|  |  |
| --- | --- |
| 3GPP TR 23.700-95 V18.1.0 (2023-03) | |
| Technical Report | |
| 3rd Generation Partnership Project;  Technical Specification Group Services and System Aspects;  Study on application enablement aspects for subscriber-aware northbound API access;  (Release 18) | |
|  | |
|  |  |
|  | |
| The present document has been developed within the 3rd Generation Partnership Project (3GPP TM) and may be further elaborated for the purposes of 3GPP. The present document has not been subject to any approval process by the 3GPPOrganizational Partners and shall not be implemented. This Specification is provided for future development work within 3GPPonly. The Organizational Partners accept no liability for any use of this Specification. Specifications and Reports for implementation of the 3GPP TM system should be obtained via the 3GPP Organizational Partners' Publications Offices. | |

|  |
| --- |
|  |
| ***3GPP***  Postal address  3GPP support office address  650 Route des Lucioles - Sophia Antipolis  Valbonne - FRANCE  Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16  Internet  http://www.3gpp.org |
| ***Copyright Notification***  No part may be reproduced except as authorized by written permission. The copyright and the foregoing restriction extend to reproduction in all media.  © 2023, 3GPP Organizational Partners (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC).  All rights reserved.  UMTS™ is a Trade Mark of ETSI registered for the benefit of its members  3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners LTE™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners  GSM® and the GSM logo are registered and owned by the GSM Association |

Contents

Foreword 5

1 Scope 7

2 References 7

3 Definitions of terms, symbols and abbreviations 7

3.1 Terms 7

3.2 Symbols 8

3.3 Abbreviations 8

4 Key issues 8

4.1 Key Issue #1: UE-originated API invocation 8

4.1.1 Description 8

4.2 Key Issue #2: AF-originated API invocation 9

4.2.1 Description 9

4.3 Key Issue #3: Providing and revoking resource owner consent upon invoking APIs 9

4.3.1 Description 9

4.4 Key Issue #4: Discovery of target API information 10

4.4.1 Description 10

5 Architectural requirements 10

5.1 General requirements 10

6 Solutions 10

6.0 Mapping of Solutions to Key Issues 10

6.1 Solution #1: Business relationship in SNA 11

6.1.1 Solution description 11

6.1.2 Solution evaluation 12

6.2 Solution #2: Functional model description for SNA 12

6.2.1 Solution description 12

6.2.1.1 General 12

6.2.1.2 Functional model 12

6.2.1.2.1 Functional model description for the CAPIF with SNA enhancements 12

6.2.1.2.2 Functional model description for the CAPIF with SNA enhancements to support 3rd party API provider 13

6.2.1.3 Deployment models 13

6.2.1.3.1 General 13

6.2.1.3.2 Deployment of the authorization function within the CAPIF core function 13

6.2.1.3.3 Collocated deployment of authorization function and CAPIF core function 13

6.2.1.3.4 Deployment of the enhanced CAPIF by different organizations within the PLMN trust domain 14

6.2.2 Solution evaluation 15

6.3 Solution #3: Obtaining resource owner consent upon service API invocation 15

6.3.1 Solution description 15

6.3.1.1 General 15

6.3.1.2 Resource owner registration 15

6.3.1.3 Obtaining resource owner consent 16

6.3.1.4 Updating resource owner consent 17

6.3.2 Solution evaluation 17

6.4 Solution #4: API invoker obtaining resource owner consent 17

6.4.1 Solution description 17

6.4.1.1 API invoker obtaining resource owner consent prior to the service API invocation 18

6.4.2 Solution evaluation 18

6.5 Solution #5: UE-originated API invocation within CAPIF 18

6.5.1 Solution description 18

6.5.2 Solution evaluation 20

6.6 Solution #6: Discover a proper AEF with owner information 20

6.6.1 Solution description 20

6.6.2 Solution evaluation 22

6.7 Solution #7: Reducing resource owner consent inquiry in a nested API invocation 22

6.7.1 Solution description 22

6.7.1.1 Reducing resource owner consent inquiry in a nested API invocation 23

6.7.2 Solution evaluation 24

7 Overall evaluation 24

7.1 General 24

7.2 Architecture evaluation 24

7.3 Key issue and solution evaluation 24

7.3.1 Introduction 24

7.3.2 Overall evaluation of solutions for Key Issue #1 25

7.3.3 Overall evaluation of solutions for Key Issue #2 25

7.3.4 Overall evaluation of solutions for Key Issue #3 25

7.3.5 Overall evaluation of solutions for Key Issue #4 26

8 Conclusions 26

Annex A (informative): Use case examples 27

A.1 AF-originated API invocation (Gaming) 27

A.1.1 General 27

A.1.2 Pre-conditions 27

A.1.3 Service flows 27

A.1.4 Post-conditions 27

A.2 UE-originated API invocation (Location tracking) 27

A.2.1 General 27

A.2.2 Pre-conditions 27

A.2.3 Service flows 28

A.2.4 Post-conditions 28

Annex B (informative): Change history 29

# Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

In the present document, modal verbs have the following meanings:

**shall** indicates a mandatory requirement to do something

**shall not** indicates an interdiction (prohibition) to do something

The constructions "shall" and "shall not" are confined to the context of normative provisions, and do not appear in Technical Reports.

The constructions "must" and "must not" are not used as substitutes for "shall" and "shall not". Their use is avoided insofar as possible, and they are not used in a normative context except in a direct citation from an external, referenced, non-3GPP document, or so as to maintain continuity of style when extending or modifying the provisions of such a referenced document.

**should** indicates a recommendation to do something

**should not** indicates a recommendation not to do something

**may** indicates permission to do something

**need not** indicates permission not to do something

The construction "may not" is ambiguous and is not used in normative elements. The unambiguous constructions "might not" or "shall not" are used instead, depending upon the meaning intended.

**can** indicates that something is possible

**cannot** indicates that something is impossible

The constructions "can" and "cannot" are not substitutes for "may" and "need not".

**will** indicates that something is certain or expected to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**will not** indicates that something is certain or expected not to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document

**might** indicates a likelihood that something will happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

**might not** indicates a likelihood that something will not happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

In addition:

**is** (or any other verb in the indicative mood) indicates a statement of fact

**is not** (or any other negative verb in the indicative mood) indicates a statement of fact

The constructions "is" and "is not" do not indicate requirements.

# 1 Scope

The present document studies the potential enhancements in CAPIF and application enablement frameworks (e.g. SEAL, EDGEAPP, vertical enabler layers) to support the subscriber-aware northbound API access (SNA), whose requirements are specified in TS 22.261 [2].

This study takes into consideration the existing works for CAPIF (3GPP TS 23.222 [3]), SEAL (3GPP TS 23.434 [4]), and EDGEAPP (3GPP TS 23.558 [5]).

Any potential enhancements to security procedures related to CAPIF and application enablement frameworks (e.g. EDGEAPP, SEAL, vertical enabler layers) to support SNA are not in the scope of this study.

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 22.261: "Service requirements for the 5G system; Stage 1"

[3] 3GPP TS 23.222: "Common API Framework for 3GPP Northbound APIs"

[4] 3GPP TS 23.434: "Service Enabler Architecture Layer for Verticals (SEAL); Functional architecture and information flows"

[5] 3GPP TS 23.558: "Architecture for enabling Edge Applications"

[6] 3GPP TS 29.513: "Policy and Charging Control signalling flows and QoS parameter mapping"

[7] 3GPP TS 23.682: "Architecture enhancements to facilitate communications with packet data networks and applications"

[8] 3GPP TS 33.501: "Security architecture and procedures for 5G system"

[9] IETF RFC 6749: "The OAuth 2.0 Authorization Framework"

[10] IETF RFC 8693: "OAuth 2.0 Token Exchange"

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

**nested API invocation:** A series of API invocations in which the first API invocation towards the API exposing function 1 triggers this API exposing function to request another API invocation towards the API exposing function 2, which is in the same API provider domain that the API exposing function 1.

**resource owner:** An entity capable of granting access to a protected resource.

## 3.2 Symbols

For the purposes of the present document, the following symbols apply:

<symbol> <Explanation>

## 3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

AEF API Exposing Function

AF Application Function

CAPIF Common API Framework

CCF CAPIF Core Function

SNA Subscriber-aware northbound API access

# 4 Key issues

## 4.1 Key Issue #1: UE-originated API invocation

### 4.1.1 Description

The UE-originated API invocation scenario corresponds to the following requirement specified in TS 22.261 [2]:

- The 5G system shall be able to provide a UE with secure access to APIs (e.g. triggered by an application that is not visible to the 5G system), by authenticating and authorizing the UE.

In this scenario, the application on the UE invokes the northbound APIs. The scenario is illustrated in figure 4.1.1-1.

From CAPIF point of view, the application on the UE in this scenario plays the role of the API invoker, as defined in TS 23.222 [3].



Figure 4.1.1-1: UE-originated API invocation

Open issues:

1) Whether the existing CAPIF functional model is sufficient for the UE-originated API invocation.

a) If the existing CAPIF functional model is insufficient, how it must be modified to accommodate for the API invoker in the UE.

## 4.2 Key Issue #2: AF-originated API invocation

### 4.2.1 Description

In the AF-originated API invocation scenario, the AF invokes the northbound APIs, and the application on the UE consumes the service from the AF. The scenario is illustrated in figure 4.2.1-1.



Figure 4.2.1-1: AF-originated API invocation

Open issues:

1) Whether the existing CAPIF functional model is sufficient for the AF-originated API invocation.

a) If the existing CAPIF functional model is insufficient, how it should be updated.

2) Study the business relationship between the user (UE), the AF and the northbound API provider.

## 4.3 Key Issue #3: Providing and revoking resource owner consent upon invoking APIs

### 4.3.1 Description

Invocation of some northbound APIs may require obtaining resource owner consent (e.g. API to retrieve the location of the UE).

In the SNA context, it is for further study how to obtain resource owner consent when the API invocation is related to the resource owner (e.g. the user).

In addition, the system should ensure that there are mechanisms for the resource owner to revoke the resource owner consent provided previously for the API invocation.

In an API provider domain, several AEFs can expose resources of the resource owner and it is possible that the AEFs themselves may interact with each other to utilize the resources. For a nested API invocation, the API invoker performs an API invocation towards the AEF1 which also results in AEF1 to perform another API invocation to AEF2. This nested API interactions may require multiple queries to obtain resource owner consent information from resource owner to support API invocations towards the AEF1 and AEF2. It is beneficial for the resource owner to reduce the resource owner consent inquiry so that the resource owner does not have to provide the consent twice for the nested API invocations. It is for further study how to reduce such resource owner consent inquiries for nested API invocations.

Open issues:

1) Whether and how CAPIF functions can determine the resource owner upon API invocation.

2) Whether and how CAPIF can support obtaining resource owner consent from resource owner.

3) Whether and how CAPIF can support revoking resource owner consent by the resource owner.

4) Whether and how CAPIF can reduce resource owner consent inquiries for a nested API invocation.

NOTE: This document studies the high-level architecture to solve this issue. The detailed procedure for providing the resource owner consent will be specified in SA3.

## 4.4 Key Issue #4: Discovery of target API information

### 4.4.1 Description

As per the below requirement from 3GPP TS 22.261 [2] (clause 6.10.2), the third party needs to be provided with information to locate the target APIs in the network.

- provide a third-party with information to identify networks and APIs on those networks.

An AF may serve UEs that belong to different PLMNs. When the AF is not aware of the PLMN provider of the UEs and has to discover the service API in relation to such UEs, then it is not clear how to discover the service APIs. How the AF can determine the right AEF instance, which is exposing the required Northbound API for which the AF wants to consume that Northbound API with the UE(s) information.

Open Issue:

- Whether and how to support discovery of the target Service API information in CAPIF with respect to UE(s) information, especially when:

- AF is serving UEs belonging to multiple PLMNs; and

- There are multiple instances of the AEF within a PLMN.

# 5 Architectural requirements

## 5.1 General requirements

[AR-5.1-a] The confidentiality of the UE's external identity (e.g. MSISDN) shall be preserved during the API invocation supported by CAPIF.

[AR-5.1-b] The CAPIF shall support the authentication of the resource owner.

[AR-5.1-c] The CAPIF shall enable the resource owner(s) to provide and revoke resource owner consent.

# 6 Solutions

## 6.0 Mapping of Solutions to Key Issues

Table 6.0-1: Mapping of Solutions to Key Issues

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Key issue 1 | Key issue 2 | Key issue 3 | Key issue 4 |
| Solution 1 | X | X |  |  |
| Solution 2 | X | X | X |  |
| Solution 3 | X | X | X |  |
| Solution 4 | X | X | X |  |
| Solution 5 | X |  |  |  |
| Solution 6 |  |  |  | X |
| Solution 7 |  |  | X |  |

## 6.1 Solution #1: Business relationship in SNA

### 6.1.1 Solution description

This solution addresses the key issue #2 with regard to the business relationship between the user (UE), the AF and the northbound API provider in the AF-originated API invocation scenario. Considering the business relationship, the resource owner (which is a UE-side entity) is a new entity that has not been in the existing CAPIF business relationship, thus the business relationship should be updated to include the resource owner.

Figure 6.1.1-1 shows the typical business relationship in SNA. This business relationship can be applied to both AF- and UE-originated API invocation scenario, as the API invoker in figure 6.1.1-1 can either be an application on the UE or the AF.



Figure 6.1.1-1: Business relationships in SNA

The API invoker has service agreement with a CAPIF provider, and the API provider provides APIs associated with the resource owner. The CAPIF provider and the API provider can be part of the same organization (e.g. PLMN operator), as described in 3GPP TS 23.222 [3] clause 5. When the CAPIF provider is a PLMN operator, the resource owner may be a subscriber of the PLMN.

NOTE: In the current release, both the CAPIF provider and the API provider should belong to the same organization (e.g., PLMN operator).

### 6.1.2 Solution evaluation

This solution enhances the existing CAPIF business relationship by introducing the resource owner, which is viable.

## 6.2 Solution #2: Functional model description for SNA

### 6.2.1 Solution description

#### 6.2.1.1 General

This solution addresses the key issues #1, #2 and #3 with regard to the functional model to support the UE- and AF-originated API invocation and to support obtaining and revoking resource owner consent from the resource owner client. As the resource owner client is a new entity for CAPIF, the functional model for the CAPIF should be updated including the resource owner client and authorization function.

#### 6.2.1.2 Functional model

##### 6.2.1.2.1 Functional model description for the CAPIF with SNA enhancements

Figure 6.2.1.2.1-1 shows the reference point based functional model for the CAPIF with enhancements to support SNA.



Figure 6.2.1.2.1-1: Functional model for the CAPIF authorization with SNA enhancements

The resource owner client(s) are application clients used by end-users or subscribers of the API provider domain's service provider.

The resource owner client(s) interacts with the authorization function via CAPIF-8. The resource owner communicates with the authorization function to provide and revoke resource owner consent. The resource owner interactions are supported via a resource owner client, which is a client-side entity. Triggering the resource owner client to provide authorization is not supported via CAPIF‑8.

NOTE 1: CAPIF-8 details will be addressed by SA3.

The API exposing function (e.g NEF) acts as a resource owner consent enforcement point as specified in 3GPP TS 33.501 [8] and interacts with the authorization function via CAPIF-9. The API exposing function can retrieve the resource owner consent parameters from.the authorization function.

The API invoker interacts with authorization function via CAPIF-10/CAPIF-10e.

NOTE 2: Security aspects including specification of the authorisation procedures are to be decided in SA3.

##### 6.2.1.2.2 Functional model description for the CAPIF with SNA enhancements to support 3rd party API provider

In the current release, 3rd party API providers (i.e., API providers outside the PLMN trust domain) are not supported.

#### 6.2.1.3 Deployment models

##### 6.2.1.3.1 General

This clause provides the deployment models for the functional model specified in clause 6.2.1.2.

##### 6.2.1.3.2 Deployment of the authorization function within the CAPIF core function

Figure 6.2.1.3.2-1 shows a deployment model for PLMN providing the API provider. In this deployment model, the authorization function resides in the CAPIF core function. Those functions are in a single node depicted by a red box.



Figure 6.2.1.3.2-1: Deployment of the authorization function within the CAPIF core function

##### 6.2.1.3.3 Collocated deployment of authorization function and CAPIF core function

The CAPIF provider and API provider can be the same organization (e.g. PLMN operator). In such cases, the authorization function of the API provider domain and the CCF is collocated in the same node to provide authorization services (e.g. authorization grant and access token).

Figure 6.2.1.3.2-1 illustrates the deployment where the authorization function and the CAPIF core function are collocated in a single node depicted by a red box. The deployment of other CAPIF functionalities are not shown in this illustration.



Figure 6.2.1.3.3-1: Collocated deployment of Authorization function and CAPIF core function in a single node

A single node (denoted by red box) provides the CAPIF APIs and the Authorization APIs towards the API invokers in this deployment case. API invokers are served by the node for the authentication and authorization (for both API invoker related authorization and resource owner related authorization) requirements.

##### 6.2.1.3.4 Deployment of the enhanced CAPIF by different organizations within the PLMN trust domain

The CAPIF provider and API provider can be different organizations (e.g. PLMN operator can be CAPIF provider and an MVNO can be the API provider) within the PLMN trust domain.

Figure 6.2.1.3.4-1 illustrates the deployment where the CAPIF entities are deployed by different organizations. Nodes (denoted by red boxes) identify one example of deployment.



Figure 6.2.1.3.4-1: Deployment of the enhanced CAPIF by different organizations within the PLMN trust domain

The node providing CAPIF core function is deployed by the CAPIF provider (e.g. PLMN operator). The functions of the API provider domain functions can be deployed in multiple nodes. Example: The AEF, APF and AMF can reside in one node and the Authorization function can reside in another node. It is also possible that all functions of the API provider domain can reside in one node.

Typically this type of deployment is where the authentication and authorization for subscribers/users of API provider (i.e. MVNO) are managed by the API provider itself.

If API invokers wish to utilize the API provider users' resources via the service API exposed by the API provider, then API invoker is required to obtain the resource owner authorization from API provider's authorization function and obtain the API invoker's authorization from the CAPIF provider's CCF.

### 6.2.2 Solution evaluation

This solution enhances the existing CAPIF functional model by introducing the resource owner client and the authorization function, which is viable.

## 6.3 Solution #3: Obtaining resource owner consent upon service API invocation

### 6.3.1 Solution description

#### 6.3.1.1 General

This solution addresses the key issues #1, #2 and #3 with regard to obtaining resource owner consent in the UE- and AF-originated API invocation scenarios. Acting as the resource owner consent enforcement point, the AEF judges whether the invoked service API requires the resource owner consent and whether a valid resource owner consent is already included in the service API invocation. If the resource owner consent is required and a valid resource owner consent is not included in the API invocation, the AEF in collaboration with the resource owner client and the authorization function obtains a valid resource owner consent before the service API execution is continued. For the interaction between the authorization function and the resource owner client, the resource owner registration may optionally be performed prior to the service API invocation.

NOTE: Whether the resource owner registration is required and used for the purpose of obtaining resource owner authorization is to be decided by SA3.

Clause 6.3.1.2 shows the procedure for resource owner registration and clause 6.3.1.3 shows the procedure for obtaining resource owner consent.

#### 6.3.1.2 Resource owner registration

Figure 6.3.1.2-1 shows the procedure for resource owner registration.

Pre-conditions:

1. The resource owner client is authorised to access the authorization function; and

2. The resource owner client has information for accessing the authorization function.



Figure 6.3.1.2-1: Procedure for resource owner registration

1. The resource owner client sends resource owner registration request to the authorization function, including the resource owner identity (e.g. GPSI) and security credentials.

2. The authorization function initiates the verification procedure and validates the resource owner registration request.

3. The resource owner registration function sends resource owner registration response.

#### 6.3.1.3 Obtaining resource owner consent

Figure 6.3.1.3-1 shows the procedure to obtain resource owner consent upon service API invocation.

Pre-conditions:

1. The resource owner client is registered to the authorization function and can be accessed by the API exposing function.



Figure 6.3.1.3-1: Procedure for obtaining resource owner consent upon service API invocation

1. The API invoker sends service API invocation request to the API exposing function via CAPIF-2.

2. The API exposing function determine if the resource owner consent verification is required to execute the service API. If the resource owner consent verification is not required for the API invocation, the steps 3-5 will be skipped.

NOTE: The API exposing function determines if the resource owner consent verification is required based on implemented API logic.

The API exposing function identifies the resource owner client in 3GPP network context by one of the following processes:

- If the service API invocation indicates the resource owner client by GPSI, the API exposing function identifies the resource owner client by retrieving the resource owner information from the authorization function, which has received the resource owner information in resource owner registration, as shown in clause 6.3.1.2.

- If the service API invocation indicates the resource owner client by UE (IP or Ethernet) address, the API exposing function identifies the resource owner client by interacting with BSF, as specified in TS 29.513 [6].

- If the service API invocation indicates the resource owner client by external group identifier, the API exposing function identifies the individual resource owner clients by interacting with HSS/UDM, as specified in TS 23.682 [7].

- If the service API invocation indicates the resource owner client by "any UE" identification, the API exposing function regards all the available resource owner clients as the target resource owner clients.

3. The API exposing function obtains the resource owner consent from the resource owner client. The result of the resource owner consent response may be stored in the API exposing function.

NOTE: The detailed procedure to obtain the resource owner consent and manage such resource owner consent information will be specified in SA3.

4. If the API invocation is allowed, the process for service API execution is continued. If the API invocation is denied, the service API execution is rejected.

5. The API exposing function sends service API invocation response to the API invoker based on the result of the service API execution.

#### 6.3.1.4 Updating resource owner consent

The resource owner client may update resource owner consent anytime to allow or deny future API invocation.

NOTE: The detailed procedure to update the resource owner consent will be specified in SA3.

### 6.3.2 Solution evaluation

This solution addresses KI#3. This solution addresses the case where API invoker has access token given by the CAPIF core function (as per clause 8.11 of TS 23.222 [3]) to perform service API invocation, but without any authorization grant by the resource owner before the service API invocation.

NOTE: The API exposing function may need a solution to contact UE through firewall in Step 3 of clause 6.3.1.2.

## 6.4 Solution #4: API invoker obtaining resource owner consent

### 6.4.1 Solution description

This solution addresses the key issues #1, #2 and #3 with regard to obtaining resource owner consent in the UE- and AF-originated API invocation scenarios. CAPIF may authorize the API invoker to invoke the service API based on the resource owner consent slightly before the API invocation. The procedures to obtain the resource owner consent may reuse well-known authorization procedures such as OAuth 2.0 [9].

Clause 6.4.1.1 shows the procedure for obtaining resource owner consent in a near real-time manner.

#### 6.4.1.1 API invoker obtaining resource owner consent prior to the service API invocation

Figure 6.4.1.1-1 shows the procedure to obtain resource owner consent prior to the service API invocation, which is used in the near real-time consent scenario.

Pre-conditions:

1. The resource owner can communicate with the API invoker.



Figure 6.4.1.1-1: Procedure for API invoker obtaining resource owner consent prior to the service API invocation

1. The API invoker requests authorization grant and access token to invoke the service API.

NOTE: The detailed procedure to obtain resource owner consent and provide an access token is studied in SA3.

2. The API invoker sends service API invocation request to the API exposing function with the access token received in step 1.

### 6.4.2 Solution evaluation

This solution addresses KI#3. Unlike Solution #3, this solution requires that the API invoker get authorized to invoke the service API by the resource owner before sending the service API invocation request. The procedure to obtain the authorization grant and access token before the service API invocation is compliant with the ordinary OAuth 2.0 flow and is a viable solution to be considered.

## 6.5 Solution #5: UE-originated API invocation within CAPIF

### 6.5.1 Solution description

KI#1 describes the scenario of UE application client (AC) invokes 3GPP northbound APIs (e.g. NEF location API). In figure 6.5.1-1, the AC plays the role of the API invoker, and triggers a number of service operations with CCF and AEF (e.g. onboarding, security, API discovery).



Figure 6.5.1-1: Overview of CAPIF operations

The existing CAPIF model and procedures for UE-originated API invocation with CAPIF may be used but it is not so efficient since UE may be installed with many application clients (e.g. 100-200 apps). To reduce the number of CAPIF interactions per UE, a CAPIF client agent in the UE can be used as a common API invoker. Such CAPIF client agent executes the onboarding, authentication and authorization for security, service API discovery (discovered result can be re-used by all ACs), CAPIF event subscription with CCF.

NOTE 1: The interactions between ACs and CAPIF client agent (API invoker) in the UE for achieving per application granularity operation is out of scope.

NOTE 2: Security aspects including specification of the authentication and authorisation procedures for UE-originated API invocation with the CAPIF client agent are to be decided in SA3.

In short, the CAPIF client agent (as an application in the UE) is seen as an API invoker (see figure 6.5.1-2).



Figure 6.5.1-2: CAPIF client agent functionality in the UE

NOTE: Reference point detail between AC and CAPIF client agent is out of 3GPP scope.

### 6.5.2 Solution evaluation

This solution addresses KI#1. It introduces an option to have a common API invoker (i.e. CAPIF client agent) for the UE so that the signalling over CAPIF reference points can be optimized. Without enforcement in the UE, i.e. updates to the UE, the application developer can choose whether to use this CAPIF client agent. Therefore, from CAPIF's perspective, the CAPIF client agent is seen as a normal API invoker, so there is no impact on the current CAPIF architecture.

## 6.6 Solution #6: Discover a proper AEF with owner information

### 6.6.1 Solution description

The API provider hosts one or more service APIs and has a service API arrangement with CAPIF provider to offer the service APIs to the API invoker, as described in 3GPP TS 23.222 [3]. A CAPIF provider may have business relationships with multiple API providers (e.g. MNOs).

The AF knows UE IP address from the Application traffic. If the AF wants to contact with an API exposing function (AEF) provided by MNO but does not know which MNO the UE belongs to, the AF can utilize the information provided by Regional Internet Registry (RIR). e.g. Asia-Pacific Network Information Centre (APNIC) is responsible for providing allocation and administration services to Asia-Pacific's Internet service providers (ISPs) and Internet users. IP "whois" service can be used to retrieve IP address related information, for instance, with input of IP address 223.167.255.255 in <https://wq.apnic.net/apnic-bin/whois.pl> it gives the following result including the owner for such IP address (only partial result is listed here for relevant info):

% APNIC found the following authoritative answer from: whois.apnic.net

% [whois.apnic.net]

% Whois data copyright terms <http://www.apnic.net/db/dbcopyright.html>

% Information related to '223.166.0.0 - 223.167.255.255'

% Abuse contact for '223.166.0.0 - 223.167.255.255' is 'hqs-ipabuse@chinaunicom.cn'

inetnum: 223.166.0.0 - 223.167.255.255

netname: UNICOM-SH

descr: CHINA UNICOM Shanghai city network

descr: China Unicom

descr: No.21,Jin Rong Street,Beijing,100033

descr: P.R.China

country: CN

admin-c: [CH1302-AP](http://wq.apnic.net/apnic-bin/whois.pl?searchtext=CH1302-AP&form_type=advanced)

tech-c: [CH1302-AP](http://wq.apnic.net/apnic-bin/whois.pl?searchtext=CH1302-AP&form_type=advanced)

remarks: service provider

mnt-by: [APNIC-HM](http://wq.apnic.net/apnic-bin/whois.pl?searchtext=APNIC-HM&form_type=advanced)

mnt-lower: [MAINT-CNCGROUP-SH](http://wq.apnic.net/apnic-bin/whois.pl?searchtext=MAINT-CNCGROUP-SH&form_type=advanced)

mnt-routes: [MAINT-CNCGROUP-RR](http://wq.apnic.net/apnic-bin/whois.pl?searchtext=MAINT-CNCGROUP-RR&form_type=advanced)

status: ALLOCATED PORTABLE

mnt-irt: IRT-CU-CN

last-modified: 2013-08-08T23:17:38Z

source: APNIC

Noting that such interaction with IP address administrative authority is out of the scope of 3GPP.

Once the AF knows which MNO it should contact for the API invocation related to a UE, the AF queries the CAPIF core function (CCF) with required API provider name (e.g. China Mobile Beijing) and CCF responds with the AEF information matching the required API provider name. The CCF knows the API provider name either via API management function triggered Registration procedure for the whole API provider domain functions or via the API publish function triggered service API information publish procedure for the AEF offered API(s). The impact for CAPIF procedures is illustrated in figure 6.6.1-1, and tables 6.6.1-1, table 6.6.1-2 and 6.6.1-3 describe the information flow impact with highlighted IEs.



Figure 6.6.1-1: API provider name used in CAPIF procedures

Table 6.6.1-1: Registration request

|  |  |  |
| --- | --- | --- |
| Information element | Status | Description |
| List of API provider domain functions | M | List of API provider domain functions including role (e.g. AEF, APF, AMF) and, if required, specific security information. |
| **API provider name** | **O** | **The API provider name uniquely identifies an API provider (e.g. Internet Service Provider).** |
| Security information | M | Information for CAPIF core function to validate the registration request |

Table 6.6.1-2: Service API publish request

|  |  |  |
| --- | --- | --- |
| Information element | Status | Description |
| API publisher information | M | The information of the API publisher may include identity, authentication and authorization information |
| Service API information | M | The service API information includes the service API name, **API provider name (optional),** service API type, communication type, description, Serving Area Information (optional), AEF location (optional), interface details (e.g. IP address, port number, URI), protocols, version numbers, and data format. |
| Shareable information | O (see NOTE) | Indicates whether the service API or the service API category can be published to other CCFs. And if sharing, a list of CAPIF provider domain information where the service API or the service API category can be published is contained. |
| NOTE: If the shareable information is not present, the service API is not allowed to be shared. | | |

Table 6.6.1-3: Service API discover request

|  |  |  |
| --- | --- | --- |
| Information element | Status | Description |
| API invoker identity information | M | Identity information of the API invoker discovering service APIs |
| Query information | M | Criteria for discovering matching service APIs (e.g. service API type, Serving Area Information (optional), preferred AEF location (optional), **required API provider name (optional)**, interfaces, protocols)  (see NOTE) |
| NOTE: It should be possible to discover all the service APIs. | | |

The API provider name can be shared between CCFs over CAPIF-6/6e reference point and the interconnection service API discovery may also include the required API provider name as query parameter.

Alternatively, to reduce the burden on API invoker side, API invoker may include subscriber IP address in the service API discover request in step 6 above and CCF uses implementation specific means (e.g. whois) to resolve the subscriber IP address to its owner information (e.g. API provider name) and then find the matching service API information with the owner information. CCF may also receive from the API provider, the mapping information between the subscriber IP address range information and the AEF information. For example, a NEF provider may share the IP address range information related to its subscribers that NEF provides the service to, while publishing the APIs at CCF.

### 6.6.2 Solution evaluation

This solution addresses KI#4, it enables the API invoker to discover a proper AEF with API provider name or subscriber IP address for invoking an API related to UE. It is assumed that all AEF instances provided by the API provider know their owner's information and API invoker knows subscriber IP address. This solution is a viable CAPIF solution to be considered.

## 6.7 Solution #7: Reducing resource owner consent inquiry in a nested API invocation

### 6.7.1 Solution description

This solution addresses the key issue #3 with regard to reduce resource owner consent inquiries for a nested API invocation, in which the first API invocation towards the API exposing function 1 triggers this API exposing function to request another API invocation towards the API exposing function 2, which is in the same API provider domain that the API exposing function 1. Some service APIs may require invoking another service APIs. For example, if the API invoker invokes SEAL locationInfoRetrieval API, the location management server (acting as an API exposing server for the API invoker and as an API invoker for the NEF) may invoke NEF API to retrieve UE location information from 5GC. This solution suggests how the CAPIF entities obtain the resource owner consent if more than one API exposing function is involved in the procedure, specifically when these API exposing functions are in the same trust domain.

#### 6.7.1.1 Reducing resource owner consent inquiry in a nested API invocation

Figure 6.7.1.1-1 shows the procedure to obtain resource owner consent in a nested API invocation, in which an API exposing function receiving the service API invocation request interacts with another API exposing function to provide the service.

Pre-conditions:

1. The resource owner can communicate with the API invoker.

2. The API exposing functions 1 and 2 are in the same trust domain.



Figure 6.7.1.1-1: Procedure for obtaining resource owner consent in a nested API invocation

1. The API invoker requests authorization grant and access token to invoke the service API exposed by the API exposing function 1.

NOTE: This step may use either the existing procedure to obtain authorization to access service API specified in TS 23.222 [3] or the procedure that involves the resource owner client to get resource owner consent. For the latter case, the detailed procedure to obtain resource owner consent and provide an access token is studied in SA3.

2. The API invoker sends service API invocation request to the API exposing function 1 with the access token received in step 1.

3. Based on the service API invocation request, the API exposing function 1 decides to invoke another service API exposed by the API exposing function 2.

4. The API exposing function 1, acting as an API invoker, obtains the access token to access the service API exposed by the API exposing function 2.

NOTE: The existing mechanisms such as OAuth 2.0 Token Exchange [10] can be used to obtain the access token for the nested API invocation. The detailed procedure for authorization for nested API invocations in this step will be studied in SA3

5. The API invoker sends service API invocation request to the API exposing function with the access token received in step 1.

### 6.7.2 Solution evaluation

This solution clarifies how to obtain the resource owner consent in a nested API invocation. The step 4 in the clause 6.7.1.1 outlines a specific case where the API exposing functions 1 and 2 are in the same trust domain and the CAPIF can provide authorization for nested API invocation. The detailed security aspects for this case will be studied in SA3.

# 7 Overall evaluation

## 7.1 General

The following clauses contain an overall evaluation of the solutions presented in this technical report, and their applicability to the identified key issues.

- Clause 7.2 provides an evaluation of the business relationship and the functional model studied in clauses 6.1 and 6.2; and

- Clause 7.3 lists the solutions for the key issues including impact on other working groups that will need consideration.

## 7.2 Architecture evaluation

The business relationship solution (Solution #1) in clause 6.1 describes the enhanced CAPIF business relationship introducing the resource owner as a new stakeholder. The functional model solution (Solution #2) in clause 6.2 describes the enhanced CAPIF functional model applicable for both the UE- and AF-originated API invocation scenarios. A summary of the architecture and key issues specified in this technical report are listed in table 7.2-1.

Table 7.2-1: Architecture evaluation

|  |  |  |
| --- | --- | --- |
| Architecture solution | Applicable key issues (clause reference) | Dependency on other working groups |
| Solution #1: Business relationship in SNA | Supports all key issues specified in clause 4 | None |
| Solution #2: Functional model description for SNA | Supports all key issues specified in clause 4 | SA3 |

Both the business relationship and the functional model are viable and will be considered in the normative work.

## 7.3 Key issue and solution evaluation

### 7.3.1 Introduction

The mapping of solutions to key issues in this technical report is listed in table 6.0-1. In addition, Table 7.3.1-1 lists the impacts to other working groups that will need consideration during the Rel-18 normative phase.

Table 7.3.1-1: Key issues, solutions, and dependency on other working groups

|  |  |  |
| --- | --- | --- |
| Key issues | Solution | Dependency on other working groups |
| Key Issue #1: UE-originated API invocation | Solution #3: Obtaining resource owner consent upon service API invocation | SA3 |
| Solution #4: API invoker obtaining resource owner consent | SA3 |
| Solution #5: UE-originated API invocation within CAPIF | SA3 |
| Key Issue #2: AF-originated API invocation | Solution #3: Obtaining resource owner consent upon service API invocation | SA3 |
| Solution #4: API invoker obtaining resource owner consent | SA3 |
| Key Issue #3: Providing and revoking resource owner consent upon invoking APIs | Solution #3: Obtaining resource owner consent upon service API invocation | SA3 |
| Solution #4: API invoker obtaining resource owner consent | SA3 |
| Solution #7: Reducing resource owner consent inquiry in a nested API invocation | SA3 |
| Key Issue #4: Discovery of target API information | Solution #6: Discover a proper AEF with owner information | None |

### 7.3.2 Overall evaluation of solutions for Key Issue #1

Key Issue #1 is an issue about the CAPIF functional model to support the UE-originated API invocation. Three solutions are proposed in this document.

Solutions #3 and #4 propose procedures to support providing and revoking the resource owner consent. These solutions have no normative impact regarding the Key Issue #1.

Solution #5 proposes that a CAPIF client agent be introduced to reduce the number of interactions between the UE and the CAPIF entities. This solution is proposed specifically for the UE-originated API invocation scenario but does not impact the existing CAPIF architecture.

For Key Issue #1, the security aspects including specification of the authentication and authorisation procedures for UE-originated API invocation within CAPIF are to be decided in SA3.

### 7.3.3 Overall evaluation of solutions for Key Issue #2

Key Issue #2 is an issue about the CAPIF functional model to support the AF-originated API invocation. Two solutions are proposed in this document.

Solutions #3 and #4 propose procedures to support providing and revoking the resource owner consent. These solutions have no normative impact regarding the Key Issue #2.

### 7.3.4 Overall evaluation of solutions for Key Issue #3

Key Issue #3 is an issue about providing and revoking resource owner consent upon invoking APIs. Three solutions are proposed in this document.

Solution #3 proposes procedures to support providing and revoking the resource owner consent. The procedure in the clause 6.3.1.3 assumes that the resource owner consent is, if needed, provided after the service API invocation request from the API invoker. To enable the interaction between the resource owner client and the authorization function after the service API invocation request from the API invoker, the resource owner registration may optionally be performed beforehand. Even if the resource owner client has performed the resource owner registration, the API exposing function may need a solution to contact UE through firewall.

Solution #4 also proposes a procedure to support providing and revoking the resource owner consent. Unlike Solution #3, this solution assumes that the API invoker get authorized to invoke the service API by the resource owner before sending the service API invocation request. The procedure to obtain the authorization grant and access token before the service API invocation is compliant with the ordinary OAuth 2.0 flow and is a viable solution to be considered.

Solution #7 proposes a procedure to reduce the resource owner consent inquiries in a nested API invocation. This solution suggests that the existing mechanisms such as OAuth 2.0 Token Exchange [10] can be used when obtaining the access token for the nested API invocation. The detailed procedure will be studied in SA3.

Whether and how security related mechanisms for these solutions can be specified should be decided by SA3.

### 7.3.5 Overall evaluation of solutions for Key Issue #4

Key Issue #4 is an issue about discovery of target API information. One solution is proposed in this document.

Solution #6 proposes a procedure to enable the API invoker to discover a proper AEF with API provider name or subscriber IP address for invoking an API related to UE. The CCF receives the API provider name and subscriber IP address range information from the API provider to discover the proper AEF. This solution supports proper API discovery in cases where the AF is serving UE belonging to multiple PLMNs and there are multiple instances of the AEF within a PLMN.

The solutions above will be considered in the normative work.

# 8 Conclusions

This technical report fulfills the objectives of the study on application enablement aspects for subscriber-aware northbound API access. The results from the study will be considered for the follow-up normative work in Release 18 as follows:

1. The architectural requirements (clause 5) will be considered the basis for the technical specification;

2. The architecture solutions including the business relationship (clause 6.1) and the functional models (clause 6.2) will be considered the enhancements to the existing CAPIF specification;

3. The individual solutions (clause 6) will be considered the candidate solutions with necessary enhancements as appropriate, according to the overall evaluation (clause 7).

SA3 feedback will be considered for progressing the architecture related solutions (e.g. enhanced CAPIF functional models and deployment models) to the normative phase.

Annex A (informative):  
Use case examples

# A.1 AF-originated API invocation (Gaming)

## A.1.1 General

This use case is an example of AF-originated API invocation with a gaming application. In this use case, the end user (also a subscriber of the MNO) allows the AF (game provider's server) to invoke the QoS API (offered by MNO) to modify the QoS of the end user.

## A.1.2 Pre-conditions

An end user (also a subscriber of the MNO) is playing a time-sensitive game using a game client application on the end user’s UE communicating with a game provider's server. The end user wants to have a high-quality and low-latency communication for better service experience, so the game server (AF or API invoker) tries to invoke the QoS API provided by the 5GC of the MNO to change the end user's QoS according to the request from the game client application on the end user's UE. Changing the QoS may affect the charging rate to the end user, so the game server needs to get authorized to invoke the API by the end user.

## A.1.3 Service flows

1. The game server triggers an authorization procedure of the QoS API provider where the MNO subscriber (end user) is asked to confirm whether the game server can invoke QoS API with extra charge.

2. The MNO subscriber (end user) authorizes the game server to apply the QoS change with extra charge.

## A.1.4 Post-conditions

After receiving this authorization as per the authorization procedure, the game server invokes the QoS API.

NOTE: This is an example of real-time or near real-time request of authorization, but the game server may also use the authorization information given by the MNO subscriber in the past authorization procedure.

# A.2 UE-originated API invocation (Location tracking)

## A.2.1 General

This use case is an example of UE-originated API invocation with a location tracking application. In this use case, the end user (also a subscriber of the MNO) on UE X allows the end user on UE Y to invoke an API to track the location of the end user on UE X.

## A.2.2 Pre-conditions

A tracking application enables the user on UE Y to track the location of a user on UE X. An API Provider AP provides location APIs for the end users on UE X and UE Y, and the tracking application on the UE utilizes the location APIs to provide the tracking functionality.

## A.2.3 Service flows

1. Tracking application of UE Y triggers the AP to obtain consent from end user on UE X.

2. Consent is obtained from the end user on UE X via an authorization procedure of the AP allowing “Tracking application on UE Y" to invoke the location API for UE X exposed by AP (UE-originated API invocation as shown in Figure 4.1.1-1).

## A.2.4 Post-conditions

After receiving the authorization as per the authorization procedure of the AP, the tracking application on UE Y invokes the location API of AP for obtaining location of user on UE X.

Annex B (informative):  
Change history

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 2021-07 | SA6#44-e | S6-211548 |  |  |  | TR skeleton | 0.0.0 |
| 2021-07 | SA6#44-e |  |  |  |  | S6-211821, S6-211852, S6-211853, S6-211854, S6-211855, S6‑211856 | 0.1.0 |
| 2021-09 | SA6#45-e |  |  |  |  | S6-211871, S6-212075, S6-212076, S6-212165 | 0.2.0 |
| 2021-10 | SA6#45-bis-e |  |  |  |  | S6-212242, S6-212475 | 0.3.0 |
| 2021-11 | SA6#46-e |  |  |  |  | S6-212809, S6-212810, S6-212811, S6-212812, S6-212826 | 0.4.0 |
| 2021-11 | SA#94-e | SP-211508 |  |  |  | Presentation for information at SA#94-e | 1.0.0 |
| 2022-02 | SA6#47-e |  |  |  |  | S6-220449, S6-220450, S6-220451, S6-220452, S6-220453 | 1.1.0 |
| 2022-04 | SA6#48-e |  |  |  |  | S6‑220774, S6‑220775, S6‑220776, S6‑220873, S6‑220944, S6‑220945, S6‑220946, S6‑220947 | 1.2.0 |
| 2022-05 | SA6#49-e |  |  |  |  | S6‑221109, S6‑221110, S6‑221111, S6‑221203, S6‑221365, S6‑221366, S6‑221367, S6‑221457, S6‑221458 | 1.3.0 |
| 2022-07 | SA6#49-bis-e |  |  |  |  | S6-221559 | 1.4.0 |
| 2022-08 | SA6#50-e |  |  |  |  | S6-222397 | 1.5.0 |
| 2022-10 | SA6#51-e |  |  |  |  | S6-222934 | 1.6.0 |
| 2022-11 | SA6#52 |  |  |  |  | S6-223176 | 1.7.0 |
| 2022-12 | SA#98-e | SP-221224 |  |  |  | Submitted for Approval at SA#98-e | 2.0.0 |
| 2022-12 | SA#98-e | SP-221224 |  |  |  | MCC Editorial update for publication after TSG SA approval (SA#98‑e) | 18.0.0 |
| 2023-03 | SA#99 | SP-230281 | 0001 | 1 | D | Updates on solution #5 evaluation | 18.1.0 |