflected packets and measure the different performance metrices of the underlying transport in end-to-end manner.

Probes can be used to measure some of the characteristics of the network (not an exhaustive list though) that may be helpful to decide adequate edge for an application

- Round-Trip Time (RTT): The time taken for a data packet to travel from the source device to the destination and back.
- Packet Delivery Delay: The time delay between the transmission of a packet and its successful delivery at the destination.
- Jitter: The variation in latency over time, which can affect the consistency of real-time applications.
- One-Way Delay (OWD): The time taken for a data packet to travel from the source device to the destination without considering the return trip.

The measurements performed by an application on a user device reflects the real-time state of the end-to-end connectivity insights which if available or known to the OP and that can be utilized in various decision-making processes e.g. edge selection, mobility triggers, QoS profile change, etc.

Different applications may have varying QoS requirements and OP can use the connectivity measurement info for application placements and orchestrations considering.

Role of UNI for Probe Measurements:

- Probes can be an optional capability which an App Client may use to request better quality of experience.
- An OP may discover over UNI the App Clients capabilities to perform or measure one or more connectivity parameters as described
- An OP may instruct App Clients to start and measure probes and report back the results over UNI.

Role of Network, Edge for Probe Measurements:

- Probes can be defined in multiple ways depending on the capabilities of the underlying network and operator policies that controls the network behaviour
- For example, a mobile network may not allow to use the ICMP packets so techniques used for probes should have consideration for multiple ways that can be used for such measurements.
- An OP may provide probe responder endpoints e.g. over TCP/UDP as part of the edge sites and inform App Client over UNI to measure probe results via those probe endpoints over edge data plane and inform the results back.
- An OP may be configured with network information on expected capabilities of the network to support applications probes and accordingly instruct UE via UNI to use specific methods for measurements.

Note: Using this alternative as an Edge selection mechanism is not recommended due to security concerns related to the risk of flooding, the operator's inability to fully determine the optimal Edge site when the device is interfering and technical concerns regarding device centric solutions.

2.13.1.3 Charging Aspects

The SED API is classified as "network capabilities exposure services with no impact on the device's data usage". Refer to SBI-CHF API [5] specification for details on possible options for charging for this category.

2.13.1.4 Roaming (Home Routed scenario) Aspect

Editor's Note: Roaming considerations are FFS.

2.13.1.5 Federation Aspects

Editor's Note: Federation considerations are FFS.

2.13.1.6 Error codes

The following errors apply to SED API

Error code	Description
400	This error code is returned when filter parameter used in the request is invalid. The associated response code is “INVALID_QUERYSTRING”
401	This error code is returned when there is an authentication problem with the client request: a request without an access token, an invalid access token, an expired access token, no Authorization header in the request If the request cannot be authenticated, the response code is " "UNAUTHENTICATED". If the authentication is no longer valid and a new authentication is needed, the response code is "AUTHENTICATION_REQUIRED"
403	This error code is returned when the application client does not have sufficient permission If the access token does not have the required scope or the user fails operational security, the response code is "PERMISSION_DENIED". If the phone number provided in the request body does not match the phone number asssociated with the access token, the response code is "INVALID_TOKEN_CONTEXT"
404	This error code is returned when the resource is not found. If the resource is not found, the response code is "NOT_FOUND" If the device identifier is not found, the response code is "DEVICE_NOT_FOUND".
406	The server cannot produce a response matching the content requested requested by the client through Accept headers (ex: Accept header is set to application/rtf and the server can only accept application/json)
422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code: "DEVICE_IDENTIFIERS_MISMATCH": Inconsistency between device identifiers not pointing to the same device, "DEVICE_NOT_APPLICABLE": Service is not available for the provided device, "UNIDENTIFIABLE_DEVICE": The device identifier is not included in the request and the device information cannot be derived from the 3-legged access token “UNSUPPORTED_DEVICE_IDENTIFIERS”: the device identifiers cannot be supported
429	This error code is returned when the application client is out of resource quota or reaching rate limiting. The following response code will be added to the returned error code: "QUOTA_EXCEEDED": Request is rejected due to exceeding a business quota limit. "TOO_MANY_REQUESTS": API Server request limit is overpassed.
501	Service not implemented
502	Internal routing problem in the Server side that blocks to manage the service properly

Table 8: Specific error codes

2.14 Call Forwarding Signal API

The version of the Call Forwarding Signal API which is mentioned in this document is version v0.2.0 in CAMARA [26].

2.14.1 Description

As defined in [26], the Call Forwarding Signal API provides the API Consumer with information about the status of the Call Forwarding Service on a specific phone number. The main scope of the Call Forwarding Signal API is "anti Fraud", to avoid fraudsters to use the Call Forwarding Service to carry on a scam. The Call Forwarding Service commonly used for this fraud is the Unconditional Call Forwarding. Other use cases are anyway supported by the Call Forwarding Signal API that also provides additional endpoints to detect the general Call Forwarding Service settings.

The Call Forwarding Service is provided by the Network to a phone number. This service redirects the incoming call to that phone number to an alternative destination such as another phone number or a voice mail system. There are two main types of call forwarding settings: unconditional and conditional. The Call Forwarding Signal API can be invoked to detect if the unconditional call forwarding service is active. The Call Forwarding Signal API can also be invoked to get the general status of the Call Forwarding Service (inactive, conditional, unconditional). If conditional call forwarding is active, the different types of settings are returned ('conditional_busy', 'conditional_unavailable', 'conditional_no_reply').

API functionality

The Call Forwarding Signal API is a REST API based on the CreateCallForwardingSignal resource. This resource can be filled during the API invocation, by the API Consumer, with the phone number on which the Call Forwarding Service status must be verified. Anyway, the phone number is also detected, by the API Producer, from the access token.

The main elements of the API are:

- **phoneNumber**: This is the end user phone number. The Call Forwarding Signal API verifies if a call forwarding service is active on this phone number. Note: the value of this optional parameter must match the value of the phone number retrieved from the access token.
- **CreateCallForwardingSignal:** this is the resource the API Consumer uses to define the phone number to be verified about the status of the Network Call Forwarding service.
- **UnconditionalCallForwardingSignal:** This is the resource the API Consumer gets back, containing the information about the Unconditional Call Forwarding Service status for the given phone number (PhoneNumber)
- **CallForwardingSignal:** this is the resource the API Consumer gets back, containing the information about the general status of the Network Call Forwarding service for the specified phone number.

The Call Forwarding Signal API provides two endpoints fulfilling the following intents:

- `/unconditional-call-forwardings`: Is the unconditional call fwd service active on a specific phone number?

- `/call-forwardings`: Which is the status of the call forwarding for a specific phone number?

The Call Forwarding Signal API is invoked by an API Consumer after the Consent Management flow.

The API Consumer can request the API Producer for the status of the Unconditional Call Forwarding service using the `/unconditional-call-forwardings` POST method. A boolean, with the information about the activation status of the call forwarding service, will be provided back via the UnconditionalCallForwardingSignal resource.

The API Consumer can also request the API Producer for the generic status of the Call Forwarding service using the `/call-forwardings` POST method. An array of strings with the information of the type of active call forwarding services will be provided back via the CallForwardingSignal resource.

2.14.2 SBI Realization

No standard SBI-NR is available for the OP to serve the Call Forwarding Signal API. The OP shall interpret the Call Forwarding Signal API requests and transform them to proprietary request(s) to the operator's backend systems.

2.14.3 Charging Aspects

The Call Forwarding Signal API is classified as a "network capabilities exposure services: without impact on device’s data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.14.4 Roaming (Home Routed scenario) Aspect

The API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.14.5 Federation Aspects

It is assumed that the API invoker has a relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the Leading Operator.

2.15 Connectivity Insights

Wider and faster adoption of network capabilities exposure solutions may be boosted by being able to meet Service Level Requirements (SLR) between ASPs and network operators. Those SLR may be based on well-known KPIs (e.g., latency, jitter) which are major factors influencing the quality of experience (or user perception), especially when gathered close to the application layer. Exposing connectivity insights to e.g., an application backend enables making optimal decisions in an opportunistic matter to improve the end user experience.

The Connectivity Insights API family allows a consumer (e.g., application backend) to request from the network the information useful for determining whether an application's performance requirements can be met for a specific end user’s session. Depending on the

response from the network, the application backend may request afterwards a network boost (e.g., via CAMARA Quality-on-Demand API call), and/or apply specific changes on the application side e.g., adjusting the resolution of the video stream in downlink.

2.15.1 Description

Connectivity Insights family consists of three APIs:

1. Application Profiles API

   As defined in [27], this API is used for creating an Application Profile, providing network performance “intents” / thresholds that need to be fulfilled for optimal end user application experience. The profile consists of the following KPIs: delay budget (`PacketDelayBudget`), packet error rate (`PacketErrorLossRate`), jitter (`Jitter`) and target minimum data rates in uplink and downlink (`TargetMinUpstreamRate` and `TargetMinDownstreamRate` respectively).

   Application Profiles API provides the following endpoints/operations:

   - `/application-profiles`
     - `POST` creates an Application Profile with network quality intents/thresholds
   - `/application-profiles/{applicationProfileId}`
     - `PATCH` updates an Application Profile with network quality intents/thresholds
     - `GET` retrieves an Application Profile with network quality intents/thresholds
     - `DELETE` removes an Application Profile with network quality intents/thresholds

2. Connectivity Insights API

   As defined in [28], this API provides essential visibility into network quality (a.k.a. insight) so e.g., an application backend may opportunistically decide on actions to enhance end user perception. Connectivity Insights API enables retrieving from the network the information useful for determining whether an application's quality requirements (in the shape of an Application Profile) can be met for a given end user session.

   Connectivity Insights API provides the following endpoints/operations

   - `/check-network-quality`
     - `POST` queries an insight. The response conveys (in a so-called `NetworkQualityInsight`) the network's confidence level that it can meet an Application Profile's quality thresholds for a given end user device:
       - for the 5 KPIs defined in the Application Profile (i.e., `PacketDelayBudget`, `TargetMinDownstreamRate`, `TargetMinUpstreamRate`, `PacketlossErrorRate` and `Jitter`) it provides a plain-language indicator of how confident the network is to meet a given network demand, namely `meets the application`

`requirements` or `unable to meet the application requirements`.

- additional information about `connectivityType` (`5G-SA`, `5G-NSA`, `4G`, `3G`) and `signalStrength` (`excellent`, `good`, `fair`, `poor`, `no signal`) is also retrieved.

3. Connectivity Insights Subscriptions API

   As defined in [29], Connectivity Insights Subscriptions API provides the following endpoints/operations

   - `/subscriptions`
     - `POST` creates a connectivity insights subscription for a device, application server and specific Application Profile. An event notification is triggered considering the following event types: `EventNetworkQuality` (carrying a `NetworkQualityInsight`) or `EventSubscriptionEnds.` (carrying a `TerminationReason` and `SubscriptionId`)
     - `GET` retrieves the connectivity insights subscriptions
   - `/subscriptions/{subscriptionId}`
     - `GET` retrieves a given subscription information based on the `subscriptionId`
     - `DELETE` removes the subscription identified by `subscriptionId`

2.15.2 SBI Realisations

As indicated in clause 6.14 in 3GPP TS 23.288 [30], an API consumer may retrieve user plane performance (i.e. average/maximum traffic rate, average/maximum packet delay, average packet loss rate) in the form of statistics or predictions using the NWDAF “DN Performance” Analytics ID.

Note: NWDAF-based realisation is only applicable for networks where NWDAF is available. 4G-only networks require a different approach which is FFS.

Note: Even though [30] mainly points to “DN Performance” analytics ID being used for edge applications, there are 3GPP specifications in which it is already considered/used beyond edge scenarios, see e.g., clause 6.1.1.3 in [31].

Furthermore, as stated in clause 6.4.1 in 3GPP TS 23.288 [30], an API consumer may retrieve information about service experience for an application over specific radio access technologies using the NWDAF “`Observed Service Experience`” Analytics ID.

The API consumer can either use a Request-Response model (suitable for retrieving insights from the network) or a Subscribe-Notify model (for getting notifications about network insights).

The API consumer may be an AF (e.g., an application backend or an Operator Platform) accessing the NWDAF services via NEF, as indicated in clauses 6.1.1.2 and 6.1.2.2 in 3GPP TS 23.288 [30].

2.15.2.1 Request-Response model

Figure 20 presents one (out of several) possible realisation for retrieving a network insight in which an OP acts as an AF gathering requests from an application backend on the NBI and transforming / forwarding them to the underlying network.

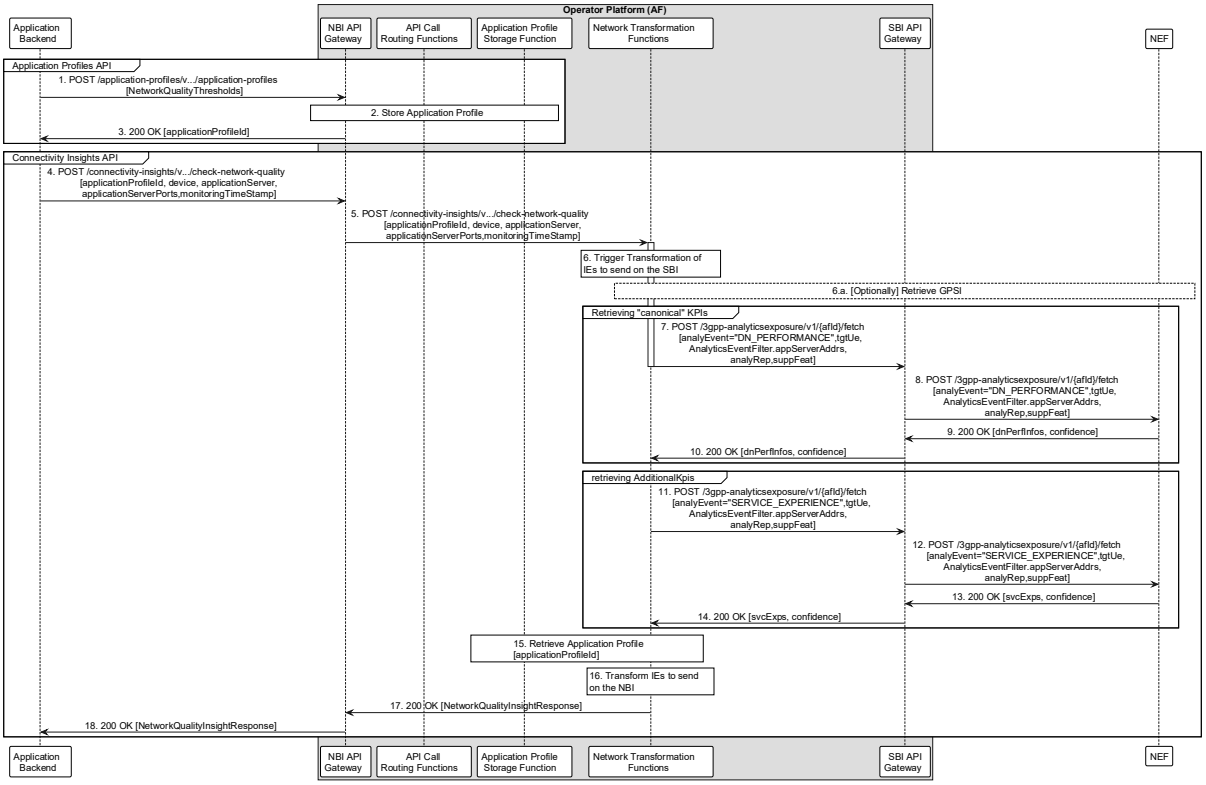

Figure 20: NBI Realisation for Application Profiles and Connectivity Insights APIs

2.15.2.2 Application Profiles API

In steps 1-3 in Figure 20, an Application Profile is created by the application backend and stored through the Operator Platform (OP).

Note: Whether the OP stores the information about the Application Profile(s) itself or rather the request is forwarded to a repository within the CSP domain is left as an implementation choice.

The POST request in step 1 conveys information about the `NetworkQualityThresholds` but no information regarding any target user. There could be scenarios (e.g., federated environments) in which, the target user is not associated with the Leading OP [32] so carrying any information for enabling the determination of the corresponding Partner OP [32] might be required (e.g., MSISDN or IP Address). Several implementation choices are available:

- The POST request does not leave the Leading OP:
    - since the intents/thresholds provided by the Application Backend are deterministic values, the confidence level primarily depends on the estimate of the associated KPIs in the network. Therefore, the intents/thresholds play no role in the calculation of any likelihood or confidence level. Nevertheless, the Leading OP is responsible for transforming the estimates into the expected plain-text indicator of

how confident the network is to meet a given network demand (i.e., `meets the application requirements` or `unable to meet the application requirements`)

- the information related to the intents/thresholds could be delivered to the corresponding Partner OP in a subsequent message after looking up the Application Profiles by using `applicationProfileId` as a key (see e.g., clause 2.15.2.4.1 in which `DnPerformanceReq` information element is used to carry information about the application thresholds)

- The POST request could be subsequently routed to the corresponding OP:
  - in this case, it is needed for the Application Backend to additionally include information related to the target user (e.g., in a custom claim), which can be used by the OP API Call Routing Functions [32]. This custom claim may be needed to be registered to IANA as Public Claims following the registration template for JSON Web Token Claims defined in clause 10.1.1 of IETF RFC 7519 [33].

Note: Further solutions for non-subscriber related API call routing may be included after coordination with other GSMA OPG groups.

In the realisation in Figure 20, the Application Profile is not routed to any other party and is kept within the OP.

2.15.2.3 Connectivity Insights API

In step 4 in Figure 20, a request for obtaining network insights for an `applicationProfileId` is received on the NBI. The request may include information about the target user (`device`) and includes information about the Application Backend (`applicationServer`).

Note: Whether or not the `device` identifier is included in the request depends on the type of access token used for the API call. If a two-legged access token is used, the `device` identifier must be provided whereas if a three-legged access token is used, it must not be included, as the target user will be uniquely identified from the access token.

If the target user is a subscriber of the operator managing the OP, the request is forwarded (step 5) to the OP Network Transformation Functions, otherwise the OP API Call Routing Functions will provide the routing information to reach the corresponding Partner OP ([32], [34]).

As part of the transformation logic in step 6, the `device` information is mapped to the `tgtUe` attribute to be sent to the NEF. The `tgtUe` is expected to be a GPSI, so if the `device` information carries:

- an IP address,
  - the procedure in clause in clause 4.15.10 in [23] may be optionally followed (step 6.a), or

- the procedure in clause 3.1 in [9] may be optionally followed (not shown in Figure 20), or

- a `phoneNumber` (GPSI in the MSISDN format) or `networkAccessIdentifier` (GPSI in the external identifier format), the mapping to `tgtUe` is straightforward and no further interaction with the operator's network is needed for filling in this attribute.

Additionally, any reporting requirements (e.g., `monitoringTimeStamp`) is mapped into `analyRep` property [35]. Finally, the information related to the `applicationServer` / `applicationServerPorts` is mapped into the attribute `appServerAddrs` of the `AnalyticsEventFilter`. The Transformation Function sets the analytics event attribute (`analyEvent`) to "`DN_PERFORMANCE`".

As indicated in clause 4.4.14.2 in 3GPP TS 29.522 [35], to fetch analytics information, an AF may send an HTTP POST request message to the NEF targeting the custom operation URI "`{apiRoot}/3gpp-analyticsexposure/v1/{afId}/fetch`". In steps 7 and 8, the OP sends the fetch request on the SBI. Further interaction with other NFs is as stated in clause 4.4.14.2 in 3GPP TS 29.522 [35] and out-of-scope of this document.

As stated in clause 6.14.3 in 3GPP TS 23.288 [30], the NWDAF outputs the DN service performance analytics which might be statistics or predictions (depending on the analytics target period provided in `analyRep` in steps 7 and 8). The DN service performance analytics include (among others) information about the Minimum / Average / Maximum / Variance for data rate, delay, and packet loss for the target UE and Application Server Instance Address provided in the request. Additionally, the DN service performance analytics may include a `confidence` value (see clause 5.1.6.2.45 in [36]), which is an integer between 0 (lowest confidence) and 100 (highest confidence) that needs to be transformed back (e.g., using the OP Network Transformation Functions) to provide the expected likelihood for the CAMARA API call. The response is sent back to the OP in the `dnPerfInfos` and `confidence` attributes of `AnalyticsData` [35] (steps 9 and 10).

Similarly, to retrieve information about `AdditionalKpis` (namely, a rough indication of the end user device radio signal conditions and the radio access technology) the same transformations and filters as before could be performed, with the difference that the Transformation Function sets now the analytics event attribute (`analyEvent`) to "`SERVICE_EXPERIENCE`", and the `allRat` and `allFreq` attributes within `RatFreqInformation` are set to true. In steps 11 and 12, the OP sends the complementary fetch request on the SBI. The service experience information and the related radio access type data are received in `svcExps` (steps 13 and 14).

Note: the calculation of the above mentioned service experience statistics/predictions considers already Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ) and Signal-to-noise and interference ratio (SINR) (see Table 6.4.2-3 in [30]) which are good indicators for the radio conditions of the target user.

In step 15 the OP retrieves / provides the Application Profile for the `applicationProfileId` signalled in step 4. Step 15 could have been triggered before, e.g., immediately after step 5.

In step 16, the OP Network Transformation Functions merge and adapt the information received on the SBI (steps 10 and 14) and the Application Profile to fit the expected format on the NBI in step 17, e.g., including the expected plain-language indicator of how confident the network is to meet a given network demand (the so-called `NetworkQualityThresholdsConfidence`), namely `meets the application requirements` or `unable to meet the application requirements`.

The reply is sent back on the NBI towards the application backend (steps 17 and 18).

2.15.2.4 Subscribe-Notify model

2.15.2.4.1 Connectivity Insights Subscriptions API

Figure 21 presents one (out of several) possible realisation for network insights subscription in which an OP acts as an AF gathering requests from an application backend on the NBI and transforming / forwarding them to the underlying network.

As indicated in clause 4.4.14.1 in 3GPP TS 29.522 [35], to subscribe/unsubscribe to retrieve analytics information via NEF, the AF may send an HTTP POST request message to the NEF to the resource "Analytics Exposure Subscriptions".

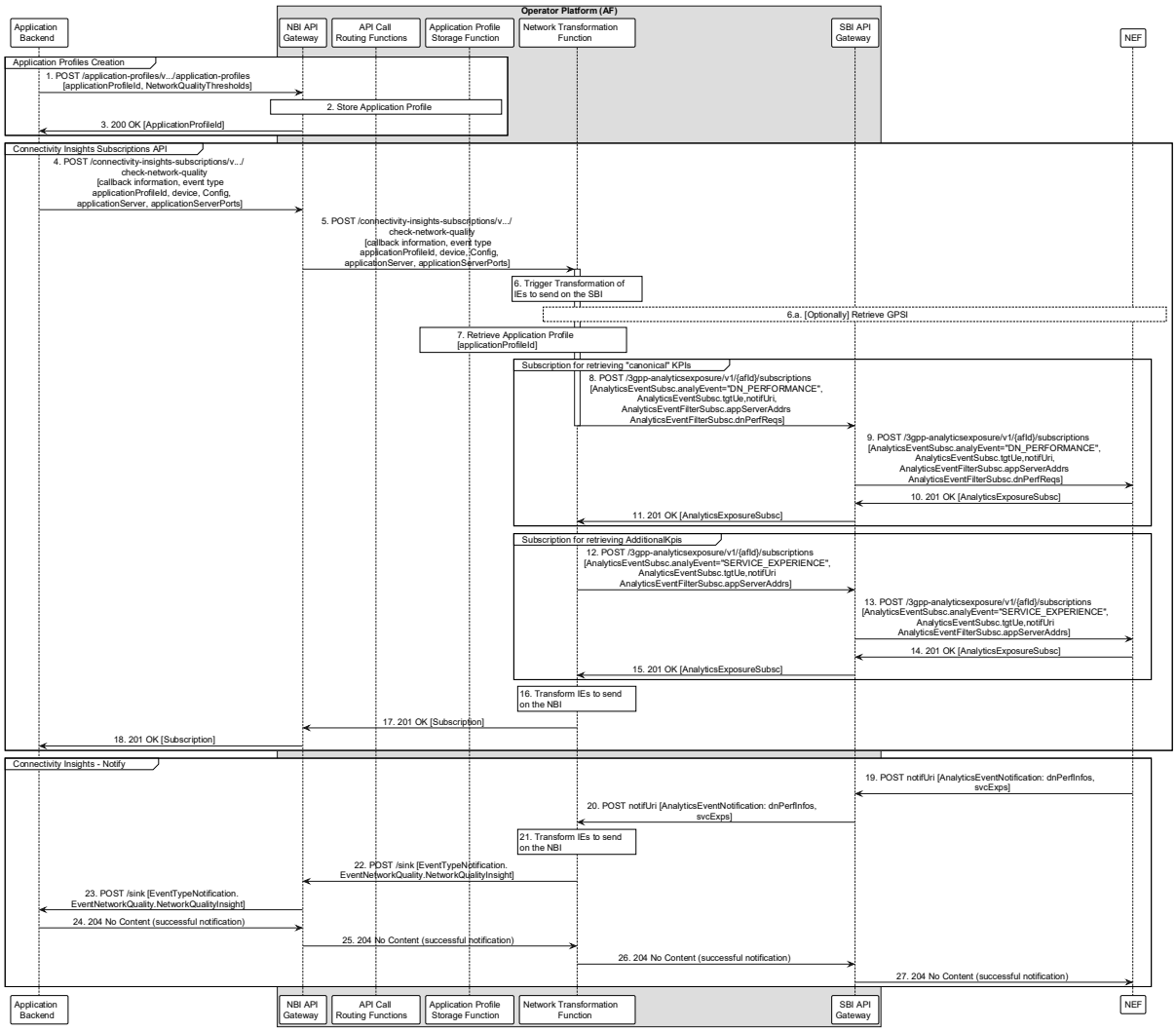

Figure 21: NBI Realisation for Connectivity Insights Subscriptions API

In steps 1-3 in Figure 21, an Application Profile is created by the Application Backend and stored in the OP (further considerations can be found in clause 2.15.2.2).

In the realisation in Figure 21, the Application Profile is not immediately routed after step 1 to any other party, but it is rather embedded in a subsequent POST request which includes target user information for routing; in this case `DnPerformanceReq` is used as a container to piggyback the Application Profile information in steps 8 and 9 on the SBI.

In step 4 the Application Backend sends a request to create a subscription providing the callback information (`sink`), `applicationProfileId`, implementation-specific configuration parameters needed by the subscription manager for acquiring events (`Config`) and the information about the event type the Application Backend is interested in (either `network-quality` or `subscription-ends`). The subscription request may include information about the target user (`device`) and includes information about the Application Backend (`applicationServer`).

Note: Whether or not the `device` identifier is included in the request depends on the type of access token used for the API call. If a two-legged access token is used, the `device` identifier must be provided whereas if a three-legged access token is used, it must not be included, as the target user will be uniquely identified from the access token.

If the target user is associated with the operator managing the OP the request is forwarded (step 5) to the OP Network Transformation Functions in the OP, otherwise the OP API Call Routing Functions will provide the routing information to reach the right operator ([32], [34]).

As part of the transformation steps starting in step 6, the `device` information is mapped to the `tgtUe` attribute to be sent to the NEF. The `tgtUe` is a GPSI, so if the `device` information carries:

- an IP address,
    - the procedure in clause in clause 4.15.10 in [23] may be optionally followed (step 6.a), or
    - the procedure in clause 3.1 in [9] may be optionally followed (not shown in Figure 21), or
- a `phoneNumber` (GPSI in the MSISDN format) or `networkAccessIdentifier` (GPSI in the external identifier format), the mapping to `tgtUe` is straight and no further interaction with the network is needed for filling this attribute.

The OP retrieves / provides the Application Profile for the `applicationProfileId` from a local repository (step 7) mapping the thresholds of the Application Profile into the `DnPerformanceReq` attribute of `AnalyticsEventFilterSubsc` [35]. Additionally, the Network Transformation Function sets the analytics event (`analyEvent`) to "`DN_PERFORMANCE`", and any relevant reporting requirements of the subscription (in `Config`) is mapped into `analyRepInfo` attribute. In Step 8, the Network Transformation Function sends a 3GPP analytics subscription request on the SBI which is forwarded to the NEF in step 9. In steps 10 and 11, a reference to the subscription

(`AnalyticsExposureSubsc` [35]) is sent to the OP including an identifier of the subscription.

Similarly, to retrieve notifications about `AdditionalKpis` the same transformations and filters as before could be performed, with the difference that the Transformation Function sets now the analytics event attribute (`analyEvent`) to "`SERVICE_EXPERIENCE`" and not performance threshold is to be piggybacked. In steps 12 and 13, the OP sends the complementary subscription request on the SBI. A reference to the subscription (`AnalyticsExposureSubsc` [35]) is sent to the OP including an identifier of the complementary subscription.

In step 16, the Network Transformation Functions adapts the response to fit the expected format of the subscription reference on the northbound, namely `Subscription` [28]. The Application Backend receives the reference to the subscription in steps 17 and 18.

As soon as one of the notification criteria is met, a notification is sent via the NEF to the OP (steps 19 and 20) carrying the subscription identifier as well as `dnPerfInfos` and `svcExps` attributes with information about the current network insights and radio conditions respectively. In step 21, the OP Network Transformation Functions convert the notification on the SBI to the expected format event type to be sent on the NBI (steps 22 and 23).

An acknowledgment without content is sent from the Application Backend to the NEF (steps 24, 25, 26 and 27).

2.15.3 Charging Aspects

The Application Profiles, Connectivity Insights and Connectivity Insights Subscriptions APIs can be classified as a "Network capabilities exposure services with no impact on device's data usage". Possible charging options for this category are provided in GSMA PRD OPG.07 [5].

2.15.4 Roaming (Home Routed scenario) Aspect

Editor's Note: Roaming considerations are FFS.

2.15.5 Federation Aspects

It is assumed that the API invoker (i.e., Application Backend in Figure 20 and Figure 21) has a relation with a single operator via the so-called Leading OP. When a target user (to whom the API requests relate) is not a subscriber of the operator managing the Leading OP, the API request could be delivered to the corresponding operator (i.e., home operator) based on an identifier of the target user (e.g., MSISDN) using the OP API Call Routing Functions ([32], [34]).This particularly holds for the Connectivity Insights and Connectivity Insights Subscriptions APIs since they carry information related to the target user. However, for Application Profiles API several implementation aspects could be considered, as already elaborated in clause 2.15.2.2. Furthermore, mechanisms for purging / cleaning registers associated with obsolete or non-relevant Application Profiles shall be in place.

2.15.6 Error codes

Table 9 presents the error codes applicable to Application Profiles API.

Error code	Description
400	This error code is returned when the application client specified an invalid argument, request body or query parameter (e.g., `packetlossErrorRate` value does not comply with the expected format). This error code applies to all the endpoints in the API definition.
401	This error code is returned when there is an authentication problem with the client request: a request without an access token, an invalid access token, an expired access token, no Authorization header in the request. This error code applies to all the endpoints in the API definition.
403	This error code is returned when the application does not have sufficient permission (e.g., access token does not have the required scope). This error code applies to all the endpoints in the API definition.
404	This error code is returned when the resource (e.g., `applicationProfileId`) is not found. It only applies to the `/application-profiles/{applicationProfileId}` endpoint.
429	This error code is returned when the application client sends too many requests to the OP exceeding a business quota limit. This is usually controlled by the NBI API Gateway. This error code applies to all the endpoints in the API definition.

Table 9: Error codes for Application Profiles API

Table 10 presents the error codes applicable to Connectivity Insights API.

Error code	Description
400	This error code is returned when the application client specified an invalid argument, request body or query parameter (e.g., `applicationProfileId` or `phoneNumber` value do not comply with the expected formats). This error code is applicable to the unique endpoint in the API definition, namely `/check-network-quality`.
401	This error code is returned when there is an authentication problem with the client request: a request without an access token, an invalid access token, an expired access token, no Authorization header in the request. This error code is applicable to the unique endpoint in the API definition, namely `/check-network-quality`.
403	This error code is returned when the application does not have sufficient permission (e.g., access token does not have the required scope). This error code is applicable to the unique endpoint in the API definition, namely `/check-network-quality`.
404	This error code is returned when a resource (e.g., `applicationProfileId`) is not found, or an identifier cannot be matched to a device. This error code is applicable to the unique endpoint in the API definition, namely `/check-network-quality`.
422	This error code is returned when using a two-legged token, no `device` identifier is provided or when using a three-legged token (with no `device` identifier provided) it is not possible to retrieve a valid identifier (yielding to a missing identifier scenario) or when using a three-legged token an additional device identifier is included (which may yield to an identifier mismatch scenario). This

Error code	Description
	error code is applicable to the unique endpoint in the API definition, namely `/check-network-quality`.
429	This error code is returned when the application client sends too many requests to the OP exceeding a business quota limit. This is usually controlled by the NBI API Gateway. This error code is applicable to the unique endpoint in the API definition, namely /check-network-quality.

Table 10: Error codes for Connectivity Insights API

Table 11 presents the error codes applicable to Connectivity Insights Subscriptions API.

Error code	Description
400	This error code is returned when the application client specified an invalid argument, request body or query parameter (e.g., `applicationProfileId` or `phoneNumber` value do not comply with the expected formats). This error code applies to all the endpoints in the API definition.
401	This error code is returned when there is an authentication problem with the client request: a request without an access token, an invalid access token, an expired access token, no Authorization header in the request. This error code applies to all the endpoints in the API definition.
403	This error code is returned when the application does not have sufficient permission (e.g., access token does not have the required scope). This error code applies to all the endpoints in the API definition.
404	This error code is returned when the resource (e.g., `subscriptionId`) is not found. It only applies to the `/subscriptions/{subscriptionId}` endpoint.
409	This error code is returned when a subscription the client is trying to create already exists or two independent processes / invocations from the same client are trying to create the same subscription. This error code only applies to the POST operation on the `/subscriptions` endpoint.
422	This error code is returned when using a two-legged token, no `device` identifier is provided or when using a three-legged token (with no `device` identifier provided) it is not possible to retrieve a valid identifier (yielding to a missing identifier scenario) or when using a three-legged token an additional device identifier is included (which may yield to an identifier mismatch scenario). This error code only applies to the POST operation on the `/subscriptions` endpoint.
429	This error code is returned when the application client sends too many requests to the OP exceeding a business quota limit. This is usually controlled by the NBI API Gateway. This error code only applies to the POST operation on the `/subscriptions` endpoint.

Table 11: Error codes for Connectivity Insights Subscriptions API

2.16 SIM Swap API

A user's MSISDN is typically associated with a SIM card (including eSIM) which is assigned with an IMSI. When the user changes the SIM but keeps the MSISDN, the IMSI is changed accordingly i.e. the MSISDN-IMSI pair changes. This is called the SIM Swap.

There may be different occurrences of SIM Swap which are described in the SIM Swap API [41]. This includes when the user activates a new SIM associated with the same MSISDN, also known as Multisim service.

2.16.1 SIM Swap

The version of the SIM Swap API which is mentioned in this document is version v2.0.0 in CAMARA [41].

2.16.1.1 Description

As defined in [41], SIM Swap API provides a programmable interface for API consumers and developers with the capability to (i) check if a SIM Swap has occurred during a past period for a given MSISDN, or (ii) to retrieve a date/timestamp of the latest SIM Swap occurrence if any.

The SIM Swap API requires the API consumer to identify the MSISDN:

- **2-legged access token**: The MSISDN shall be identified from the mandatory input `phoneNumber`.
- **3-legged access token**: The MSISDN shall be identified from the access token. The input `phoneNumber` must not be provided.

API functionality

The API provides two endpoints/operations:

- `/check`**:** checks if a SIM Swap has been performed during a past period for a given MSISDN. The '`maxAge`' defines the period in hours to be checked for SIM Swap. `/retrieve-date`**:** provides the date/timestamp of the latest SIM Swap, if any, for a given MSISDN.

2.16.1.2 SBI Realization

The OP shall interpret the CAMARA SIM Swap API requests and redirects it to SBI when possible. For the network side, the SIM Swap implementation is specific to the respective mobile network.

The SIM Swap API essentially requires only the MSISDN and the maxAge for the SIM Swap check. The SIM Swap implementation will thus need to be able to carry out the following:

- To query or track the MSISDN-IMSI pairing along with the date/timestamp from a defined source in the network
- To detect any change in the MSISDN-IMSI pairing along with the date/timestamp of the change
- If no SIM Swap has been performed and if allowed to return the SIM activation date, the SIM Swap implementation will need to be able to query the corresponding network node that tracks the SIM activation date
- To determine if the latest SIM Swap date can be provided in line with the local regulations or MNO internal privacy policy. This may determine the response to be provided.

Therefore, the SIM Swap implementation may interwork with the relevant network nodes to obtain the required information and to process the SIM Swap logics as per the SBI requests.

2.16.1.3 Error codes

The following errors apply to the SIM Swap API.

Error code	Description
400	This error code is returned when an invalid argument is used, when there is a generic syntax exception or when there is an out-of-range error: "INVALID_ARGUMENT": the phone number or max age is invalid "OUT_OF_RANGE": the max age is outside the allowed limit
401	This error code is returned when there is an authentication problem with the client request; missing, invalid or expired credentials, or when new authentication is required: "UNAUTHENTICATED": no authorization header or invalid access_token, "UNAUTHENTICATED" or "AUTHENTICATION_REQUIRED": expired access_token
403	This error code is returned when the application client does not have sufficient permission. If the access token does not have the required scope or the user fails operational security, the response code is "PERMISSION_DENIED".
404	This error code is returned when some resource cannot be matched. The response code is " NOT_FOUND".
422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code: "SERVICE_NOT_APPLICABLE": Service is not available for the provided identifier, “MISSING_IDENTIFIER”: If the MSISDN cannot be identified from the access token and the optional `phoneNumber` is not included in the request “UNNECESSARY_IDENTIFIER”: If the MSISDN can be identified from the access token and the optional `phoneNumber` is also included in the request. This will be the case even if the same device is identified by these two methods, as the server is unable to make this comparison.
429	This error code is returned when the application client is out of resource quota or reaching rate limiting. The following response code will be added to the returned error code: "QUOTA_EXCEEDED": Request is rejected due to exceeding a business quota limit. "TOO_MANY_REQUESTS": API Server request limit is overpassed.

Table 12: Specific error codes

2.16.2 SIM Swap Subscriptions

The version of the SIM Swap Subscriptions API which is mentioned in this document is version v0.2.0 in CAMARA [42].

2.16.2.1 Description

As defined in [42], SIM Swap Subscriptions API provides a programmable interface for API consumers and developers with the capability to manage notification subscription on SIM Swap event.

The SIM Swap Subscriptions API requires the API consumer to identify the MSISDN:

- **2-legged access token**: The MSISDN shall be identified from the mandatory input `phoneNumber`.
- **3-legged access token**: The MSISDN shall be identified from the access token. The input `phoneNumber` must not be provided.

API functionality

The API provides two endpoints:

- `/subscriptions`**:** There are two operations defined (i) `GET` – to retrieve the list of SIM Swap event subscriptions and (ii) `POST` – to create new event subscriptions.
- `/subscriptions/{subscriptionId}`**:** There are two operations defined (i) `GET` – to retrieve a particular event subscription by id and (ii) `DELETE` – to delete an event subscription also using the id.

Notifications are sent when the sim swap events occur and when the subscription ends. For the latter, the subscription end notification is used when:

- the subscription expiration time is reached
- the maximum number of subscription events is reached
- the API server stops sending notification prematurely
- the Access Token `sinkCredential` expiration time is reached

2.16.2.2 SBI Realization

The OP shall interpret the CAMARA SIM Swap Subscriptions API requests and redirects it to SBI when possible. For the network side, the SIM Swap Subscriptions implementation is specific to the respective mobile network.

One of the possible realizations for the SIM Swap Subscriptions is via an extension of the SIM Swap implementation with the addition of managing the subscriptions logics and mechanisms. The SIM Swap implementation may handle the SIM Swap information and events in real time hence the subscriptions manager may include triggers for such events to carry out the ensuing logics such as the notification callbacks. The subscriptions manager may handle the related requests to create new event subscriptions, to assign subscription IDs, to retrieve the list of subscriptions, and to delete specified subscriptions.

2.16.2.3 Error codes

The following errors apply to the SIM Swap Subscriptions API.

Error code	Description
400	This error code is returned when an invalid argument is used, when there is a generic syntax exception, when there is an out-of-range error, when an invalid

Error code	Description
	protocol is used for events subscription management, invalid sink credential type or token type: "INVALID_PROTOCOL": subscription creation with invalid protocol, only HTTP is supported "INVALID_CREDENTIAL": subscription creation with invalid credential, only access_token is supported "INVALID_TOKEN": subscription creation with invalid token, only bearer token is supported "INVALID_ARGUMENT": subscription creation with invalid event type, invalid expire time, missing required properties, invalid sink "OUT_OF_RANGE": specific Syntax Exception used when a given field has a pre-defined range or an invalid filter criteria combination is requested.
401	This error code is returned when there is an authentication problem with the client request; missing, invalid or expired credentials, or when new authentication is required: "UNAUTHENTICATED": no authorization header or invalid access_token, "UNAUTHENTICATED" or "AUTHENTICATION_REQUIRED": expired access_token
403	This error code is returned when the application client does not have sufficient permission. If the access token does not have the required scope or the user fails operational security, the response code is "PERMISSION_DENIED". The response code “SUBSCRIPTION_MISMATCH” is returned when there is inconsistent access token for requested subscription.
404	This error code is returned when some resource cannot be matched: “NOT_FOUND”: request to retrieve or delete a non-existing subscription.
409	This error code is returned when there is a conflict in the request. The response code “ABORTED” is returned when there is a concurrency conflict of processes of the same nature/scope. The response code “ALREADY_EXISTS” is returned when a client tried to create resource that already exists.
422	This error code is returned when there is an unprocessable entity in the request. The following response code will be added to the returned error code: "SERVICE_NOT_APPLICABLE": Service is not available for the provided identifier, “MISSING_IDENTIFIER”: If the MSISDN cannot be identified from the access token and the optional `phoneNumber` is not included in the request “UNNECESSARY_IDENTIFIER”: If the MSISDN can be identified from the access token and the optional `phoneNumber` is also included in the request. This will be the case even if the same device is identified by these two methods, as the server is unable to make this comparison.
429	This error code is returned when the application client is out of resource quota or reaching rate limiting. The following response code will be added to the returned error code: "QUOTA_EXCEEDED": Request is rejected due to exceeding a business quota limit.

Error code	Description
	"TOO_MANY_REQUESTS": API Server request limit is overpassed.

Table 13: Specific error codes

2.16.3 Charging Aspects

The SIM Swap API is classified as a "network capabilities exposure services: without impact on device's data usage". Possible options for charging for this category are detailed in GSMA PRD OPG.07 [5].

2.16.4 Roaming (Home Routed scenario) Aspect

The SIM Swap API will always be addressed to the home network, as such no roaming aspects need to be considered.

2.16.5 Federation Aspects

It is assumed that the application provider has a relation with a single operator. The request could be provided to the home operator based on the UE or subscriber identification (see section 2.1) when the UE or subscriber to whom the request relates is not a subscriber to the Leading Operator.

Annex A Document Management

A.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
1.0	16 Feb 2024	New PRD (OPG.09)	ISAG	H. Mariotte/Orange
2.0	20 Sep 2024	Update implementing OPG.09 CR1002	ISAG	H. Mariotte/Orange
3.0	28 Feb 2025	Update implementing OPG.09 CR1003	ISAG	H. Mariotte/Orange
4.0	28 July 2025	Update implementing OPG.09 CR1004	ISAG	H. Mariotte/Orange

A.2 Other Information

Type	Description
Document Owner	Operator Platform Group
Editor / Company	H. Mariotte/Orange

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com

Your comments or suggestions & questions are always welcome.