AID.02 Automotive Identity Technical Specification
Version 2.1
26 January 2024

Security Classification: Non-confidential

Copyright Notice

Disclaimer

Compliance Notice

The information contain herein is in full compliance with the GSM Association's 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 in GSMA AA.35 – Procedures for Industry Specifications.

.

Table of Contents

1 Introduction

This document specifies two interfaces as part of the framework to enable mobile, content and automotive services to be easily federated and provisioned into cars and accessed via authentication.

1. The interface between Mobile Service Provider (MSP) and Car Mobility Provider (CMP), also referred to as the AiD1 interface.
2. The interface between the Car Mobility Provider (CMP) and the CMP app [car OS] on the vehicle, also referred as the AiD2 interface.

In particular, one of the key aims is to enable individuals to seamlessly use their mobile subscription and mobile services via an in-car system and consumer eSIM across multiple cars.

1.1 Scope

This document specifies the backend interface between CMP and MSP which allows to facilitate the account federation. The account federation enables the individual to provision their mobile subscription along with content and automotive services into multiple cars through a unified login.

In addition, this document also specifies the interface between the CMP backend and the CMP app on the vehicle (also referred to as the car OS). The aim here is to simplify the integration of vehicles with the backend of different CMPs. This would allow to integrate one vehicle design with the backend of different OEMs (e.g. when the vehicle is designed for different brands in an OEM cooperation) or to integrate the vehicle with other CMPs who are not the OEM. An example here could be rental car or car sharing companies who provide the AiD service for their fleet.

In Scope	Out of Scope
• Automotive identity (AID) service definition & proposition • Definition of roles/actors & user stories • Core processes • Additional considerations (security; privacy etc.) • Account federation between MSP and CMP • Car provisioning	▪ Go-To-Market business models and considerations ▪ Methods of authentication (implementation-specific)

Table 1: Document scope

NOTE: Whilst the AID framework is defined using automotive as a leading use case to guide the user stories and functional requirements, the framework is extensible to other industry sectors and use cases.

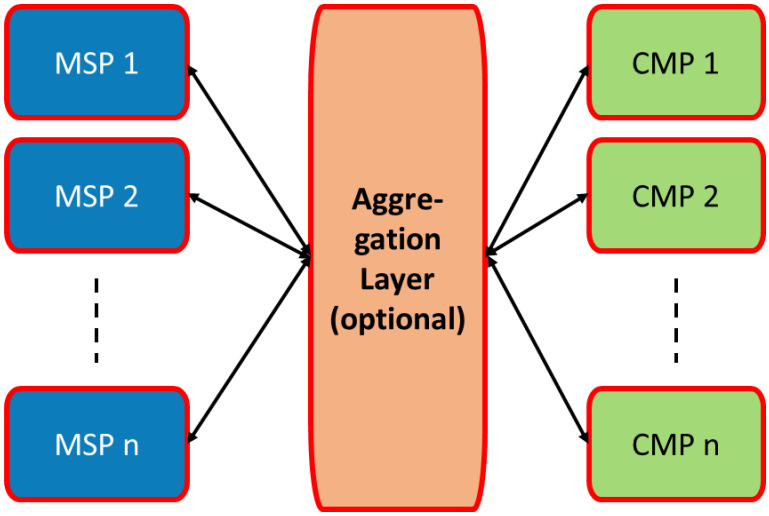


Figure 1: Optional Aggregation Layer in AID1 interface

The interface specification for the AID1 interface will allow the implementation of an aggregation or mediation layer between MSP and CMP, so that each MSP has to deploy only one interface to the aggregation layer in order to interact with many CMPs and vice versa. However, the implementation of such aggregation layer will be optional.

1.2 Abbreviations

Term	Description
AID	Automotive Identity
AUP	AID User Profile
CMP	Car Mobility Provider (such as fleet management, automotive manufacturer or car sharing provider)
CSIM	Consumer SIM
eUICC	Embedded Universal Integrated Circuit Card
MSP	Mobile Service Provider
OP	RSP Operational Profile
SM-DP+	Subscription Manager - Data Preparation +
AC	Activation Code

Table 2: Abbreviations

1.3 Defintions

Term	Description
AID Service	Service provided to a user that enables them to provision a car with usage of a mobile subscription and a collection of services from the MSP, CMP and other Value Added Service Providers and access these capabilities and services within different cars

Term	Description
AID User Profile	Single Profile encompassing the user's services & preferences and MSP metadata[1] that is downloaded into a car to enable that car to be personalised to that user
Operational Profile	As it is defined in SGP.21 [1].
Activation Code	Activation Code according to GSMA SGP.22 RSP Technical Specification [2]. Information issued by an MSP to request a download of an Operational Profile

Table 3: Definitions

1.4 References

Ref	Doc Number	Title
[1]	SGP.21	RSP Architecture Specification Version 2.4
[2]	SGP.22	RSP Technical Specification Version 2.4
[3]	GPC_SPE_034	GlobalPlatform Technology Card Specification Version 2.3.1

Table 4: References

1.5 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 RFC2119 https://www.ietf.org/rfc/rfc2119.txt

2 Actors

The following actors are defined within the AID framework:

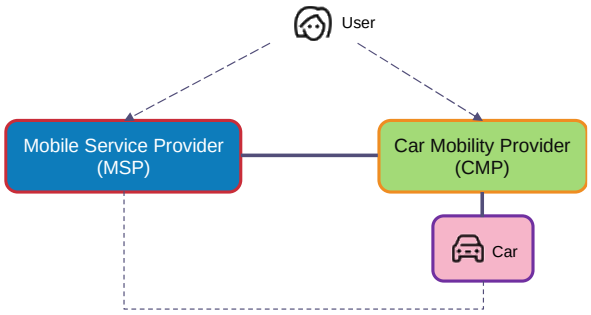


Figure 2 AID actors

User:

- An individual using the AID service within one or more Cars

[1] E.g., MSP name, icon etc. for display via the Car HMI

- Establishes or uses an existing mobile subscription with a Mobile Service Provider (MSP) for use with the AID service
- Registers with a Car Mobility Provider (CMP) able to support the Car(s) in which the user wishes to use the AID service
- Generates an AID User Profile with the CMP for personalising the Car(s) to the user
- Links their mobile subscription to the AID User Profile for use in the Car(s)
- Manages the AID service via the MSP and CMP as appropriate

MSP (Mobile Service Provider):

- Provides the user with a mobile subscription associated with the AID service
- Onboards the user to the AID service in conjunction with the CMP
- Provides information as needed to the CMP for generating the AID User Profile with the user and setting up the AID service
- Provisions the Car eUICC with an Operational Profile (OP) for the mobile subscription on user request
- Supports lifecycle management of the AID service

CMP (Car Mobility Provider):

- Onboards the user in conjunction with the MSP to the AID service
- Enables the user to set up their AID User Profile for use within the Car(s) linked to the CMP
- Provisions the AID User Profile to the target vehicle
- Supports lifecycle management of the AID service

Car:

- Interfaces with a CMP and MSP to support:
    - Personalisation of the Car to the user based on the AID User Profile downloaded to the Car
    - Provisioning of an RSP Operational Profile into the Car eUICC to enable the user to access their mobile subscription via the Car
- Provides an authentication mechanism through which the user can access their AID User Profile and associated OP in the Car
- Supports an AID User Profile (and OP) per individual hence enabling multiple people to have personalised use of the Car
- The interfaces specified in this document are not using the connectivity which is provided by the AiD Operational Profiles. The car has to be equipped with another connectivity for AiD administration and profile handling purposes. This can e.g. be the normal machine type telematic connectivity of the car, a WiFi connectivity tethered to a smartphone, or any other IP based connectivity.

3 Framework Architecture

3.1 Architecture Overview

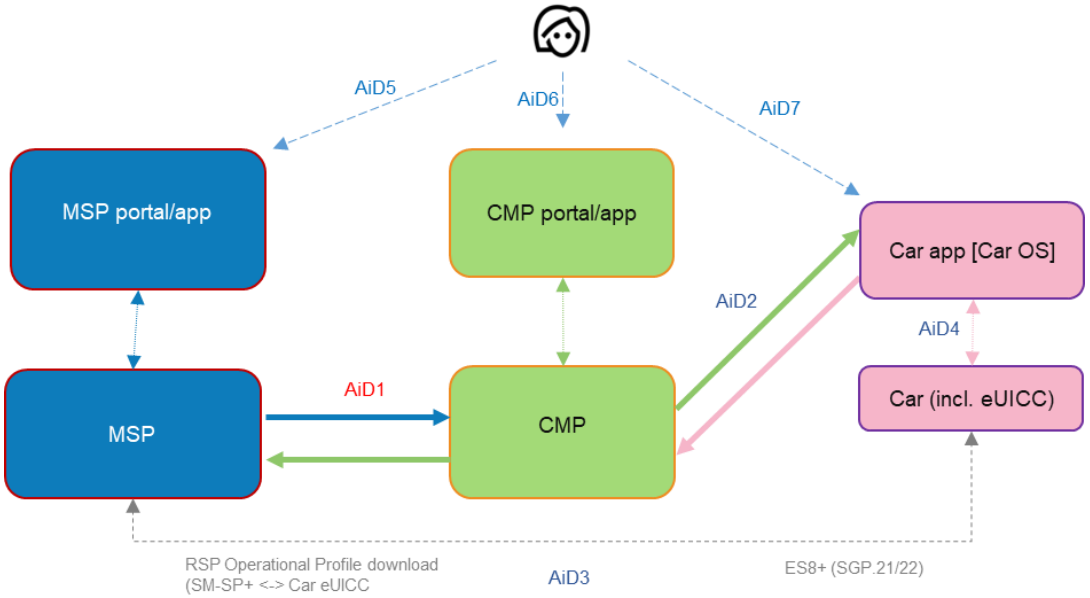


Figure 3: Architecture overview

3.2 Interfaces

AID1 (MSP <-> CMP)

- Managing user onboarding, Car provisioning (depending on Category)
- AID service lifecycle management

AID2 (CMP <-> CMP app [Car OS])

- Downloading the AID User Profile to the Car
- Providing information to enable the CMP app to instigate provisioning of the user's mobile subscription to the Car (where applicable)
- Operational Profile lifecycle management

AID3 (MSP <-> Car eUICC)

- Secure transport for the delivery of an Operational Profile between the SM-DP+ and the Car eUICC [ES8+ SGP.21/22]

AID4 (Car <-> CMP application [Car OS])

- Exposure of APIs to a CMP application that runs in the Car OS for AID service lifecycle management

AID5 (User <-> MSP)

- Onboarding the user to the AID service and in-life service management (implementation specific user interface)

AID6 (User <-> CMP portal)

- Onboarding the user to the AID service and in-life service management (implementation specific user interface)

AID7 (User <-> CMP app [Car OS])

- User authentication to the AID service (Car entry), logout (Car exit) and management of the AID service within the Car (proprietary user interface administered by the CMP towards the user)

4 Procedures

4.1 Onboarding via CMP Touchpoint (e.g. app)

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

actor "User"
participant "Vehicle" as vehicle
participant "CMP User Touchpoint" as touchpoint


participant "CMP Backend" as backend
box "MSP"
     participant "MSP touchpoint"
     participant "Backend" as MSP
end box


activate User
activate  touchpoint
activate backend
User -> touchpoint : Login and start onboarding
User -> touchpoint : Choose MSP
touchpoint -> backend: Request account id (MSP)
backend --> touchpoint: account id
deactivate backend
touchpoint -> "MSP touchpoint" : Redirect to MSP content \n(account ID)
deactivate touchpoint

activate "MSP touchpoint"
activate MSP
User -> "MSP touchpoint": Login at MSP
"MSP touchpoint" -> MSP: Account ID
"MSP" -> "MSP": Verify account ID
"MSP" -> "MSP touchpoint": Account ID verified
User -> "MSP touchpoint": Provide information for onboarding
"MSP touchpoint" -> MSP: Onboarding information
"MSP" -> "MSP": Verify onboarding information \nand provision service
activate backend
MSP -> backend: Send MSP token \n(MSP token, account ID)
backend --> MSP: Send MSP token response
MSP -> "MSP touchpoint": Onboarding result
activate touchpoint
"MSP touchpoint" -> touchpoint: Redirect back to CMP touchpoint
deactivate touchpoint
deactivate backend
deactivate "MSP touchpoint"
```

```
deactivate MSP


@enduml
```

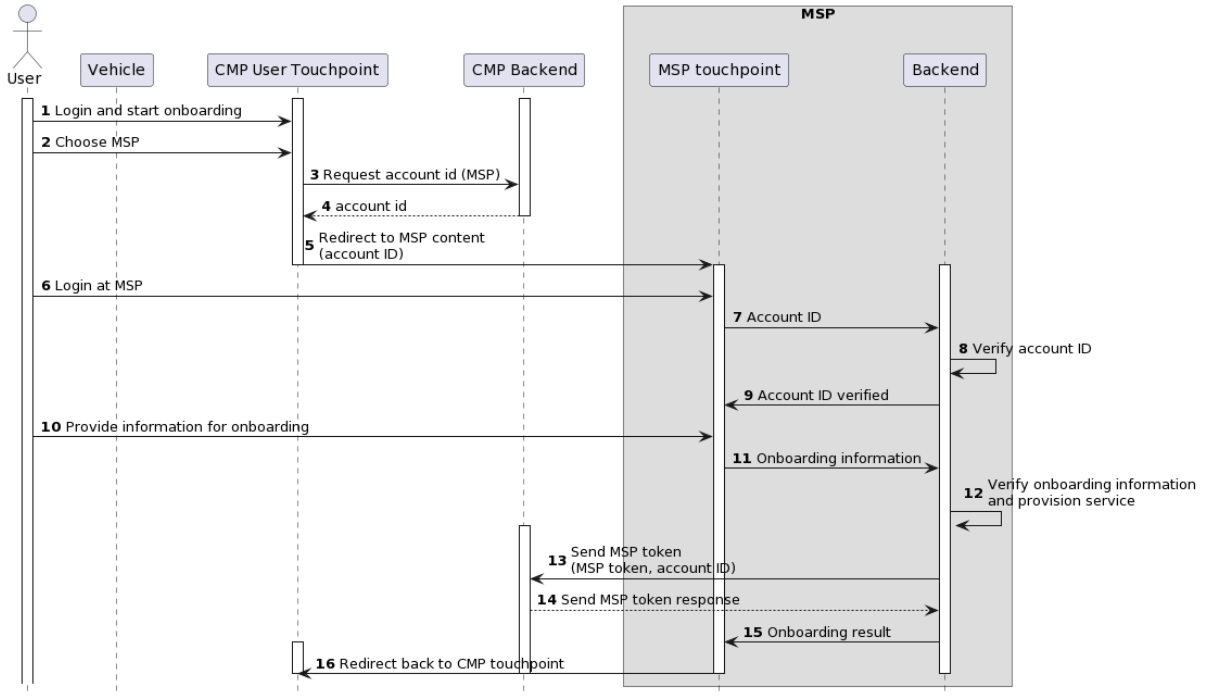


Figure 4: MSP Onboarding via CMP Touchpoint procedure

Start conditions:

a) User has an active account with the CMP with valid account credentials
b) User has an active account with the MSP with valid account credentials

Procedure:

[1] User logs in at a CMP touchpoint (CMP app or web portal)

[2] User selects an MSP. The CMP might offer the user a variety of MSPs for which the AID account federation is possible.

[3] The touchpoint requests an ‘account id’ for the particular user from the CMP backend. The ‘account id’ identifies the CMP user account and can be shared with the MSP without sharing any account credentials.

[4] The CMP backend returns the ‘account id’ to the CMP touchpoint.

[5] The CMP touchpoint redirects the user to an MSP touchpoint. Within this redirect, the ‘ account id’ is shared with the MSP.

[6] The user now needs to login to their MSP account.

[7] The MSP touchpoint sends the ‘account id’ to his own backend.

[8] The MSP backend verifies the ‘account id’. In case a secret knowledge is agreed within the ‘account id’ (see parameter ‘ver’ in format of accout_id, chapter 5.2) it can be verified here. Otherwise, only the format will be verified.

[9] The MSP touchpoint is informed about the verification of the ‘ account id’.

[10] The user then decides on adding the AID option to his contract and confirms potential fees (the upsell for the MSP).

[11] The onboarding or upsell information is sent to the MSP backend for further processing.

[12] The onboarding (upsell) information is processed in the MSP backend and the AID service is provisioned for the particular user on the MSP side. An ‘MSP token’ is created to identify the user during the lifecycle of the service.

[13] The ‘MSP token’ together with the ‘account id’ is sent to the CMP backend. The CMP backend uses the “account id’ to identify the correct CMP user account. The ‘MSP token’ will then be stored within the user account data and used for further communication with the MSP. MSP and CMP account are now federated.

[14] The CMP backend confirms receiving the ‘MSP token’.

[15] The MSP backend informs the MSP touchpoint that the onboarding is competed.

[16] The MSP touchpoint may confirm the successful onboarding to the user and redirect the user to the CMP touchpoint.

End conditions:

a) Both the CMP and the MSP account of the user are federated, meaning both accounts are linked by the MSP token.
b) The user is able to log into any of the CMPs vehicle and provision this vehicle for the AID service.

4.2 Onboarding via MSP Touchpoint

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

actor "User"
participant "Vehicle" as vehicle
participant "CMP User Touchpoint" as touchpoint


participant "CMP Backend" as backend
box "MSP"
     participant "MSP touchpoint"
     participant "Backend" as MSP
end box


activate User
activate "MSP touchpoint"
```

```
User -> "MSP touchpoint" : Login and start onboarding
User -> "MSP touchpoint" : Choose CMP
activate touchpoint
"MSP touchpoint" -> touchpoint: Redirect to CMP content (redirect to MSP URL)
User -> touchpoint : Login to CMP
activate backend
touchpoint -> backend: Request account id (MSP)
backend --> touchpoint: account id
deactivate backend
touchpoint -> "MSP touchpoint" : Redirect back to MSP content \n(account ID)
deactivate touchpoint

activate MSP
"MSP touchpoint" -> MSP: account ID
"MSP" -> "MSP": Verify account ID
"MSP" -> "MSP touchpoint": account ID verified
User -> "MSP touchpoint": Provide information for onboarding
"MSP touchpoint" -> MSP: Onboarding information
"MSP" -> "MSP": Verify onboarding information \nand provision service
activate backend
MSP -> backend: Send MSP token \n(MSP token, account ID)
backend --> MSP: Send MSP token response
MSP -> "MSP touchpoint": Onboarding result

deactivate backend
deactivate "MSP touchpoint"
deactivate MSP

@enduml
```

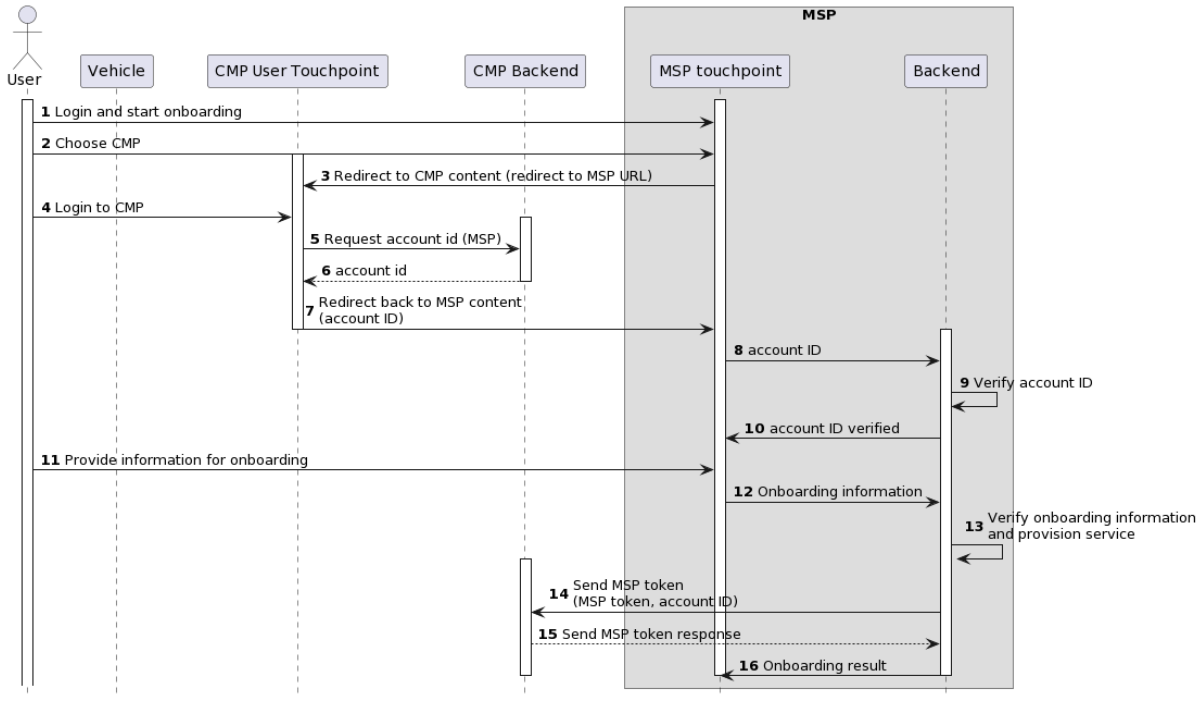


Figure 5: Onboarding via MSP Touchpoint procedure

Start conditions:

a) User has an active account with the CMP with valid account credentials
b) User has an active account with the MSP with valid account credentials

Procedure:

[1] User logs in at a MSP touchpoint (MSP app or web portal)

[2] User selects a CMP. The MSP might offer the user a variety of CMPs for which the AID account federation is possible.

[3] The user is redirected to a CMP touchpoint (app or portal). As a path parameter, a return URL to redirect to the MSP is passed to the CMP.

[4] The user logs into his CMP account.

[5] The touchpoint requests an ‘account id’ for the particular user from the CMP backend. The ‘account id’ identifies the CMP user account and can be shared with the MSP without sharing any account credentials.

[6] The CMP backend returns the ‘account id’ to the CMP touchpoint.

[7] The CMP touchpoint redirects the user back to the MSP touchpoint. Within this redirect, the ‘account id’ is shared with the MSP.

[8] The MSP touchpoint sends the ‘account id’ to his own backend.

[9] The MSP backend verifies the ‘account id’. In case a secret knowledge is agreed within the ‘account id’ (see parameter ‘ver’ in format of accout_id, chapter 5.2) it can be verified here. Otherwise, only the format will be verified.

[10] The MSP touchpoint is informed about the verification of the ‘account id’.

[11] The user then decides on adding the AID option to his contract and confirms potential fees (the upsell for the MSP).

[12] The onboarding or upsell information is sent to the MSP backend for further processing.

[13] The onboarding (upsell) information is processed in the MSP backend and the AID service is provisioned for the particular user on the MSP side. An ‘MSP token’ is created to identify the user during the lifecycle of the service.

[14] The ‘MSP token’ together with the ‘account id’ is sent to the CMP backend. The CMP backend uses the “account id’ to identify the correct CMP user account. The ‘MSP token’ will then be stored within the user account data and used for further communication with the MSP. MSP and CMP account are now federated.

[15] The CMP backend confirms receiving the ‘MSP token’.

[16] The MSP backend informs the MSP touchpoint that the onboarding is competed.

End conditions:

a) Both the CMP and the MSP account of the user are federated, meaning both accounts are linked by the MSP token.

b) The user is able to log into any of the CMPs vehicle and provision this vehicle for the AID service.

4.3 Car Provisioning

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

actor "User"

participant "Vehicle" as vehicle

participant "CMP Backend" as backend
box "MSP"
     participant "Backend" as MSP
     participant "SM-DP+" as smdp
end box

activate "User"
activate vehicle
User -> vehicle : Login
|||
deactivate User

|||
activate backend
vehicle -> backend: Request activation code
activate MSP
backend -> MSP: Request activation code
MSP -> MSP: Bind Operational Profile to eid
MSP --> backend: Send activation code
deactivate MSP
backend -> vehicle: Send activation code
activate smdp
vehicle -> smdp: Request profile download
smdp --> vehicle: Operational profile
deactivate smdp
vehicle -> vehicle: Install profile
vehicle -> backend: Send Profile Download Install Info
vehicle -> vehicle: Enable profile
vehicle -> backend : Send CSIM status (EID, ICCID, status="profile enabled")
|||
activate User
User -> vehicle: Logout
deactivate User
vehicle -> vehicle: Disable profile
vehicle -> backend: Send CSIM status (EID, ICCID, status="profile disabled")
deactivate backend
@enduml
```

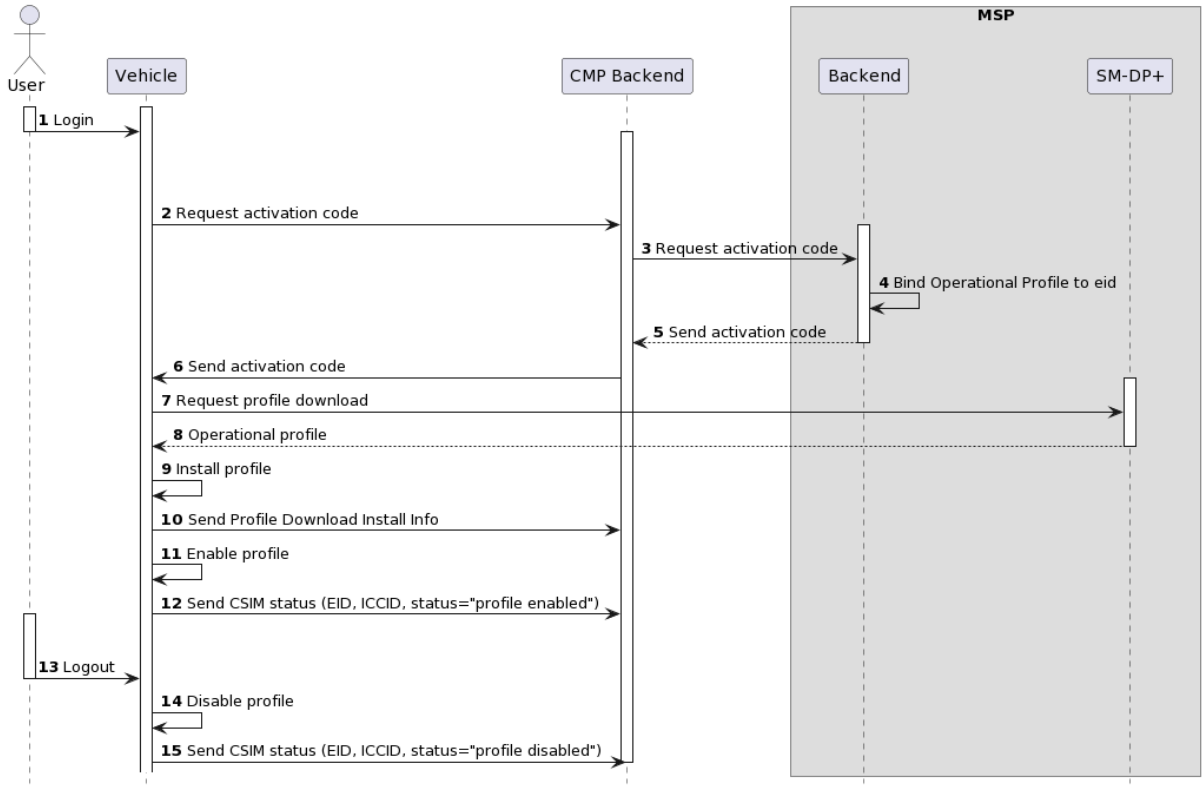


Figure 6: Car Provisioning Procedure

Start conditions:

a) The onboarding has been completed.
b) The user's vehicle is technically enabled for the AID service of the particular CMP.

Procedure:

[1] User logs into the vehicle.

[2] The vehicle determines that no Operational Profile is loaded and requests Activation Code from CMP backend.

[3] CMP backend requests Activation Code (*synonym 'download information'*) for the particular user and EID of the vehicle from the MSP. The user is identified by the 'MSP token'.

[4] The MSP binds an Operational Profile to the EID which was sent in the request. After binding the OP to the EID the profile can only be downloaded to the targeted vehicle. The Activation Code can be sent in clear text as it cannot be used anymore for any fraudulent purposes.

[5] MSP sends Activation Code to the CMP backend.

[6] CMP backend sends Activation Code to the vehicle. Only the asynchronous mode is shown in Figure 6.

[7-9] Download and installation of the Operational Profile from the MSP SM-DP+ server to the vehicle according to SGP.21/22

[10] A notification about the installation of the OP is sent to the CMP backend. This notification is also sent when the download or installation was not successful, including a respective error code. Optionally, the CMP backend can also notify the MSP backend.

[11] In case the user is still logged into the vehicle the Operational Profile will be enabled and the user can enjoy the MSPs services.

[12] With the message Send CSIM status the CMP backend is notified that the OP is successfully enabled.

[13-14] As soon as the user logs out of the vehicle (either actively or by parking the vehicle) the Operational Profile on the eUICC will be disabled.

[15] With the message Send CSIM status the CMP backend is notified that the OP is successfully disabled.

End conditions:

a) The vehicle is provisioned with the AID service for the particular user.
b) The AID service is active in the vehicle when the user is logged into the vehicle with their CMP account credentials.
c) The AID service is not active in the vehicle when the user is not logged into the vehicle.

4.3.1 Profile Swap

CMP will request a new AC (Activation Code), when the customer gets into a vehicle which has no Operational Profile for the customer yet, but the onboarding is performed and a valid MSP token is available in the CMP backend. Then the Operational Profile will be downloaded into the vehicle in which the customer has logged in, using the AC. Example: If the customer uses three different vehicles, three Operational Profiles (one in each vehicle) will be downloaded.

The customer’s subscription MAY have a limited number of Operational Profiles to be used concurrently with their mobile subscription. When the customer has for example ten Operational Profiles for concurrent use, the customer could download a maximum number of ten Operational Profiles on ten devices and have all of them installed. When reaching the limit, the customer experience would change, because before reaching the limit, the customer gets Operational Profiles for each new vehicle in which the customer logs in and after reaching the limit, the customer would not get any new Operational Profile seamlessly.

In order to deal with this unsatisfactory user experience when reaching the limit, the profile swap is introduced which makes the request of new profiles seamless, even when reaching the maximum number of Operational Profiles in a given subscription.

Using the profile swapping principle, the customer gets new Operational Profiles seamlessly on each vehicle the customer logs in even when the customer's subscription has a maximum number of Operational Profiles.

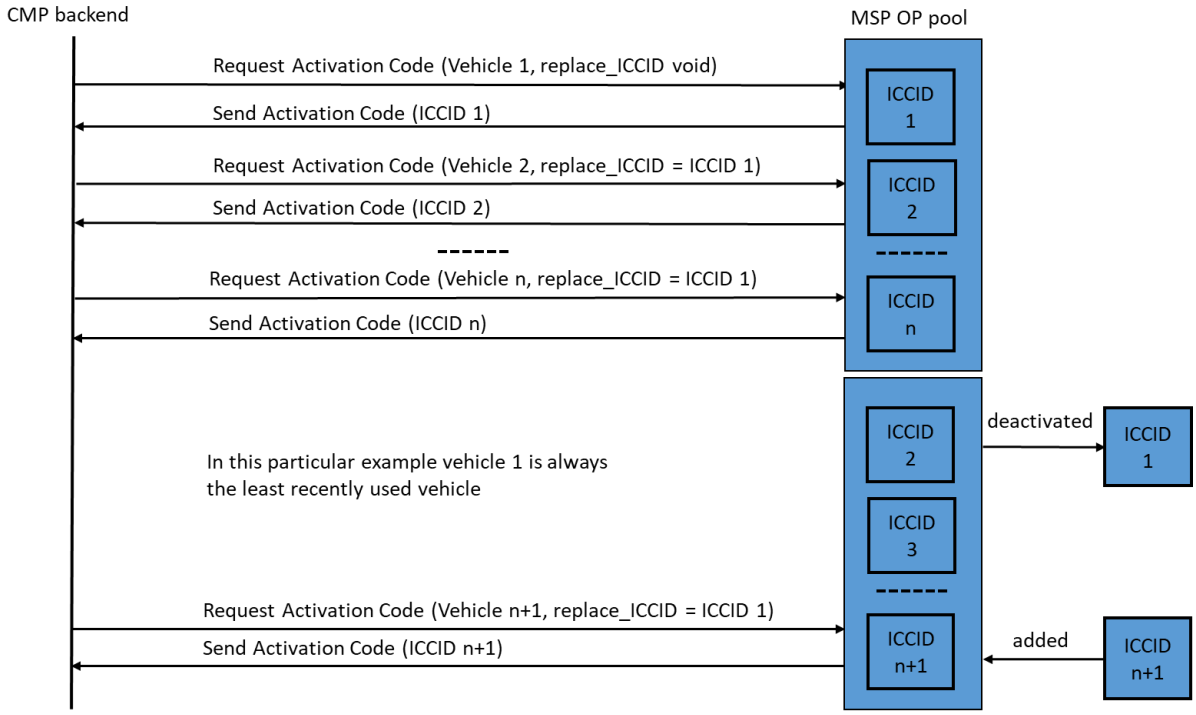


Figure 7 Process for Profile Swap

- Each MSP provides a limited number of Operational Profiles per account. The number n (a pool of n OPs) can vary from MSP to MSP and needs to be minimum 1.
- The user can download the n OPs into n vehicles. Once the user enters vehicle n+1 one OP in the pool of n OPs needs to be swapped or replaced.
- In order to give an indication to MSP which profile should be replaced, the CMP indicates the least recently used profile in a parameter "replace_ICCID".
- The number n might not be known to the CMP. Therefore the replace_ICCID is offered already with the request for the second profile.

The profile swap works as following:

1. A parameter indicating a candidate OP for replacement is sent when CMP uses the API to request an AC: CMP will send an ICCID which belongs to a profile which was downloaded for the customer in another vehicle. When no profile has

been downloaded for the customer yet, the parameter will not be sent from CMP. Following table shows how the ICCID is selected by CMP:

Scenario	Candidate OP for replacement
Customer gets into first vehicle and no Operational Profile has been downloaded for the customer in any vehicle (e.g. after initial onboarding) or all previously installed profiles are deleted.	Parameter will not be sent
Customer gets into second vehicle, meaning there is one other vehicle on which an Operational Profile is downloaded for the customer.	CMP will send the ICCID of the other vehicle (first vehicle) on which a profile is downloaded for the customer
Customer has downloaded profiles in multiple vehicles	CMP will determine an ICCID of a profile which is in one of the vehicles which have already a downloaded profile and send this ICCID. For example, this could be the ICCID of least recently used profile.

2. When the CMP sends a candidate OP for replacement in the AC request, the MSP SHALL react the following way:

    a. The MSP checks if the customer has reached the maximum number of Operational profiles for the subscription. The maximum number means, the AC request from CMP can not be answered with an AC, because the customer cannot download any more Operational Profiles.

    b. If the customer has NOT reached the maximum number of Operational Profiles for the subscription, the MSP SHALL return an AC and indicate, that the profile was not swapped. The indication, that the MSP did not swap the Operational Profile is done by the MSP setting a respective parameter to null.

    When the MSP indicates that the profile has not been swapped, the CMP will initiate the profile download in the vehicle for which the AC was requested.

    c. If the customer has reached the maximum number of Operational Profiles for the subscription, the MSP SHALL also return an AC and indicate, that the profile was swapped. This means, the MSP has removed an ICCID from the customer's subscription and replaced this old profile with the profile which belongs to the newly issued AC.The CMP indicates via replaceIccid parameter which OP can be swapped

from CMPs point of view (e.g. the least recently used OP), but the MSP SHALL decide which particular OP will be swapped. . The indication, that the MSP swapped an Operational Profile is done by setting the respective parameter (“swappedIccid”). When the CMP receives this indication together with the new AC, the CMP will delete the Operational Profile which was specified by the MSP from the vehicle on which it was downloaded.

NOTE: It cannot be guaranteed, that the swapped profile will be actually deleted from the vehicle, because the vehicle with the swapped profile might be unavailable for a long period of time (several weeks or longer) or the telematic device with the swapped profile might be exchanged in a workshop and will be forever unavailable.

To make the profile swap mechanism robust, following requirements SHALL be fulfilled:

- The MSP SHALL only consider the parameter indicating a candidate OP for replacement, if a profile swap is necessary.
- In case no profile swap is necessary, the MSP SHALL ignore the content of the parameter, meaning the CMP is not required to send this attribute.
- In case no profile swap is necessary, the content or the existence of the parameter SHALL NOT lead to any API errors and the MSP SHALL send a new AC to the CMP in any case.

Further points to consider:

- The CMP will only suggest profiles as candidate OPs for replacement which were not yet deleted from the vehicle.
- The profile swap is no mechanism to inform the MSP that a profile was deleted, because deleted profiles are not considered for replacement.
- CMP will use the ICCIDs which the MSP provides with the AC for Operational Profiles which were downloaded and for profiles which were not downloaded, e.g. in case the vehicle indicated to CMP backend a profile download error. This means that the MSP SHALL accept replace ICCIDs regardless if the Operational Profile was downloaded or if the profile download attempt failed.

NOTE: The profile swap is only considered as a fallback mechanism and shall not be used each time the customer changes the vehicle. The preferred way to support seamless connectivity in multiple vehicles is to have multiple concurrently used Operational Profiles in the customer’s mobile subscription. For each new vehicle, a new Operational Profile should be downloaded and these profiles should permanently stay in the vehicle until the customer deletes the profile explicitly.

The reliability of the service is increased by using multiple Operational Profiles instead of the profile swap, because the profile swap involves multiple steps on CMP and MSP side which increases the error likeliness.

Therefore, the profile swap MAY only be used, when a request for a new profile exceeding the number of profiles agreed upon is submitted.

4.4 Withdraw Onboarding

4.4.1 Withdraw Onboarding via CMP Touchpoint

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

actor "User"

participant "CMP user touchpoint" as touchpoint1

participant "Vehicle" as vehicle

participant "CMP Backend" as backend

box "MSP"
     participant "Backend" as MSP
     participant "MSP user touchpoint" as touchpoint2
     participant "SM-DP+"
end box

activate "User"
activate touchpoint1
activate vehicle

"User" -> touchpoint1 : Login
"User" -> touchpoint1 : Withdraw onboarding request


activate backend
touchpoint1 -> backend : Withdraw onboarding request (CMP account)
touchpoint1 -> backend : Get withdraw redirect URL (msp id)

activate MSP
backend -> MSP : Request invalidate MSP token (MSP token)
deactivate MSP

activate touchpoint2
     group Withdraw redirect option
     touchpoint1 -> touchpoint2 : Redirect to MSP touchpoint
     deactivate touchpoint1
     "User" -> touchpoint2: Cancel Subscription
     end
deactivate "User"
deactivate touchpoint2


loop For each profile associated with the MSP token
     backend -> vehicle : Request delete profile (EID, account_id, ICCID)
     deactivate backend
     |||
     vehicle -> vehicle : Delete profile
     activate "SM-DP+"
     group if configured in profile
```

```
            |||
            vehicle -> "SM-DP+": Send delete notification
            activate backend
            |||
      end
      vehicle -> backend : Send CSIM status (EID, ICCID, status="profile
deleted",...)
      deactivate "SM-DP+"
      activate MSP
      |||

      backend -> MSP: Send CSIM status (EID, ICCID, status="profile deleted")
      deactivate vehicle
      |||
end

backend -> backend: Remove MSP token

MSP -> MSP: Remove MSP token

@enduml
```

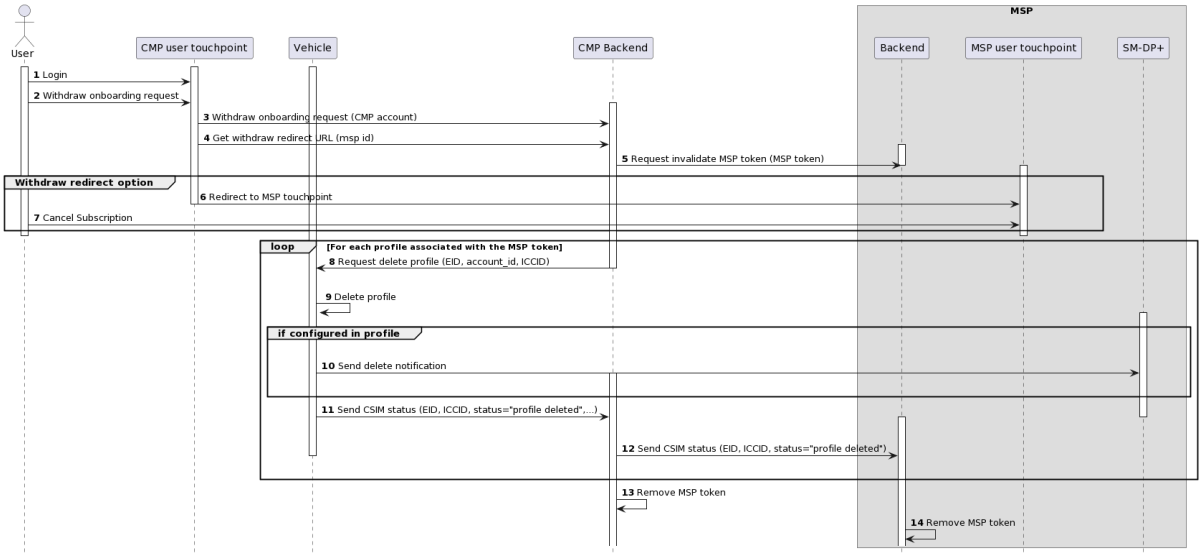


Figure 8 Withdraw Onboarding via CMP Touchpoint

Start conditions:

a) User is onboarded to the AID service

Procedure:

[1] User logs into CMP touchpoint.

[2] User wants to end the AID service and requests withdraw onboarding.

[3] CMP touchpoint sends withdraw onboarding to CMP backend.

[4] CMP touchpoint gets URL to redirect the user to MSP touchpoint.

[5] The CMP backend sends ‘invalidate MSP token’ to the MSP. With this command, the account federation between MSP and CMP will be released. The contractual relationship with the MSP is not impacted though. The MSP token SHALL NOT be used again to request a new OP. However, the MSP token SHALL NOT be removed from the system as it still is used in [12] “Send CSIM Status” when the remaining profiles are deleted.

[6] Optional: User is redirected to the MSP touchpoint.

[7] User cancels their MSP subscription via the MSP touchpoint. (Withdraw onboarding on the CMP touchpoint does not influence the contractual relationship of the user with the MSP. The MSP service can only be cancelled directly with the MSP.)

[8] CMP backend sends a delete order for the Operational Profile to each individual vehicle which has an Operational Profile downloaded.

[9] The Operational Profile on each individual vehicle is deleted.

[10] Each vehicle sends delete notification back to the MSPs SM-DP+ platform according to SGP.21/.22.

[11] Each vehicle sends a status update to the CMP backend indicating that the Operational Profile is deleted on the CMP side.[12] For each individual deleted Operational Profile a status update message is sent to the MSP backend.

[13] Once all Operational Profiles are deleted on all vehicles, the MSP token MAY be deleted on the CMP side.

[14] Once all Operational Profiles are deleted on all vehicles, the MSP token SHALL be invalidated on the MSP side.

End conditions:

a) The AID service is not available anymore for the respective user.
b) The MSP is notified about the service cancellation by the user and terminates the service according to contractual conditions.
c) Any remaining Operational Profile on vehicles will be deleted by the CMP once the vehicles become connected.

4.4.2 Withdraw Onboarding via MSP Touchpoint

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20


participant "Vehicle" as vehicle

participant "CMP Backend" as backend
```

```
box "MSP"
     participant "Backend" as MSP
     participant "SM-DP+"
end box

activate MSP
activate backend
MSP -> backend : Notify MSP token invalid (MSP token)
backend --> MSP : ok
deactivate MSP

activate vehicle
loop For each profile associated with the MSP token
     backend -> vehicle : Request delete profile (EID, account_id, ICCID)
     |||
     activate "SM-DP+"
     vehicle -> vehicle : Delete profile
     group if configured in profile
          |||
          vehicle -> "SM-DP+": Send delete notification
          |||
     end
     vehicle -> backend : Send CSIM status (EID, ICCID, status="profile
deleted",...)
     deactivate "SM-DP+"
     activate MSP
     |||

     backend -> MSP: Send CSIM status (EID, ICCID, status="profile deleted")
     deactivate vehicle
     |||
end

backend -> backend: Remove MSP token
MSP -> MSP: Remove MSP token

@enduml
```

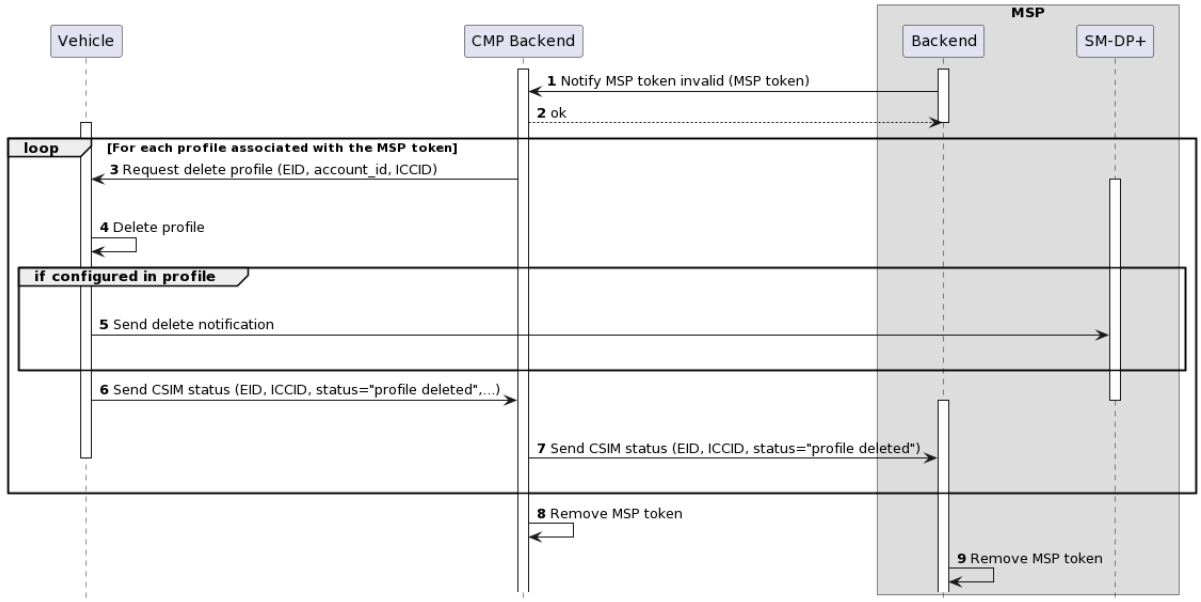


Figure 9 Withdraw Onboarding via MSP Touchpoint

Start conditions:

a) User is onboarded to the AID service

Procedure:

[1-2] The MSP backend instructs the CMP backend to invalidate the ‘MSP token’. The CMP backend acknowledges this message. The MSP token SHALL NOT be used again to request a new OP. However, the MSP token SHALL NOT be removed from the system as it still is used in [7] “Send CSIM Status” when the remaining profiles are deleted.

[3] CMP backend sends a delete order for the Operational Profile to each vehicle which has an Operational Profile downloaded.

[4] The Operational Profile on each individual vehicle is deleted.

[5] Each vehicle sends delete notification back to the MSPs SM-DP+ platform according to SGP.21/.22.

[6] Each vehicle sends a status update to the CMP backend indicating that the Operational Profile is deleted.

[7] For each individual deleted Operational Profile, a status update message is sent to the MSP backend.

[8] Once all Operational Profiles are deleted on all vehicles, the MSP token MAY be deleted in the CMP backend.

[9] Once all Operational Profiles are deleted on all vehicles, the MSP token MAY be deleted in the MSP backend.

NOTE: For OP deletion, it cannot be guaranteed, that the all OP will be actually deleted from the vehicles, because some vehicles might be unavailable for a long period of time (several weeks or longer) or the telematic device with the swapped profile might be exchanged in a workshop and will be forever unavailable.

End conditions:

a) The AID service is not available anymore for the respective user.
b) The MSP terminates the service according to contractual conditions.
c) Any remaining Operational Profile on vehicles will be deleted by the CMP once the vehicles become connected.

4.5 Delete Operational Profile

```
activate portal


User -> portal : Login

User -> portal : Delete profile
activate backend
portal -> backend : Delete profile \n(VIN, EID, account_id, ICCID)
deactivate User
```

```
backend -> vehicle: Request delete profile (EID, account_id, ICCID)
|||

     deactivate portal
     activate vehicle
     deactivate backend
     vehicle -> vehicle: Delete profile
     group if configured in profile
          activate "SM-DP+"
          vehicle -> "SM-DP+": Send delete notification
     end
deactivate "SM-DP+"

|||
activate backend
vehicle -> backend: Send CSIM status (EID, ICCID, status="profile
deleted/disabled",...)


group If configured
          backend -> MSP : Send CSIM status (EID, ICCID, status="profile
deleted/disabled")
     end
deactivate vehicle
deactivate backend
deactivate MSP

@enduml
```

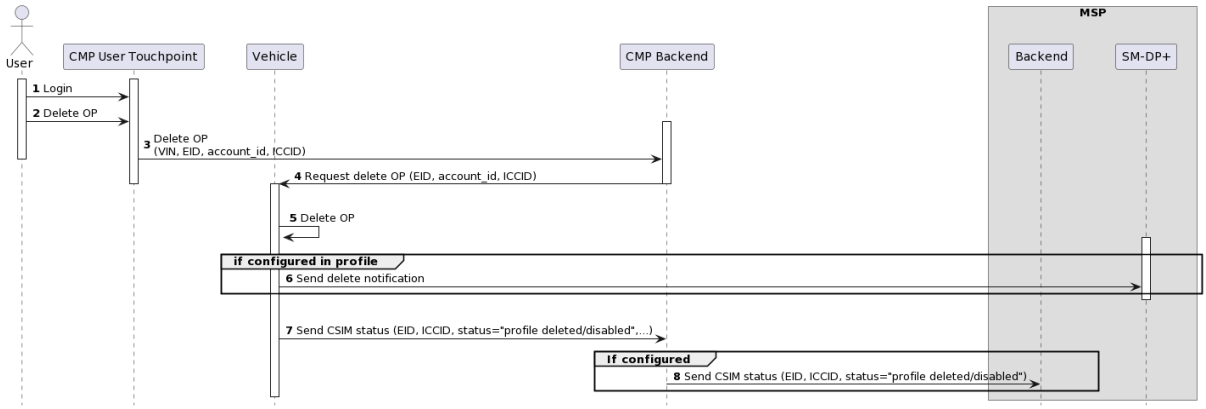


Figure 10 Delete Operational Profile

Start conditions:

a) The user is onboarded to the AID service.
b) The user has as least one vehicle provisioned for the AID service.

Procedure:

[1] User logs into the CMP touchpoint.

[2] User requests to delete a specific Operational Profile on a specific vehicle. The touchpoint displays the vehicles which have profiles downloaded and allows the user to select one of these vehicles.

[3] A delete Operational Profile request is sent to the CMP backend, including the vehicle identification.

[4] A delete Operational Profile request is sent from the CMP backend to the respective vehicle.

[5] If the Operational Profile is enabled, the Profile SHALL be disabled, and then, the Operational Profile is deleted from the eUICC on the vehicle.

[6] A delete notification is sent to the SM-DP+ platform of the MSP. The notification is defined by GSMA SGP.22 RSP Technical Specification.

[7] A delete notification is sent to the CMP backend.

[8] Optionally, another delete notification is sent from CMP backend to MSP backend.

End conditions:

a) The Operational Profile is deleted from the vehicle.
b) The CMP backend is notified about the Operational Profile deletion.
c) The MSP backend is notified about the Operational Profile deletion.
d) The MSP SM-DP+ platform is notified about the Operational Profile deletion.

4.6 Profile Status Synchronization

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

participant "CMP Backend" as backend

box "MSP"
     participant "Backend" as MSP
     end box

activate MSP
activate backend
backend -> MSP : Request Profile Status (MSP token)
MSP --> backend : Profiles ([{iccid, status}...])
backend -> backend : Synch Profiles
deactivate MSP

@enduml
```

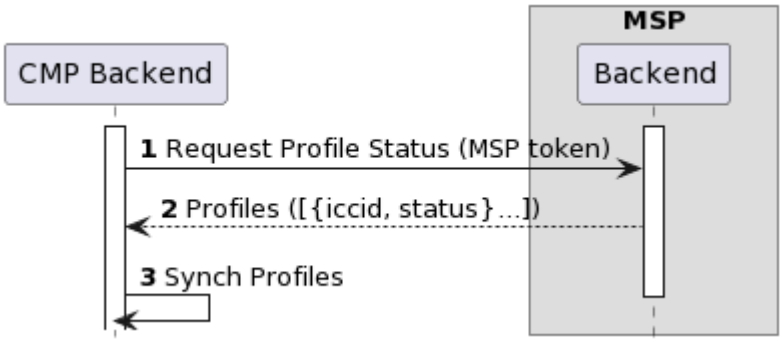


Figure 11 Profie Status Synchronization

Start conditions:

a) The user is onboarded for the AID service
b) There are OPs downloaded to vehicles for the particular user

Procedure:

[1] This interface is called by the CMP to request the status of all Operational Profiles or a specific Operational Profile that are not deleted for a user account on MSP side.

Using a specific ICCID in the parameters can be added to get the status of a single Operational Profile.

[2] MSP backend provides the status of the Operational Profile(s).

[3] The CMP backend updates the status for the Operational Profiles according to the status returned from the MSP in order to resolve potential inconsistencies which might have resulted from process errors.

End conditions:

a) The status of all OPs are synchronized between the MSP and the CMP.

4.7 Send Profile Information

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20


participant "CMP Backend" as backend
box "MSP"
     participant "Backend" as MSP
end box

activate backend
activate MSP
MSP -> backend: Send Profile info ([{iccid, status}…])
|||
|||
deactivate backend
```

```
deactivate MSP

@enduml
```

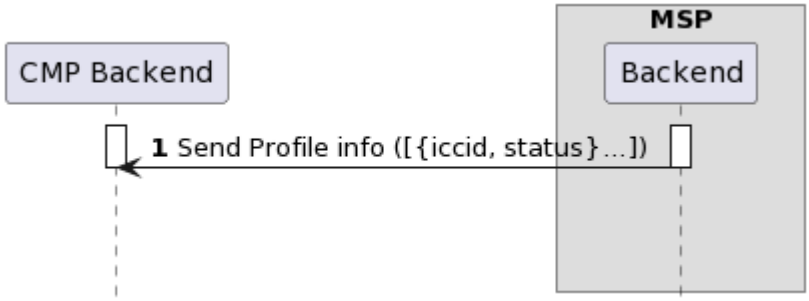


Figure 12 Send Profile Information

Start conditions:

a) Onboarding is completed (msp_token exists and is known to CMP)
b) At least one OP exists

Procedure:

[1] MSP sends the profile information to the CMP. Profile information can be valid, invalid, or suspended.

End conditions:

a) CMP then has the information that an OP does not provide service. In case the OP is in status “invalid” the OP has to be deleted by the CMP. In case the profile is “suspended” it is to the discretion of the CMP how to block the service usage and/or inform the user.

4.8 Update Subscription Info

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20


participant "CMP Backend" as backend
box "MSP"
     participant "Backend" as MSP
end box

activate backend
activate MSP
MSP -> backend: Update subscription info (phone-number, subscription type,
customerGroup)
|||
|||
deactivate backend
deactivate MSP

@enduml
```

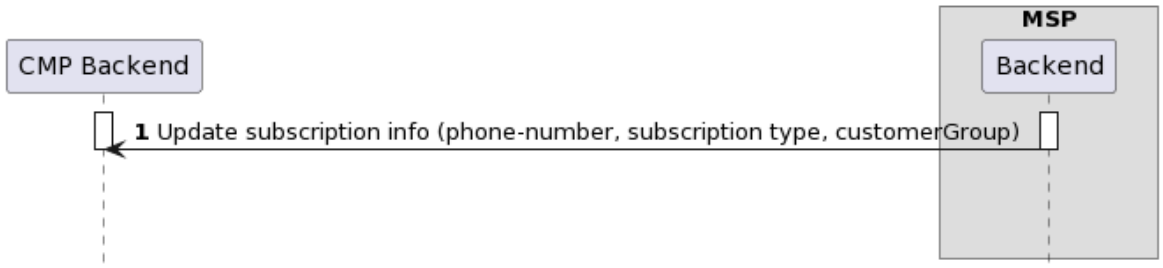


Figure 13 Update Subsription Info

Start conditions:

a) Onboarding is completed (msp_token exists and is known to CMP)

Procedure:

[1] MSP sends an update to the following information for a user's subscription: phoneNumber, subscriptionType, customerGroup

End conditions:

a) The subscription data relevant for the CMP is updated on the CMP side

4.9 Enable and Disable Profile

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

actor "User"

participant "Vehicle" as vehicle

participant "CMP Backend" as backend



activate "User"
activate vehicle
User -> vehicle : Login
vehicle -> vehicle : Enable Profile
activate backend
alt Enable successful
     vehicle -> backend : Send CSIM status (EID, ICCID, status="profile enabled")
else Enable failure
     vehicle -> backend : Send Enable Profile Error
end
|||
|||
User -> vehicle: Logout
vehicle -> vehicle: Disable profile
alt Disable successful
     vehicle -> backend : Send CSIM status (EID, ICCID, status="profile disabled")
else Disable failure
     vehicle -> backend : Send Disable Profile Error
end
|||
```

```
deactivate User
deactivate vehicle
deactivate backend
@enduml
```

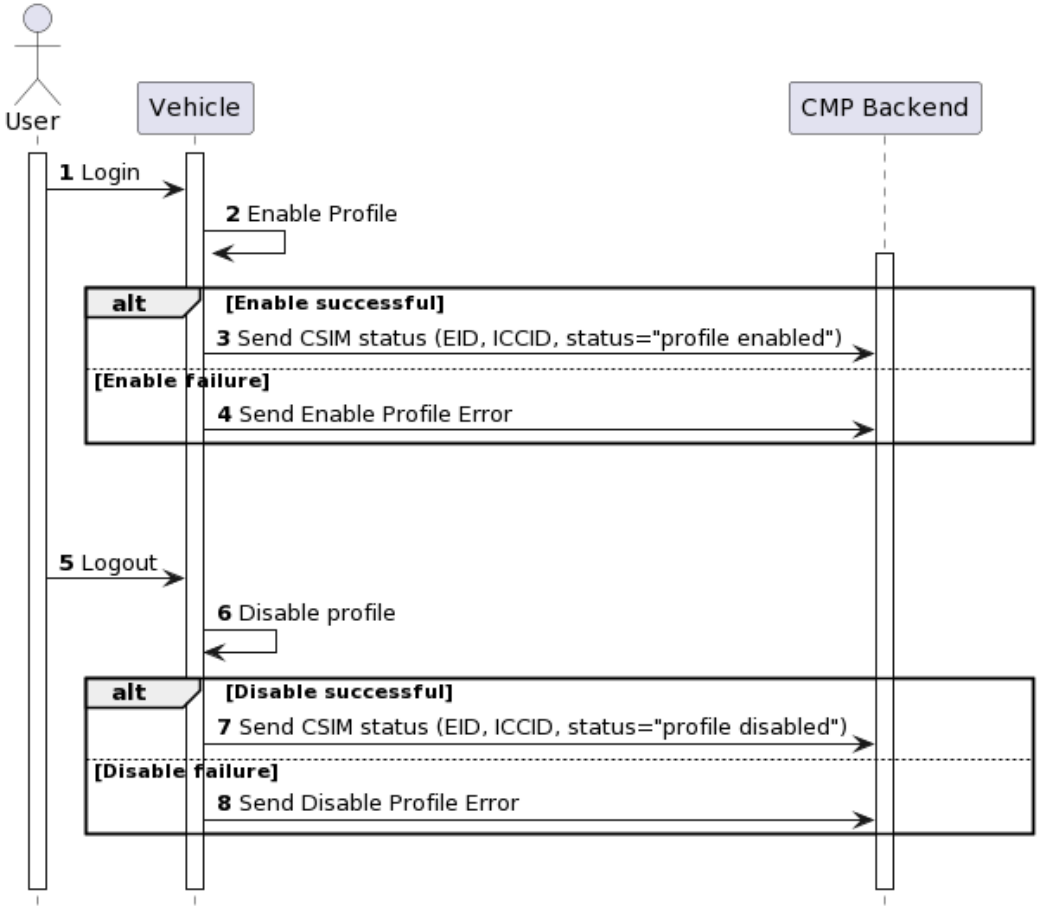


Figure 11 Enable and Disable Profile

Start conditions:

b) The user has an active CMP account
c) Car is provisioned for AiD service and an Operational Profile is available for the particular user on the eUICC

Procedure:

[1] User logs into the vehicle.

[2] After the login the Operational Profile of this particular User is enabled on the eUICC

[3] When the enable is successful, the CMP backend is informed via the message Send CSIM status about the new status of the profile.

[4] In case the enabling of the Operational Profile is not successful, a respective error message is sent to the CMP backend. Otherwise, if the enable is successful, the user is able to enjoy the AID service.

[5] At some point in time, typically at the end the vehicle usage, the user logs out of the vehicle. Very often the logout is triggered by locking the vehicle doors.

[6] Immediately after logout, the respective Operational Profile is disabled on the eUICC.

[7] When the disable is successful, the CMP backend is informed via the message Send CSIM status about the new status of the profile.

[8] In case the disabling of the Operational Profile is not successful, a respective error message is sent to the CMP backend.

End conditions:

d) Without any error, the user is logged out of the vehicle and their Operational Profile is disabled
e) In case the able did not work, the backend is informed about failure
f) In case the disable did not work, the backend is also informed about the failure

4.1 Request CSIM Info

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

participant "Vehicle" as vehicle

participant "CMP Backend" as backend

activate vehicle
activate backend
backend -> vehicle : Request CSIM info
||30||
vehicle -> backend : Send CSIM status
||30||
@enduml
```

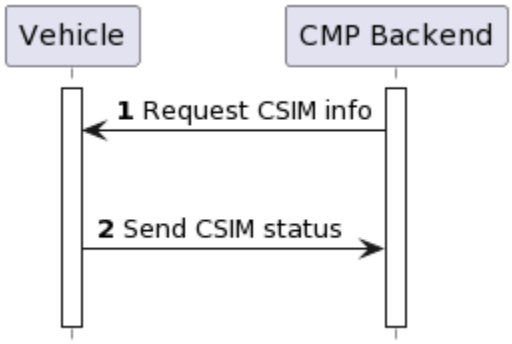


Figure 12 Request CSIM Info

Start conditions:

g) The vehicle is generally enabled for AiD services.

Procedure:

[1] In case there is a need to synchronize CSIM status between backend and vehicle, e.g. after an error condition indicating wrong backend information, the CMP backend will send a Request CSIM Info message ot the vehicle.

[2] The vehicle will respond with Send CSIM status to the CMP backend.

End conditions:

h) The CMP backend has the status information of the Profiles currently on the eUICC available.

4.2 Delete all profiles

```
@startuml

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20

participant "Vehicle" as vehicle

participant "CMP Backend" as backend

activate vehicle
activate backend
backend -> vehicle : Request delete all profiles
```

```
||30||
vehicle -> vehicle : delete all profiles
vehicle -> backend : Send CSIM status
||30||
@enduml
```

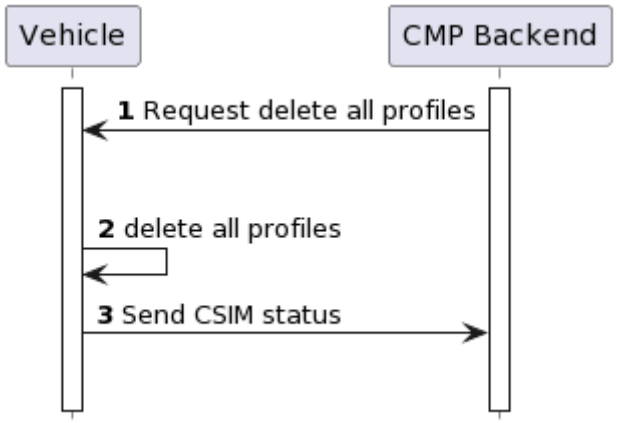


Figure 13 Delete all profiles

Start conditions:

i) The vehicle is generally enabled for AiD services.

Procedure:

[1] The CMP backend requests all Operational Profiles to be deleted on the vehicle

[2] All Operational Profiles will be deleted from the eUICC.

[3] The deletion of all profiles is reported back to the CMP backend

NOTE: The connectivy to report back to the CMP backend is out of scope and device is specific.

End conditions:

a) All Operational Profiles are deleted from the vehicle

5 Data Elements

5.1 Data Elements for AID1 Interface

5.1.1 HTTP Headers

Field	Value Description	Mandatory/ Optional		Schema	Value Desc
CMP-request-id	UUID identifying a request.	Mandatory		uuid	14abc1e2-3 9e71128cae
correlation-id	UUID identifying a transaction	Mandatory		uuid	31715660-0 6f7ad4c269
api-key	Used by service to decide which resources and procedures the client may access.	Mandatory if API key based authentication is done, if different authentication mechanism(s) is used, this field shall be omitted.		String, ([A-Za-z0-9])	1234abcdef
client-id	For interfaces called by the MSP: This parameter identifies the MSP at the CMP. For interfaces called by the CMP: This parameter identifies the CMP at the MSP. The value is agreed between CMP and MSP and is constant for each request.	Mandatory		String, ([A-Za-z0-9])	dk3kdwkef1
target-id	For interfaces called by the MSP: This parameter identifies the CMP which is called. For interfaces called by the CMP: This parameter identifies the MSP which is called. The value is agreed between CMP and MSP and is constant for each request.	Mandatory		String, ([A-Za-z0-9])	dk3kdwkef1

Table 5 HTTP Header Data Element

5.1.2 Account ID

Field	Value Description	Schema	Example
account_id	Account ID which must be used in “Send MSP token”.	String, ([A-Za-z0-9._-])	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOiIxNTgwNzI0MDAwIiwidmVyIjoiNzg5Iiwic2lkIjoiYTYyNTIxMWEtMmNiYS00MTM2LWFhOGItMzRjMzkwYjhhN2Q3Iiwia WF0IjoxNTgwNzIwNDAwfQ.ZbD3EEYryEp4yub_8Ca_PYi22TatV2N3TBuf2nNy5Bs

Table 6 Account ID Data Element

Since the ‘account id’ is added as URL parameter in the redirect URL to the MSP onboarding page (see chapter 4.1 Onboarding via CMP Touchpoint (e.g. app)), it must be ensured that the ‘account id’ was not modified by an untrusted third party before it is received by the MSP and that the MSP is able to verify the authenticity. This can be achieved e.g. by using a JSON web token (JWT) format for the ‘account id’.

Parameter	Example	Description

header	{
	"alg": "HS256",
	"typ": "JWT"
	}
		Header describing the token type and algorithm to verify the token signature

payload	{
	"exp": 1648541816,
	"ver": "6789",
	"sid": "a625211a-2cba-4136-aa8b-34c390b8a7d7",
	"iat": 1648541816
	}
		exp: Expiration date of the token in Unix time (i.e. seconds past 1970-01-01 00:00:00Z)
		ver: Secret knowledge to verify by the MSP, e.g. the last four digits of the user’s phone number. (optional)
		sid: Unique session ID (UUID) to identify the onboarding request.
		iat: Time when the token was created in Unix time (i.e. seconds past 1970-01-01 00:00:00Z).

Parameter	Example	Description
signature	HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), msp-specific-secret)	The signature of the JWT. The msp-specific-secret has to be exchanged between CMP and MSP beforehand once in a secure way. The same msp-specific-secret will then be used for every account id creation which is forwarded to the specific MSP.

The signature for the JWT is then created in the following way:

signature =

HMACSHA256(base64urlEncoding(header) + "." + base64urlEncoding(payload), msp-specific-secret)

The account id (as JWT) is then built in following format:

account id = base64urlEncoding(header) + '.' + base64urlEncoding(payload) + '.' + base64urlEncoding(signature)

The signature of the JWT SHALL be checked by the MSP then to ensure that the JWT was not altered. Additionally, the MSP checks the validity (via the expiration date in the exp field) and the secret knowledge (e.g. the user's phone number in the ver field). The secret knowledge SHALL be provided by the user when the account id is requested from the CMP.

NOTE**:** Since the account id is transmitted to the MSP in a URL parameter, it shall be ensured that only characters are contained in the account id which are URL safe. Therefore base64urlEncoding is used instead of regular base64 which additionally excludes the characters '+', '/' and '='. A definition of base64urlEncoding can be found in: https://datatracker.ietf.org/doc/html/rfc4648#section-5

5.1.3 MSP Token

Field	Value Description	Schema	Example
msp_token	Federated ID generated by the MSP to be used by both the MSP and the CMP to refer to the user. The msp_token is created by the MSP. It must be created in a way that it is unique for a customer for a given MSP. This means, if the CMP sends for example an activation code request to the MSP, there must	uuid	25bca1e2-338f-11d6-ac61-9e71138fd521

Field	Value Description	Schema	Example
	not be any ambiguity at the MSP.		

Table 7 MSP Token Data Element

5.1.4 ICCID

Field	Value Description	Schema	Example
Integrated Circuit Card ID, unique profile identifier.	String, 19-22 digits	98341201501601380129	Integrated Circuit Card ID, unique profile identifier.

Table 8 ICCID Data Element

5.1.5 Error Codes

In case an error occurs (HTTP status code not 2xx) with any API call, the called party SHALL reply with a response body containing the following fields:

Name	Description	Schema	Required	Example
code	Error code (see table below)	String	Only in case of error	10
error	Error description (see table below)	String	Only in case of error	The Account ID was not found

Table 9 Error Codes Response Body

The error descriptions may also be used in the error field of functions “Send MSP token” and “Send Activation Code”, e.g. by concatenating the error code and the description, e.g. “19: Other account ID error.”

Following error codes and descriptions can occur:

Error code	Description
10	The account ID was not found

Error code	Description
11	The account ID was found but is no longer valid
12	The account ID has an invalid signature
19	Other account ID error
20	The msp_token was not found
21	The msp_token was found but is no longer valid
23	The msp_token is already assigned to a user
24	The user already has a msp_token
29	Other msp_token id error
30	The specified request ID was not found
31	The specified Request ID was found but is no longer valid
39	Other request ID error
44	Other format error of activation code
47	The activation code has an invalid format
48	The msp_token already has an activation code assigned to it
49	Other activation code error
50	Profile type unknown
51	The specified profile type is unsupported for this request
59	Other profile type error
60	The request body is empty or contains null values
61	The profile with the ICCID %s could not be found
62	The profile with the ICCID %s is not linked to the MSP token with the federated_id %s
63	The profile with the ICCID %s is invalid and may not return to a valid state
64	None of the profiles could be processed
65	The msp_token identified by the federated_id or the profile(s) identified by the ICCID could not be found
90	Internal Server Error
91	Invalid request syntax
92	The correlation-id does not match
93	The MSP with the MSP identifier %s could not be found
94	The user identified by the account id could not be found
100	Unspecified CMP content error
101	User login not possible on CMP side
102	Onboarding aborted by user
103	CMP onboarding content currently unavailabe

Error code	Description
…	
200 to 300	Reserved for proprietary error codes

Table 10 error codes

The above metioned error codes can also be used as error messages in redirect procedures.

5.2 Data Elements for AID2 Interface

5.2.1 RequestID

Field	Value Description	Schema	Example
VehicleID	Vehicle Identification Number according to ISO3779:2009	String, ([A-Za-z0-9._-]), 17 digits	WBAUX11030A334483
Timestamp	Unix timestamp when the event was started (first method-call/message in a related sequence). If the method-call / message is directly related to a previous message then the same (old) timestamp shall used	32-bit integer	1 700 000 000 equals Nov. 14, 2023

5.2.2 CmpUserId

Field	Value Description	Schema	Example
CmpUserID	Unique customer Id within the CMP systems	String	9876543210

6 General API Requirements

The following API requirements should be applied:

- Following content type has to be used for the APIs: application/json
- An API key SHALL be used for each request to “CMP CESIM MSP API”. For “CMP CESIM MSP API”, the CMP will provide the API key to the MSP. For “MSP CESIM CMP API”, the MSP provides a username and password to the CMP for basic authentication or provides an API key. The MSP must specify if an API key or basic authentication SHALL be used for “MSP Interfaces”. The MSP provides the username and password or API key to the CMP.
- Two way TLS (mutual authentication) SHALL be used for each request.
- For “CMP CESIM MSP API” (MSP calls APIs provided by the CMP), the following procedure will be used:
    - CMP provides the server certificate, which must be trusted by the MSP
    - The MSP provides the client certificate to CMP, so CMP can add the MSP’s client certificate to CMP’s list of trusted clients
- For “MSP CESIM CMP API” (CMP calls APIs provided at MSP), the following procedure will be used:
    - The MSP provides the server certificate, which must be trusted by the CMP
    - CMP provides the client certificate to the MSP, so the MSP can add the CMP’s client certificate to the MSP’s list of trusted clients.
- All communication between CMP backend and MSP backend, and all communication between touchpoints and backends (e.g. from MSP touchpoint to MSP backend), and all other communication related to the AID service must be encrypted.

7 Functions

7.1 Functions for AiD1 Interface

The functions below are running over AID2 as described in Figure 3.

In Annex B there is Json Code available which allows the implementation of the functions described in this chapter. In case the json code differs from the description in this chapter the json code is the valid specification.

In Annex D there is Protobuf Code (Google Protocol Buffers) available which allows the implementation of the functions described in this chapter based on MQTT protocol. In case the Protobuf Code differs from the description in this chapter the protobuf code is the valid specification.

7.1.1 Redirect to MSP Content

Related procedure: Onboarding via CMP touchpoint

Description:

CMP will use following redirect from CMP touchpoint to MSP touchpoint:

https://consumer-esim-msp.com/portal/login?id=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOiIxNTgwNzI0MDAwIiwidmVyIjoiNzg
5Iiwic2lkIjoiYTYyNTIxMWEtMmNiYS00MTM2LWFhOGItMzRjMzkwYjhhN2Q3IiwiaWF0Ij
oxNTgwNzIwNDAwfQ.ZbD3EEYryEp4yub_8Ca_PYi22TatV2N3TBuf2nny5Bs&redirect_
uri= https%3A%2F%2Fconsumer-esim-cmp.com

“consumer-esim-msp.com/portal/login” is an example for the URL of the MSP’s touchpoint. The MSP will provide this information to the CMP for pre-production and production environment of the MSP.

Data elements:

- account_id (as defined in section 5.1.2)

Additional input data:

Field	Value Description	Mandatory/ Optional	Schema	Example
redirect_uri	URL back to a CMP touchpoint when onboarding finished. Important: For security reasons, the domains to which the MSP performs redirects must be agreed and specified and the MSP is only allowed to redirect to one of the agreed domains. Example 1: Agreed redirects to: *.cmp.de and redirect_uri = https://www.cmp.de/esim --> Redirect is allowed. Example 2: Agreed redirects to: *.cmp.com and redirect_uri = https://www.example.de/esim --> Redirect is NOT allowed.	Mandatory	URI (encoded)	https%3A%2F%2Fconsumer-esim-cmp.com

7.1.2 Redirect to CMP content

Related procedure: Onboarding via MSP touchpoint

Description:

MSP will use following redirect from MSP touchpoint to CMP touchpoint:

https://consumer-esim-cmp.com/portal/login? redirect_uri=https%3A%2F%2Fconsumer-esim-msp.com%2Freturn%3Ffoo%3Dbar

“consumer-esim-cmp.com/portal/login” is an example for the URL of the CMP’s touchpoint. The CMP will provide this information to the MSP for pre-production and production environment of the CMP.

Additional input data:

Field	Value Description	Mandatory/ Optional	Schema	Example
redirect_uri	This field contains the URL to connect back to the MSP touchpoint. The URL MAY contain parameters in the path, e.g. status information used by the MSP to keep a transaction. For security reasons, the domain or custom url scheme to which the CMP performs redirects SHALL be agreed and specified and the CMP SHALL be only allowed to redirect to one of the agreed domains. Example 1: Agreed redirects to: *.msp.de and redirect_uri = https://www.msp.de/esim --> Redirect is allowed. Example 2: Agreed redirects to: *.msp.de and redirect_uri = https://www.example.de/esim --> Redirect is NOT allowed.	Mandatory	URI (encoded)	https%3A%2F%2Fconsumer-esim-msp.com

7.1.3 Redirect back to MSP content

Related procedure: Onboarding via MSP touchpoint

Description:

CMP will reconnect back to the MSP touchpoint after completion of the CMP Content procedure and the creation of the account_id. The CMP SHALL use the MSP URL provided by the MSP via redirect_uri parameter defined in section 7.2 and append data fields as outlined below.

7.1.3.1 Sucessful Completion

If the CMP Content procedure yields an account id, then the following redirect back must be used:

```
https://consumer-esim-cmp.com/return?foo=bar&id=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOiIxNTgwNzI0MDAwIiwidmVyIjoiNzg
5Iiwic2lkIjoiYTYyNTIxMWEtMmNiYS00MTM2LWFhOGItMzRjMzkwYjhhN2Q3IiwiaWF0Ij
oxNTgwNzIwNDAwfQ.ZbD3EEYryEp4yub_8Ca_PYi22TatV2N3TBuf2nny5Bs
```

Data elements:

- account_id (as defined in section 5.1.2)

7.1.3.2 Unsucessful Completion

If the CMP Content procedure does no yield an account id, then the following redirect SHALL be used:

```
https://consumer-esim.com/return?foo=bar&code=100&error=”Unspecified CMP content
error”
```

Data elements:

- code
- error

(see chapter 5.5. error codes)

7.1.4 Send MSP Token

Related procedure: Onboarding via CMP touchpoint

Description:

```
POST /cesim/msp/v1/users/{account_id}
```

This interface is called by the MSP to submit an msp_token to the CMP. The msp_token for a customer cannot be updated.

When sending a new msp_token from MSP to CMP, the CMP will check if the account_id is valid. Only if the account_id is valid, the CMP will accept the msp_token. Otherwise, the CMP will return HTTP status 422 and reject the msp_token.

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- account_id (as defined in section 5.1.2)

Request body:

- msp_token (as defined in section 5.1.3)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
phoneNumber	The MSISDN associated with the user's subscription. This must be the number which is known to the customer (not a pure technical number), because this number will be used at CMP's touchpoints to show the customer which subscriptions are added to the customer's account. The phone number must be sent encrypted and Base64 encoded (see below). Format of unencrypted phone number: CC + NPA + SN CC = Country Code (No leading "+" sign) NPA = Number Planning Area SN = Subscriber Number Examples: 919961345678 4918974020143 Following content must be sent:	Mandatory Set value to null if the error parameter is not null in the request body	String	b3YsYkQqi5xw1 DxcfaRC1PANh ElCHXNvj

Name	Description	Mandatory /Optional	Schema	Example
	Base64(encrypt_function(CC + NPA + SN)) encrypt_function will be agreed separately by MSP and CMP.			
subscriptionType	Indicates whether the msp_token is linked to a private mobile subscription or business mobile subscription. This information will be shown to the customer at a CMP touchpoint so the customer can easily identify private and business subscriptions	Mandatory Set value to null if the error parameter is not null in the request body	String ("private"|"business"|"unknown")	private
error	Not set, or error describing why the msp_token cannot be generated. If error parameter is not null all other parameters are ignored.	Mandatory Set value to null if msp_token is sent within the request body	String	1000:Customer eligible Maximum length 512 characters.

Name	Description	Mandatory /Optional	Schema	Example
customerGroup	An optional identifier which can be used to group customers. This field is intended for reporting use cases. The value for this field is agreed between MSP and CMP if required.	optional	String	Market_Germar
iv	Base64 String of the initialization vector, used for phone number encryption (see below)	Mandatory Set value to null if the error parameter is not null in the request body	String	N1RnMnHhudF

The “phoneNumber” and “subscriptionType” will be used to display these informations to the customer, so the customer can differentiate between several subscriptions, if the customer has onboarded multiple subscriptions, e.g. a private and a business contract.

Phone number encryption:

For security reasons, the phoneNumber must not be sent as plain text in the field “phoneNumber”, but must be encrypted. This is additional to the TLS encryption of the message itself, meaning, that the message must be encrypted using TLS and additionally, the phone number must be encrypted in the field “phoneNumber”.

Details on encryption:

The phoneNumber will be encrypted in the MSP backend. The details about the type of encryption and algorithm will be agreed separately by MSP and CMP.

Response:

HTTP Status Code:

HTTP-Code	Description
201	The msp_token has been created and assigned to the appropriate user.
401	Unauthorized
403	The client does not have the necessary permissions to add an msp_token to the user
404	The user identified by account_id could not be found
422	msp_token format is incorrect, or the msp_token is duplicate or the account_id is invalid or the phone number cannot be decrypted

HTTP-Code	Description
500	Internal Server Error

7.1.5 Notify MSP Token Invalid

Related procedure: Withdraw Onboarding via MSP touchpoint

Description:

POST /cesim/msp/v1/users/{msp_token}/invalidate

This interface is called by the MSP to notify the CMP that the user's subscription has ended or the user chose to delete the link between CMP account and MSP account. The 'msp_token' cannot be used to request any new ACs and all OPs of the user will be deleted as a consequence.

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
reason	An optional short description why the 'msp_token' is invalid. The reason is logged for support purposes in case customer contacts customer care.	Optional	String	user subscription ended

Response:

HTTP Status Code:

HTTP-Code	Description
204	The 'msp_token' has been invalidated
401	Unauthorized
404	The user identified by the 'msp_token' cannot be found
500	Internal Server Error

7.1.6 Send Profile Information

Related procedure: Send profile information

Description:

POST /cesim/msp/v1/users/{msp_token}/profiles

This interface is called by the MSP to send information which is relevant for one specific profile or a list of specified profiles, which all belong to the same user and same subscription (same MSP token).

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
Profiles object	The list of affected ICCIDs of the customer.	Mandatory	Array of profile object	“profiles”:[{ “iccid”: “894450080517203 953”, “status”: “suspended”, “reason”: “contract was suspended” }, { “iccid”: “894450080517203 954”, “status”: “invalid”

Name	Description	Mandatory /Optional	Schema	Example
				“reason”: “profile defect” }
status	The new status of the profiles.	Mandatory	String ("invalid"|"valid"|"suspended")	invalid
reason	An optional short description why the profile(s) information was sent. The reason is logged for support purposes in case customer contacts customer care.	Mandatory Set value to null if no reason is given	String	profile defect

Response:

HTTP Status Code:

HTTP-Code	Description
200	The profile information has been updated successfully.
401	Unauthorized
403	The client does not have the necessary permissions to add information to the specified user or profile
404	The user identified by the ‘msp_token’ or the profile identified by the ICCID cannot be found
422	Invalid request body
500	Internal Server Error

Information on usage of “Send profile information”:

This API shall be used when information on a specific OP, or list of OPs, needs to be sent from MSP to CMP. For example, if one profile is not working due to a defect, the MSP can inform CMP to delete this specific profile. If the MSP needs to send information for several profiles, e.g. if two profiles have to be deleted, the MSP can specify several ICCIDs in an array, however all specified profiles must belong to the same user (same ‘msp_token’). The request “Send profile information” shall not be sent together with “Notify MSP token invalid”, because in this case, the CMP automatically considers all profiles which are linked to this ‘msp_token’ as invalid and will delete these profiles.

The request “Send profile information” shall also not be sent, when the MSP sends an AC to the CMP with swappedIccid={iccid} to indicate that a profile swap was done, because in this case, the CMP automatically considers the profile which was specified in MSP’s request in swappedIccid as invalid and will delete this profile.

Profile status information:

Status	Description
invalid	Profile definitely in a state in which it cannot be used for voice or data and will never return to “valid” state →Profile deletion necessary.
suspended	Profile temporary in a state in which it cannot be used for voice or data. Profile might return to state “valid” →Considered as an information at CMP side
valid	Profile can be used normally again when it was in state “suspended” before.

The profile status “invalid” SHALL only be sent from MSP to CMP when the OP is in a state in which the customer cannot use telephony/data and the OP state will never return to a state in which the customer can use telephony/data via this profile again. When the CMP receives an “invalid” status, the CMP will delete the profile from the vehicle. Examples when OP status “invalid” can be sent from MSP to CMP are:

- OP has a technical defect and needs to be replaced with a new OP
- OP was removed from user’s subscription and user will never be able to use the OP in the future

The profile state “suspended” & “valid” will trigger no direct action on the CMP side, e.g. no OP deletion will be executed, but the information will be logged for support cases, e.g. when the customer calls the CMPs customer care. The profile status “suspended” and “valid” SHALL NOT be sent during normal OP usage, e.g. when the OP gets enabled or disabled. These profile state SHALL only be sent, when the MSP blocks the usage of the OP for a limited time. Block usage means, that the user will not be able to use telephony/data after login at the vehicle because the MSP set the OP in a state which prohibits telephony/data services. Example use cases are:

- Customer did not pay the MSP’s bill and therefore, the MSP decides to block the telephony/data usage of the customer, until the bills get payed
- Customer loses main phone and calls MSP’s customer care and MSP blocks the telephony/data usage of all OPs which are associated with the customer’s contract for security reasons until the customer calls the MSP to release the usage blocker.

The CMP will use the profile state “suspended” & “valid” in the following example support scenario: The customer calls the CMPs customer care stating that some service in the vehicle which is using is not working. The CMP will then check, if a profile status “suspended” was sent. If yes, the CMP can explain to the customer, that the service is not working because the OP is currently in a state in which telephony/data cannot be used.

7.1.7 Update Subscription Info

Related procedure: Update subscription info

Description:

PATCH /cesim/msp/v1/subscriptions/{msp_token}

This interface is called by the MSP to update the following information for a user's subscription:

- phoneNumber
- subscriptionType
- customerGroup

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
phoneNumber	See description in "Send MSP token". The phone number must be sent encrypted.	Mandatory	See Schema in "Send MSP token".	See Example in "Send MSP token".
subscriptionType	See description in "Send MSP token".	Mandatory	See Schema in "Send MSP token".	See Example in "Send MSP token".
customerGroup	See description in "Send MSP token".	Optional	See Schema in "Send MSP token".	See Example in "Send MSP token".
iv	Base64 String of the initialization vector for phone number decryption.	Mandatory	String	N1RnMnHhudFLgi1

Response:

HTTP Status Code:

HTTP-Code	Description
201	The phone number, subscription type and customer group have been updated successfully for the user
401	Unauthorized
403	The client does not have the necessary permissions to update the subscription
404	The user identified by msp_token could not be found
422	msp_token format is incorrect
500	Internal Server Error

7.1.8 Request Activation Code

Related procedure: Car provisioning

Description:

POST /v1/activation-code-requests/{msp_token}

The CMP calls this interface to request an AC for a specific user. The AC will then be sent to a vehicle where it will be used to download an OP.

The MSP can send back the AC either in the response body (synchronous mode) or with the API Send Activation Code (asynchronous mode). Both modes will be described in detail below.

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
profileType	The type of the profile. Currently, there is only the profile type “personal”.	Mandatory	String, ([ASCII] {20})	personal

Name	Description	Mandatory /Optional	Schema	Example
deviceType	A type identifier for the eSIM device in the vehicle.	Mandatory	String, ([ASCII] {20})	tcu
imei	International Mobile Equipment Identity. This parameter will be empty but present for compatibility reasons.	Optional	String, 15 digis	
eid	Embedded Integrated Circuit Card ID.	Mandatory	String, 32 digits	
activationCodeRequestID	A UUID identifying the request for a new activation code.	Mandatory Value must not be set to null	uuid	14abc1e2-338f-11e6-ac61-9e71128cae77
replaceIccid	Integrated Circuit Card ID, unique profile identifier. The profile which belongs to this iccid is the replace candidate, when all eSIM profile slots of the customers are already used.	Mandatory Set value to null if no replace ICCID can be given	String, 19-22 digits or null	98341201501601380129

Response:

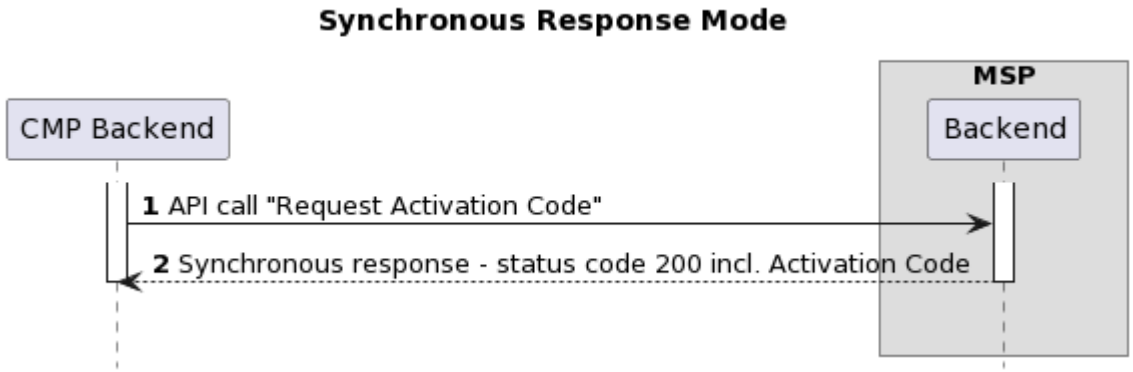
HTTP Status Code:

HTTP-Code	Description
200	Success. Activation code is included in response body
201	The request has been stored and will be processed
401	Unauthorized
404	The user identified by the msp_token or the request could not be found
422	No activation code can be created
423	Invalid eid
500	Internal Server Error

When the CMP uses “Request activation code”, the MSP has two options on how to respond: In a synchronous or in an asynchronous response mode.

7.1.8.1 Synchronous Response Mode

```
 @startuml

title Synchronous Response Mode

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20


participant "CMP Backend" as backend
box "MSP"
     participant "Backend" as MSP
end box

activate backend
activate MSP
backend -> MSP: API call "Request Activation Code"
MSP --> backend: Synchronous response - status code 200 incl. Activation Code
|||
deactivate backend
deactivate MSP

@enduml
```



Successful response (activation code generated / available):

The MSP uses http status code 200 in “Request activation code” response and includes the activation code and profile type in the response body as described in chapter 7.6.1.1 below.

Unsuccessful response (activation code cannot be generated):

The MSP uses a 4xx status code and includes an error description in response body (as described in error handling).

The synchronous response mode is only allowed if the activation code is already available at the MSP’s system and can be returned within 3 seconds. This means especially, that no more user interaction is necessary. If any user interaction is necessary, the asynchronous response mode must be used.

7.1.8.1.1 Sychronous Response Body

Input data (response body):

- Iccid (as defined in section 5.1.4)

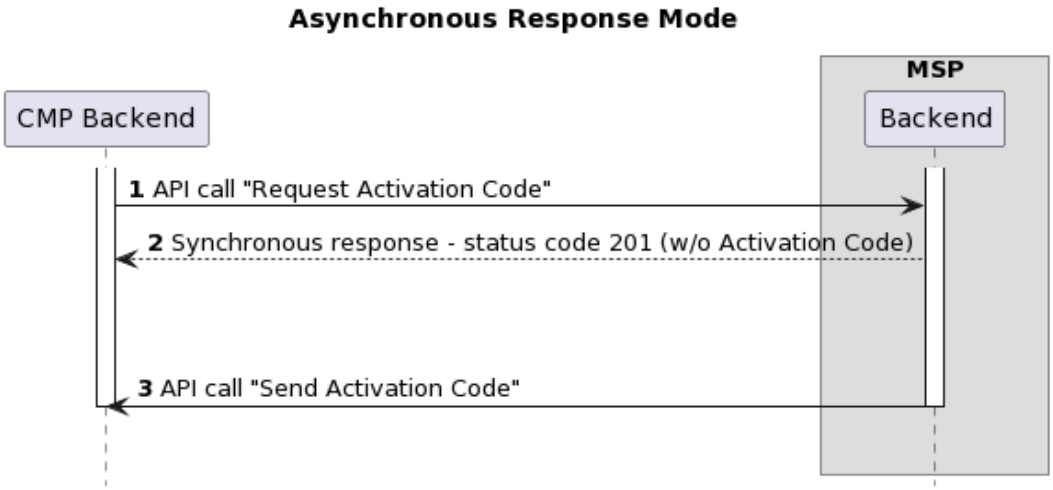
Additional input data (response body):

Name	Description	Mandatory /Optional	Schema	Example
activationCode	The activation code which is used to download an OP from the SM-DP+ server.	Mandatory	String	1$VR-01-43-CUST.SC.MY-ESIM.COM$DEF30A27E3CEFD34FA24B4A38D9681A5
profileType	The type of the profile.	Mandatory	String	personal
swappedIccid	Integrated Circuit Card ID, unique profile identifier of the profile which the MSP deactivated to be able to send out the activation code for the new profile. In case no profile swap was done, the value is set to null.	Mandatory Set value to null if no profile was swapped	String, 19-22 digits or null	98341201501601380129

7.1.8.2 Asynchronous Response Mode

```
@startuml

title Asynchronous Response Mode

autonumber
hide footbox
skinparam BoxPadding 10
skinparam ParticipantPadding 20


participant "CMP Backend" as backend
box "MSP"
     participant "Backend" as MSP
end box

activate backend
activate MSP
backend -> MSP: API call "Request Activation Code"
MSP --> backend: Synchronous response - status code 201 (w/o Activation Code)
|||
|||
MSP -> backend: API call "Send Activation Code"
|||
deactivate backend
deactivate MSP
```

```
@enduml
```



Successful response:

The MSP uses status code 201 indicating that the CMP's request will be processed. When the activation code is available, the MSP uses the CMP's API "Send activation code" to deliver the activation code to the CMP (see chapter 7.6.2.1 below).

Sequence details:

a) CMP sends "Request activation code" to MSP
b) MSP makes the synchronous response to "Request activation code" with status code 201
c) MSP performs necessary steps in MSP backend to prepare a valid activation code (profile needs to be downloadable)
d) MSP sends activation code to CMP using "Send activation code".

Note: Step b is a precondition of step d. The MSP cannot time out the CMP's request "Request activation code" and then send the activation code to the CMP using "Send activation code" or send the activation code to the CMP with "Send activation code" BEFORE making the synchronous response to "Request activation code".

Unsuccessful response:

The MSP uses status code 201 indicating that the CMP's request will be processed. If an error occurs during processing, the MSP uses the CMP's API "Send activation code" to deliver the error description to the CMP.

Sequence details:

Same sequence as in "Successful response" applies.

7.1.8.2.1 Asynchronous Response: Send Activation Code

Related procedure: Car provisioning

Description:

POST /cesim/msp/v1/activation-codes/{msp_token}

The MSP calls this interface to asynchronously send an AC for a specific user which was requested by the CMP via Request Activation Code before. The AC will then be sent to a vehicle where it will be used to download an OP.

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Input data (request body):

- iccid (as defined in section 5.1.4)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
activationCode	The activation code which is used to download an OP from the SM-DP+ server.	Mandatory Set value to null if the error parameter is not null in the request body	String	1$VR-01-43-CUST.SC.MY-ESIM.COM$DEF30A27E3CEFD34FA24B4A38D9681A5
profileType	The type of the profile.	Mandatory Set value to null if the error parameter is null in the request body	String	personal
activationCodeRequestID	UUID, it is the same activationCodeRequestID which	Mandatory	uuid	14abc1e2-338f-11e6-ac61-9e71128cae77

Name	Description	Mandatory /Optional	Schema	Example
	was received in Request Activation Code			
swappedIccid	Integrated Circuit Card ID, unique profile identifier of the profile which the MSP deactivated to be able to send out the activation code for the new profile. In case no profile swap was done, the value is set to null.	Mandatory Set value to null if no profile was swapped or if error parameter is not null	String, 19-22 digits or null	98341201501601380129
error	Only set if activation code could not be generated. Contains the description why the activation code could not be generated. See chapter 5.5 Error Codes	Mandatory Set value to null if msp_token is sent within the request body	String	1000:Customer not eligible

Response:

HTTP Status Code:

HTTP-Code	Description
200	Success. Activation code has been assigned to the appropriate user.
401	Unauthorized
404	The user identified by the msp_token could not be found.
422	The activationCodeRequestID could not be matched.
500	Internal Server Error

7.1.9 Send CSIM Status

Related procedure: Withdraw Onboarding

Description:

POST /v1/statuses/{msp_token}

This interface is called by the CMP to notify the MSP of OP status updates. In addition to the notifications set in the SGP.22, a backend-to-backend profile notification is also sent to MSPs. It is according to a pre-arrangement between MSP and CMP which status changes are exchanges (e.g.: whether enable or disable is sent).

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Input data (request body):

- iccid (as defined in section 5.1.4)

Additional input data (request body):

Name	Description	Mandatory /Optional	Schema	Example
eid	Embedded Integrated Circuit Card ID.	Mandatory	String, 32 digits	
status	A string describing the new status of the profile.	Mandatory	String, ("deleted"|"enabled"|"disabled"|"installed"|"installation_failed")	"deleted"

Response:

HTTP Status Code:

HTTP-Code	Description
201	The request has been stored and will be processed
401	Unauthorized
404	The user identified by the msp_token or the request could not be found
422	The request cannot be processed
500	Internal Server Error

7.1.10 Request Invalidate MSP Token

Related procedure: Withdraw onboarding

Description:

```
DELETE /v1/users/{msp_token}
```

This interface is called by the CMP to request the invalidation of a MSP token. This call happens after the user chooses to cancel the link between CMP account and MSP account. The MSP token cannot be used to request any new ACs any longer and all OPs of the user will be deleted as a consequence. Note: The MSP must deactivate all profiles immediately after receiving “Request invalidate MSP token” and not wait until CSIM status with profile deleted is sent from the CMP.

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token (as defined in section 5.1.3)

Response:

HTTP Status Code:

HTTP-Code	Description
204	The msp_token has been invalidated successfully
401	Unauthorized
404	User cannot be found
422	The request cannot be processed
500	Internal Server Error

7.1.11 Request Profile Status

Related procedure: Profile status synchronization

Description:

```
GET /v1/profiles/status/{msp_token}?iccid={iccid}
```

This interface is called by the CMP to request the status of all profiles or a specific profile that are not deleted for a federated id on MSP side. Using a specific iccid in the parameters can be added to get the status of a single iccid.

Data elements:

- HTTP header (as defined in section 5.1.1)

Path parameter:

- msp_token
- iccid (optional)

Response:

HTTP Status Code:

HTTP-Code	Description
200	Success. Profiles included in the response body
401	Unauthorized
404	User or specified profile cannot be found
422	The request cannot be processed
500	Internal Server Error

Response body (required if the http status code returned is 200):

Name	Description	Mandatory /Optional	Schema	Example
profiles	The profile(s) linked to the msp_token.	Mandatory	Array of Objects	{ "profiles": [{ "iccid": "28918293818491828371", "status": "enabled" }, { "iccid": "10918293818491828371", "status": "disabled" }

Name	Description	Mandatory /Optional	Schema	Example
				], }

Allowed profile states within the result:

Status	Explanation
enabled	The profile is installed and currently enabled. See also remark below
disabled	The profile is installed and currently disabled. See also remark below
installed	Default state if no information on profile enablement is available
prepared	In case a profile was already assigned to a user but was not downloaded yet

Remark on enable/disable:

Use only in case the current enable/disable state information is available on MSP side. This should not be the information which is sent from the CMP to the MSP via Send CSIM status, because this information will only be sent once per profile.

NOTE: The table above show the profile states which are not the same profile states as the one defined in SGP.22.

7.2 Functions for AiD2 Interface

The functions below are running over AID2 as described in Figure 3.

In order to achieve a high degree of interoperability, it is recommended to implement the functions of the AiD2 interface in MQTT (Message Queuing Telemetry Transport) as published by “https://mqtt.org”. However the functions described below are specified protocol independent, but based on a client/server-architecture.

7.2.1 Request Activation Code

Description:

The CMP app (vehicle) sends a message to the CMP backend to request an Activation Code for the download of an OP.

Data elements:

- RequestId (as defined in section 5.2.1)
- CmpUserId (as defined in section 5.2.2)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
imei	International Mobile Equipment Identity. This parameter can be empty but present for compatibility reasons.	Mandatory	String, 15 digits	
eid	Embedded Integrated Circuit Card ID.	Mandatory	String, 32 digits	

Response:

Error	Description
UNKNOWN_VIN	Vehicle Id is unknown by the backend system
UNKNOWN_EID	EID is unknown by the backend system or inconsitent
UNKNOWN_IMEI	IMEI is unknown by the backend system or inconsitent
UNKNOWN_USER	User is unknown by the backend system
NO_SUBSCRIPTION_FOUND	User in known but has no valid subscription
NO_ACTIVATION_CODE_AVAILABLE	Backend cannot get an AC from the MSP
TERMS_NOT_ACCEPTED	User has a valid subscription but has not agreed to the specific terms and conditions for the AiD service.

7.2.2 Send Activation Code

Description:

The CMP backend send the Activation Code it has received from the MSP to the vehicle CMP app.

Data elements:

- RequestId (as defined in section 5.2.1)
- CmpUserId (as defined in section 5.2.2)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
ProfileType	Any description of the profile in case different profile types are provided by the MSP	Mandatory	String	PERSONAL, DEFAULT, PRIVATE, BUSINESS
activationCode	The activation code which is used to download an OP from the SM-DP+ server.	Mandatory	String	1$VR-01-43-CUST.SC.MY-ESIM.COM$DEF30A27E3CEFD34FA24B4A38D9681A5

Response:

There is no specific response to this message as the success/failure of the download SHALL be reported in Send Profile Download Install Info message.

7.2.3 Send Profile Download Install Info

Description:

The CMP app (vehicle) sends a message to the CMP indicating the result of the download and install process of the Operational Profile.

Data elements:

- RequestId (as defined in section 5.2.1)
- CmpUserId (as defined in section 5.2.2)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
ICCID	Integrated Circuit Card ID, unique profile identifier.	Mandatory	String, 19-22 digits	9834120150160138012 9

Name	Description	Mandatory/ Optional	Schema	Example
eid	Embedded Integrated Circuit Card ID.	Mandatory	String, 32 digits	
activationCode	The activation code which was used to download the OP	Mandatory	String	1$VR-01-43-CUST.SC.MY-ESIM.COM$DEF30A27E3CEFD34FA24B4A38D9681A5
DownloadInstallResult	Result of the process for download and installation of an Operational Profile. Values: See table below.	Mandatory	String	SUCCESS
FunctionExecutionStatus	FunctionExecutionStatus as defined in the SGP.22 RSP Technical Specification [2], chapter 6.5.1.4, as JSON String	Mandatory Set value to empty string when DownloadInstallResult = SUCCESS	String	status : Failed, statusCodeData : { subjectCode : 8.2.5, reasonCode : 3.7, message : No more Profile
csimErrorCode	General error conditions of eUCC according to SW1 and SW2 values in GlobalPlatform Card Specification [3]	Mandatory Set value to 0 when no error code is available	UInt16 2 Byte code	27268 (= ‘6A’ ‘84’) – Not enough memory space

Possible values for “DownloadInstallResult”

Value	Description
PROFILE_INSTALLATION_SUCCESS	The Operational Profile was downloaded and installed successfully.
PROFILE_INSTALLATION_UNKNOWN_ERROR	Unknown error
PROFILE_DOWNLOAD_FAILED_NO_NETWORK_CONNECTION	Failed because of no network connectivity available

Value	Description
PROFILE_DOWNLOAD_FAILED_PROXY_COMMUNICATION_ERROR	Unable to establish IP connection via proxy. Only possible in case a IP proxy is used.
PROFILE_DOWNLOAD_FAILED_SMDP_COMMUNICATION_ERROR	Communication with SM-DP+ platform failed.
PROFILE_DOWNLOAD_FAILED_SMDP_DOWNLOAD_ERROR	Communication with SM-DP+ established but download of Operational Profile failed.
PROFILE_INSTALLATION_FAILED	Download of Operational Profile was successful but installation on eUICC failed.

7.2.4 Request Delete Profile

Related procedure: Delete Operational Profile

Description:

The CMP backend requests the CMP app (vehicle) to a Operational Profiles on the eUICC. The OP is identified by the ICCID.

Data elements:

- RequestId (as defined in section 5.2.1)
- CmpUserId (as defined in section 5.2.2)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
ICCID	Integrated Circuit Card ID, unique profile identifier.	Mandatory	String, 19-22 digits	98341201501601380129

Response:

Name	Description	Mandatory/ Optional	Schema	Example
RequestDeleteProfileResult	Indication whether the profile deletion was successful or not.	Mandatory	String	SUCCESS
csimErrorCode	General error conditions of eUCC according to SW1 and SW2 values in GlobalPlatform Card Specification [3]	Mandatory Set value to 0 when no error code is available	UInt16 2 Byte code	27268 (= ‘6A’ ‘84’) – Not enough memory space

Possible values for “RequestDeleteProfileResult”

Value	Description
DELETE_PROFILE_SUCCESS	The Operational Profile was successfully deleted.
DELETE_PROFILE_FAILED	Unknown error
DELETE_PROFILE_NOT_FOUND	No profile found with given ICCID
DELETE_PROFILE_USER_NOT_FOUND	No user found for the given CmpUserId

7.2.5 Send CSIM Status (vehicle to CMP backend)

Related procedure: Withdraw onboarding, Delete Operational Profile

Description:

The CMP app on the vehicle send the status of the CSIM to the CMP backend.

Data elements:

- RequestId (as defined in section 5.2.1)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
eid	Embedded Integrated Circuit Card ID.	Mandatory	String, 32 digits	
imei	International Mobile Equipment Identity. This parameter can be empty but present for compatibility reasons.	Mandatory	String,15 digits	
csimNetwork	Network name to which the currently Consumer eSIM is currently connected to. Purpose is to detect roaming errors.	Mandatory Set value to empty string when no value can be provided	String	OperatorExample Name
opsNetwork	Network name of the operational network which is used to connect to the CMP backend and to the MSP SM-DP+.	Mandatory Set value to empty string when no value can be provided	String	WLAN, OperatorExample Name, …
Profiles[]	Array of profiles	Mandatory	profile, profile, ..	

Format of the field “profile”:

Name	Description	Mandatory/ Optional	Schema	Example
ICCID	Integrated Circuit Card ID, unique profile identifier.	Mandatory	String, 19-22 digits	98341201501601380129
CmpUserId	Unique customer Id within the CMP systems	Mandatory	String	9876543210
ProfileStatus	Status of the individual operational profile. Values: ENABLED, DISABLED, DELETED	Mandatory	String	DISABLED
ProfileType	Any description of the profile in case different profile types are provided by the MSP	Mandatory	String	PERSONAL, DEFAULT, PRIVATE, BUSINESS

Name	Description	Mandatory/ Optional	Schema	Example
MCC	Mobile Country Code of the MSP which the OP belongs to	Mandatory Set value to empty string when no value can be provided	String	123 for OperatorExample Country
MNC	Mobile Network Code of the MSP which the OP belongs to	Mandatory Set value to empty string when no value can be provided	String	01 for OperatorExample Network

Additional requirements:

(1) If backend is not reachable because there is no mobile connectivity, the request SHALL not be dismissed by vehicle and SHALL be resent, when mobile connectivity is available again.

(2) If backend is not reachable because there is a backend outage, a retry mechanism SHALL be applied by vehicle. Retry strategy is subject to implementation.

7.2.6 Send Enable Profile Error

Related procedure: Enable and Disable Profile

Description:

In case the enabling of an Operational Profile fails, this message is sent from the vehicle to the backend.

Data elements:

- RequestId (as defined in section 5.2.1)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
profile	Please refer to chapter 7.2.5. for the format of the field “profile”	Mandatory		

Name	Description	Mandatory/ Optional	Schema	Example
csimErrorCode	General error conditions of eUCC according to SW1 and SW2 values in GlobalPlatform Card Specification Version 2.3.1 [3]	Mandatory Set value to 0 when no error code is available	UInt16 2 Byte code	27268 (= ‘6A’ ‘84’) – Not enough memory space

7.2.7 Send Disable Profile Error

Related procedure: Enable and Disable Profile

Description:

In case the disabling of an Operational Profile fails, this message is sent from the vehicle to the backend.

Data elements:

- RequestId (as defined in section 5.2.1)

Additional input data (request body):

Name	Description	Mandatory/ Optional	Schema	Example
profile	Please refer to chapter 7.2.5. for the format of the field “profile”	Mandatory		
csimErrorCode	General error conditions of eUCC according to SW1 and SW2 values in GlobalPlatform Card Specification Version 2.3.1 [3]	Mandatory Set value to 0 when no error code is available	UInt16 2 Byte code	27268 (= ‘6A’ ‘84’ – Not enough memory space

7.2.8 Request CSIM info

Related procedure: Request CSIM Info

Description:

The CMP backend requests the current status information of all Operational Profiles of the eUICC.

Data elements:

- RequestId (as defined in section 5.2.1)

Response:

Name	Description	Mandatory/ Optional	Schema	Example
RequestCSIMInfoResult	Information whether the eUICC has received the request and can process it.	Mandatory	String	CSIM_INFO_SUCCESS, CSIM_INFO_GENERAL_ERROR
csimErrorCode	General error conditions of eUCC according to SW1 and SW2 values in GlobalPlatform Card Specification [3]	Mandatory Set value to 0 when no error code is available	UInt16 2 Byte code	27268 (= ‘6A’ ‘84’) – Not enough memory space

Possible values for “RequestCSIMInfoResult”

Value	Description
SUCCESS	eUICC has received the request and can process it
GENERAL_CSIM_ERROR	General error reported by eUICC, the operation is not successfull

7.2.9 Request delete all profiles

Related procedure: Delete all profiles

Description:

The CMP backend requests all Operational Profiles to be deleted from the eUICC in the vehicle.

Data elements:

- RequestId (as defined in section 5.2.1)

Response:

Name	Description	Mandatory/ Optional	Schema	Example
RequestDeleteProfileResult	Information whether the Operational Profiles have been deleted or not	Mandatory	String	DELETE_ALL_PROFILES_SUCCESS, DELETE_ALL_PROFILES_FAILED, DELETE_ALL_PROFILES_NO_PROFILES_FOUND
csimErrorCode	General error conditions of eUCC according to SW1 and SW2 values in GlobalPlatform Card Specification [3]	Mandatory Set value to 0 when no error code is available	UInt16 2 Byte code	27268 (= ‘6A’ ‘84’) – Not enough memory space

Possible values for “RequestDeleteProfileResult”

Value	Description
SUCCESS	The Operational Profiles have been successfully deleted.
FAILED	Error reported by eUICC. csimErrorCode has to be provided
NO_PROFILES_FOUND	eUICC has been empty and therefore no pofiles have been deleted

Annex A Use Case description (Informative)

A.1 Onboarding via Touchpoint

The user enables their account for the use of the AID Service together with an MSP. During the onboarding process, the user prepares their mobile subscription contract for the use of the AID Service, e.g. by buying an appropriate multi-SIM option. When all preconditions for the use of the AID Service are fulfilled on MSP side, the MSP notifies the CMP. After the onboarding is completed successfully, the user will automatically get an Operational Profile to each vehicle, in which the customer logs in with their CMP account.

In this use case, the user uses a CMP touchpoint (e.g. CMP App) to initiate the process.

Pre-conditions: User has CMP account and MSP account with valid MSP contract.

End-conditions:

- User's mobile subscription is prepared for AID Service, i.e. the CMP backend can ask for new activation codes at MSP when the user logs in at a new vehicle so the user can automatically get an OP to the new vehicle.
- User's CMP account and user's MSP account are linked, so information about the user can be exchanged, e.g. to request new activation codes for the user.

A.2 Profile Download / Enablement

The user logs in at a new vehicle and there is no AUP on the vehicle for the user, yet. An activation code will be requested by CMP backend from MSP backend and then forwarded to the vehicle. The vehicle uses the activation code to download an Operational Profile from the MSP's SM-DP+.

After completing this use case, the user can use telephony and data via the new Operational Profile on the eUICC.

Pre-condition: Onboarding was completed successfully

End-condition: Operational Profile downloaded and installed on eUICC in vehicle and Operational Profile active for communication.

A.3 Withdraw Onboarding

During the onboarding process, a link between the user's CMP account and the user's MSP account is created (see above). There needs to be a way for the user to remove this link. After the user removes this link, no more activation codes will be requested by CMP backend from MSP backend. This means, when the user enters a new vehicle and logs in, no OP will be downloaded for this vehicle. Additionally, all existing OP which resulted from the onboarding process will be deleted. On the MSP side, all OP that were requested or downloaded during the validity of the MSP token, will be also deleted/reset by the MSP and MSP deactivates these OP from their network.

Pre-condition: Onboarding was completed successfully

End-condition: Link between MSP account and MSP account deleted and all OPs which originate from the relevant onboarding process are deleted. The user's account on CMP side and MSP side are cleaned and are in a state, as they were before the onboarding. If the user wants to use the AID Service again, a new onboarding process must be performed.

NOTE: If the user does an onboarding and then a withdraw onboarding and then an onboarding again, the CMP treats the first and second onboardings as completely separate processes and will not take over any information from the first onboarding to the second onboarding.

A.4 Delete an Operational Profile

The customer (or the CMP or the MSP) may want to delete a specific OP. This will only affect a specific OP and not all OPs as in "withdraw onboarding". The MSP SHALL make sure that after a deletion of the OP was triggered, the OP slot is released again so that a new OP can be downloaded.

There are situations where the CMP will trigger an OP delete. This includes:

- eUICC storage is full and other user wants to install an OP
- Customer uses a rental car and CMP becomes aware that the rental period ends
- Vehicle maintenance (e.g. vehicle maintenance in workshop)
- Exchange of electronic control unit in the vehicle

Pre-condition: Onboarding was completed successfully and at least one OP was downloaded.

End-condition: The OP which was specified by the user (or the CMP or the MSP) is deleted. A new OP can be downloaded for the related Operational Profile slot in case the customer logs in at the same or new vehicle.

A.5 Profile Status Synchronization

Due to errors in the process, the OP status on CMP side and MSP side can become inconsistent. The Profile status synchronization resolves this data inconsistencies by exchanging the current OP information which the MSP has stored. CMP will call the interface in support cases when inconsistencies are detected.

Pre-condition: Onboarding was completed.

End-condition: The current OPs of a user which belong to the CMP Consumer eSIM are provided from MSP to CMP.

Annex B JSON Code

B.1 REST APIs provided by CMP and called by MSP

{
  "openapi": "3.0.1",

```
 "info": {
  "title": "CMP CESIM MSP API",
  "description": "REST APIs provided by CMP and called by MSP.",
  "version": "1.0.0"
 },
 "servers": [
  {
   "url": "https"
  }
 ],
 "paths" : {
  "/cesim/msp/v1/activation-codes/{msp_token}": {
   "post": {
    "tags": [ "Send activation code" ],
    "summary": "Send Activation Code",
  "description": "The MSP calls this interface to asynchronously send an activation code for a specific user which was requested by the CMP via 'Request Activation Code' before. The activation code will then be sent to a vehicle where it will be used to download an operational profile.",
     "operationId": "submitActivationCode",
     "parameters": [
    {
        "name": "CMP-request-id",
        "in": "header",
        "description": "UUID identifying a request.",
        "required": true,
        "example": "d534080b-9794-4351-88e7-8af9e079ca1e",
        "schema": {
         "type": "string",
         "format": "uuid"
        }
       },
    {
        "name": "correlation-id",
        "in": "header",
        "description": "UUID identifying a transaction.",
        "required": true,
        "example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
```

```
        "schema": {
          "type": "string",
      "format": "uuid"
        }
       },
       {
        "name": "client-id",
        "in": "header",
        "description": "This parameter identifies the MSP at the CMP.",
        "required": true,
        "example": "dk3kdwkef1",
        "schema": {
          "type": "string"
        }
       }, {
        "name": "target-id",
        "in": "header",
        "description": "This parameter identifies the CMP which is called.",
        "required": true,
        "example": "gKl32ko4",
        "schema": {
          "type": "string"
        }
       },
    {
        "name": "msp_token",
        "in": "path",
        "required": true,
        "example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",
        "description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",
        "schema": {
          "type": "string",
      "format": "uuid"
        }
       }
  ],
```

"requestBody": {
"description": "Request body",
"content": {
"application/json": {
"schema": {
"$ref": *"#/components/schemas/SendActivationCodeBody"*
}
}
},
"required": true
},
"responses": {
"200": {
"description": "Success. The activation code has been assigned to the appropriate user."
},
"401": {
"description": "Unauthorized.",
"content": {
"application/json": {
"schema": {
"$ref": *"#/components/schemas/ApiCmpUnauthorizedErrorResponse"*
}
}
}
},
"404": {
"description": "The user identified by the msp_token could not be found.",
"content": {
"application/json": {
"schema": {
"$ref": *"#/components/schemas/ApiCmpTokenErrorResponse"*
}
}
}
},
"422": {
"description": "The activationCodeRequestID could not be matched.",

```
          "content": {
            "application/json": {
              "schema": {
                "$ref": "#/components/schemas/ApiCmpClientErrorResponse"
              }
            }
          }
        },
        "500": {
          "description": "Internal Server Error.",
          "content": {
            "application/json": {
              "schema": {
                "$ref": "#/components/schemas/ApiCmpServerErrorResponse"
              }
            }
          }
        }
      },
      "security": [ {
        "basicAuth": [ ]
      }, {
        "apiKey": [ ]
      } ]
    }
  },
  "/cesim/msp/v1/subscriptions/{msp_token}": {
    "patch": {
      "tags": [ "Update subscription info" ],
      "summary": "Update Subscription Info",
    "description": "This interface is called by the MSP to update the phoneNumber, the subscriptionType or the customerGroup for a user's subscription.",
      "operationId": "updateSubscriptionInfo",
      "parameters": [
    {
        "name": "CMP-request-id",
        "in": "header",
        "description": "UUID identifying a request.",
```

```
        "required": true,
        "example": "d534080b-9794-4351-88e7-8af9e079ca1e",
        "schema": {
          "type": "string",
          "format": "uuid"
        }
      },
    {
        "name": "correlation-id",
        "in": "header",
        "description": "UUID identifying a transaction.",
        "required": true,
        "example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
        "schema": {
          "type": "string",
     "format": "uuid"
        }
      },
      {
        "name": "client-id",
        "in": "header",
        "description": "This parameter identifies the MSP at the CMP.",
        "required": true,
        "example": "dk3kdwkef1",
        "schema": {
          "type": "string"
        }
      }, {
        "name": "target-id",
        "in": "header",
        "description": "This parameter identifies the CMP which is called.",
        "required": true,
        "example": "gKl32ko4",
        "schema": {
          "type": "string"
        }
      },
```

```
    {
        "name": "msp_token",
        "in": "path",
        "required": true,
        "example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",
        "description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",
        "schema": {
          "type": "string",
      "format": "uuid"
        }
      }
  ],
    "requestBody": {
      "description": "Request body",
      "content": {
        "application/json": {
          "schema": {
            "$ref": "#/components/schemas/UpdateSubscriptionInfoBody"
          }
        }
      },
      "required": true
    },
    "responses": {
      "201": {
        "description": "The phone number, subscription type and customer group have been updated successfully for the user."
      },
      "401": {
        "description": "Authentication information is missing or invalid.",
    "content": {
        "application/json": {
          "schema": {
            "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
          }
        }
```

```
        }
      },
      "403": {
        "description": "The client does not have the necessary permissions to update the subscription.",
    "content": {
        "application/json": {
          "schema": {
            "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
          }
        }
      }
      },
      "404": {
        "description": "The user identified by the msp_token could not be found.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpTokenErrorResponse"
            }
          }
        }
      },
      "422" : {
        "description": "The msp_token format is incorrect.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpClientErrorResponse"
            }
          }
        }
      },
      "500" : {
        "description": "Internal Server Error.",
        "content": {
          "application/json": {
            "schema": {
```

```
            "$ref": "#/components/schemas/ApiCmpServerErrorResponse"
          }
        }
      }
    }
  },
  "security": [ {
    "basicAuth": [ ]
  }, {
    "apiKey": [ ]
  } ]
 }
},
"/cesim/msp/v1/users/{account_id}": {
 "post": {
  "tags": [ "Send MSP token" ],
  "summary": "Send MSP Token",
  "description": "This interface is called by the MSP to submit an MSP token to the CMP. The MSP token for a customer cannot be updated. When sending a new MSP token from MSP to CMP, the CMP will check if the Account ID is valid. Only if the Account ID is valid, the CMP will accept the MSP Token. Otherwise, the CMP will return HTTP status 422 and reject the MSP Token.",
  "operationId": "sendMspToken",
  "parameters": [
   {
    "name": "CMP-request-id",
    "in": "header",
    "description": "UUID identifying a request.",
    "required": true,
    "example": "d534080b-9794-4351-88e7-8af9e079ca1e",
    "schema": {
     "type": "string",
     "format": "uuid"
    }
   },
   {
    "name": "correlation-id",
    "in": "header",
    "description": "UUID identifying a transaction.",
```

"required": true,
"example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "client-id",
"in": "header",
"description": "This parameter identifies the MSP at the CMP.",
"required": true,
"example": "dk3kdwkef1",
"schema": {
"type": "string"
}
},
{
"name": "target-id",
"in": "header",
"description": "This parameter identifies the CMP which is called.",
"required": true,
"example": "gKl32ko4",
"schema": {
"type": "string"
}
},
{
"name": "account_id",
"in": "path",
"required": true,
"example": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOiIxNTgwNzI0MDAwIiwidmVyIjoiNzg5Iiwic2lkIjoiYTYyNTIxMWEtMmNiYS00MTM2LWFhOGItMzRjMzkwYjhhN2Q3IiwiaWF0IjoxNTgwNzIwNDAwfQ.ZbD3EEYryEp4yub_8Ca_PYi22TatV2N3TBuf2nNy5Bs",
"schema": {
"type": "string"
}

```
    }
     ],
     "requestBody": {
       "description": "Request body",
       "content": {
         "application/json": {
           "schema": {
             "$ref": "#/components/schemas/SendMspTokenBody"
           }
         }
       },
       "required": true
     },
     "responses": {
       "201": {
        "description": "The msp_token has been created and assigned to the appropriate user."
       },
       "401": {
         "description": "Unauthorized.",
    "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
            }
          }
        }
       },
       "403": {
            "description": "The client does not have the necessary permissions to add an msp_token to the user.",
    "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
            }
          }
        }
       },
```

```
      "404": {
        "description": "The user identified by account_id could not be found.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpAccountIdErrorResponse"
            }
          }
        }
      },
      "422": {
        "description": "The msp_token format is incorrect, or the msp_token is duplicate or the account_id is invalid or the phone number cannot be decrypted.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpClientErrorResponse"
            }
          }
        }
      },
      "500": {
        "description": "Internal Server Error.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpServerErrorResponse"
            }
          }
        }
      }
    },
    "security" : [ {
      "basicAuth" : [ ]
    }, {
      "apiKey" : [ ]
    } ]
  }
```

},
"/cesim/msp/v1/users/{msp_token}/invalidate" : {
"post" : {
"tags" : ["MSP Token invalidation"],
"summary" : "Notify MSP Token Invalid",
"description": "This interface is called by the MSP to notify the CMP that the user's subscription has ended or the user chose to delete the link between CMP account and MSP account. The msp_token cannot be used to request any new ACs and all OPs of the user will be deleted as a consequence.",
"operationId" : "notifyMspTokenInvalid",
"parameters" : [
{
"name": "CMP-request-id",
"in": "header",
"description": "UUID identifying a request.",
"required": true,
"example": "d534080b-9794-4351-88e7-8af9e079ca1e",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "correlation-id",
"in": "header",
"description": "UUID identifying a transaction.",
"required": true,
"example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "client-id",
"in": "header",
"description": "This parameter identifies the MSP at the CMP.",
"required": true,

```
          "example": "dk3kdwkef1",
          "schema": {
            "type": "string"
          }
        }, {
          "name": "target-id",
          "in": "header",
          "description": "This parameter identifies the CMP which is called.",
          "required": true,
          "example": "gKl32ko4",
          "schema": {
            "type": "string"
          }
        },
    {
          "name": "msp_token",
          "in": "path",
          "required": true,
          "example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",
          "description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",
          "schema": {
            "type": "string",
       "format": "uuid"
          }
        }
   ],
      "requestBody": {
        "description": "Request body",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/NotifyTokenInvalidBody"
            }
          }
        },
     "required": true
```

```
    },
    "responses": {
      "204": {
        "description": "The msp_token has been invalidated."
      },
      "400": {
        "description": "Invalid request syntax.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpClientErrorResponse"
            }
          }
        }
      },
      "401": {
        "description": "Unauthorized.",
  "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
            }
          }
        }
      },
      "404": {
        "description": "The user identified by the msp_token cannot be found.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpTokenErrorResponse"
            }
          }
        }
      },
      "500": {
        "description": "Internal Server Error.",
```

```
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpServerErrorResponse"
            }
          }
        }
      }
    },
    "security": [ {
      "basicAuth": [ ]
    }, {
      "apiKey": [ ]
    } ]
  }
},
"/cesim/msp/v1/users/{msp_token}/profiles": {
  "post" : {
    "tags": [ "Send profile information" ],
    "summary": "Send Profile Information",
  "description": "This interface is called by the MSP to send information which is relevant for one specific profile or a list of specified profiles, which all belong to the same user and same subscription (same MSP token).",
    "operationId": "sendProfileInformation",
    "parameters": [
  {
      "name": "CMP-request-id",
      "in": "header",
      "description": "UUID identifying a request.",
      "required": true,
      "example": "d534080b-9794-4351-88e7-8af9e079ca1e",
      "schema": {
        "type": "string",
        "format": "uuid"
      }
    },
  {
      "name": "correlation-id",
```

```
        "in": "header",
        "description": "UUID identifying a transaction.",
        "required": true,
        "example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
        "schema": {
          "type": "string",
      "format": "uuid"
        }
       },
       {
        "name": "client-id",
        "in": "header",
        "description": "This parameter identifies the MSP at the CMP.",
        "required": true,
        "example": "dk3kdwkef1",
        "schema": {
          "type": "string"
        }
       }, {
        "name": "target-id",
        "in": "header",
        "description": "This parameter identifies the CMP which is called.",
        "required": true,
        "example": "gKl32ko4",
        "schema": {
          "type": "string"
        }
       },
    {
        "name": "msp_token",
        "in": "path",
        "required": true,
        "example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",
        "description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",
        "schema": {
          "type": "string",
```

```
      "format": "uuid"
        }
       }
   ],
    "requestBody": {
     "description": "Request body",
     "content": {
      "application/json": {
       "schema": {
        "$ref": "#/components/schemas/SendProfileInformationBody"
       }
      }
     },
     "required": true
    },
    "responses": {
     "200": {
      "description": "The profile information has been updated successfully."
     },
     "401": {
      "description": "Unauthorized.",
  "content": {
       "application/json": {
        "schema": {
         "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
        }
       }
      }
     },
     "403": {
      "description": "The client does not have the necessary permissions to add information to the specified user or profile.",
  "content": {
       "application/json": {
        "schema": {
         "$ref": "#/components/schemas/ApiCmpUnauthorizedErrorResponse"
        }
       }
```

```
        }
      },
      "404": {
        "description": "The user identified by the msp_token or the profile identified by the ICCID cannot be found.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpTokenErrorResponse"
            }
          }
        }
      },
      "422": {
        "description": "Invalid request body.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpClientErrorResponse"
            }
          }
        }
      },
      "500": {
        "description": "Internal Server Error.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiCmpServerErrorResponse"
            }
          }
        }
      }
    },
    "security": [ {
      "basicAuth": [ ]
    }, {
      "apiKey": [ ]
```

```
      } ]
    }
  }
 },
 "components" : {
  "schemas" : {
  "ApiCmpTokenErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
     "code": {
      "type": "string",
      "description": "Error code",
      "example": "20"
     },
     "error": {
      "type": "string",
      "description": "Error description ",
      "example": "The MSP token could not be found."
     }
    }
   },
  "ApiCmpAccountIdErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
     "code": {
      "type": "string",
      "description": "Error code",
      "example": "10"
     },
     "error": {
      "type": "string",
      "description": "Error description ",
      "example": "The Account ID could not be found."
     }
    }
```

```
  },
  "ApiCmpUnauthorizedErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
      "code": {
        "type": "string",
        "description": "Error code",
        "example": "104"
      },
      "error": {
        "type": "string",
        "description": "Error description ",
        "example": "The autorization failed."
      }
    }
  },
  "ApiCmpServerErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
      "code": {
        "type": "string",
        "description": "Error code",
        "example": "90"
      },
      "error": {
        "type": "string",
        "description": "Error description ",
        "example": "An internal server occurred."
      }
    }
  },
  "ApiCmpClientErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
```

```
        "code": {
          "type": "string",
          "description": "Error code",
          "example": "91"
        },
        "error": {
          "type": "string",
          "description": "Error description ",
          "example": "Failed to process the request."
        }
      }
    },
    "NotifyTokenInvalidBody": {
      "type": "object",
      "properties": {
        "reason": {
          "type": "string",
          "description": "An optional short description why the msp_token is invalid. The reason is logged for support purposes in case customer contacts customer care.",
          "example": "User subscription ended."
        }
      }
    },
    "SendMspTokenBody": {
  "required": ["msp_token", "phoneNumber", "subscriptionType", "error", "iv"],
      "type": "object",
      "properties": {
    "msp_token": {
          "type": "string",
          "description": "The federated ID (UUID) which was generated by the MSP to be used by both the MSP and the CMP to refer to the same user. Set the value to null if the error parameter is not null.",
          "example": "649821b4-ca2d-4034-95f1-296bbf8d5088"
        },
    "phoneNumber": {
          "type": "string",
          "description": "The encrypted and Base64 encoded phone number associated with the user's subscription. Set the value to null if the error parameter is not null. Format of
```

unencrypted phone number: [Country Code (no leading '+' sign)] + [Number Planning Area] + [Subscriber Number], example: 919961345678",

"example": "enjzI1TPN4Dtf10uAnpfitDxJzf7SBsXNEDzdA=="

},

"subscriptionType": {

"type": "string",

"description": "Indicates whether the msp_token is linked to a private mobile subscription or business mobile subscription. Set the value to null if the error parameter is not null.",

"example": "private",

"enum": ["private", "business", "unknown"]

},

"error": {

"type": "string",

"description": "Error describing why the MSP token could not be generated by the MSP. Set the value to null if msp_token could be generated by MSP and is sent within the request body.",

"example": "1000: Customer not eligible"

},

"customerGroup": {

"type": "string",

"description": "An optional identifier which can be used to group customers.",

"example": "Market_Germany"

},

"iv": {

"type": "string",

"description": "Base64 String of the initialization vector which is required to decrypt the phone number. Set the value to null if the error parameter is not null.",

"example": "5FRF9spUpq/25/gr"

}

}

},

"SendProfileInformationBody": {

"required": ["profiles"],

"type": "object",

"properties": {

"profiles": {

"type": "array",

"example": [{

"iccid": "8944500805172032953",
"status": "suspended",
"reason": "The contract was suspended."
}],
"items": {
"required": ["iccid", "status", "reason"],
"type": "object",
"properties": {
"iccid": {
"type": "string",
"description": "The ICCID of the profile.",
"example": "8944500805172032953"
},
"status": {
"type": "string",
"description": "The new status of the profile.",
"example": "suspended",
"enum": ["invalid", "valid", "suspended"]
},
"reason": {
"type": "string",
"description": "A short description why the profile(s) information was sent. The reason is logged for support purposes in case customer contacts customer care. Set the value to null if no reason can be given.",
"example": "The contract was suspended."
}
}
}
}
},
"SendActivationCodeBody" : {
"required": ["activationCode", "profileType", "activationCodeRequestID", "error", "iccid", "swappedIccid"],
"type": "object",
"properties": {
"activationCode": {
"type": "string",

"example": "1$VR-01-43-CUST.SC.MY$ESIM.COM$DEF30A27E3CEFD34FA24B4A38D9681A5",

"description": "The activation code which is used to download an operational profile from the SM-DP+ server. Set value to null if the error parameter is set in the request body."

},

"profileType": {

"type": "string",

"example": "personal",

"description": "The type of the profile, e.g. 'personal'. Set value to null if the error parameter is set in the request body."

},

"activationCodeRequestID" : {

"type": "string",

"format": "uuid",

"example": "8b5233eb-592e-49ed-9892-7e32c3cb8871",

"description": "The UUID identifying the request for an activation code which was sent to MSP in the 'Request Activation Code' call. Set value to null if the error parameter is set in the request body."

},

"error": {

"type": "string",

"description": "Error describing why the activation code could be generated. Only non-null if the activation code could not be generated, i.e. set value to null if activation code is sent within the request body.",

"example": "1000: Customer not eligible"

},

"iccid": {

"type": "string",

"description": "The ICCID of the profile which will be downloaded with the provided activation code. Set value to null if the error parameter is set in the request body.",

"example": "925816732327827958834 3"

},

"swappedIccid": {

"type": "string",

"description": "ICCID of the profile which the MSP deactivated (i.e. the swapped profile) to be able to send out a new activation code for the new profile. Set value to null if no profile was swapped (more free slots were available) or if error parameter is set.",

"example": "925816732327827958834 3"

}

}

},
"UpdateSubscriptionInfoBody" : {
"required": ["iv", "phoneNumber", "subscriptionType"],
"type": "object",
"properties": {
"phoneNumber" : {
"type": "string",
"description": "The encrypted and base64 encoded phone number of the user identified by msp_token.",
"example": "enjzI1TPN4Dtf10uAnpfitDxJzf7SBsXNEDzdA=="
},
"subscriptionType": {
"type": "string",
"description": "Indicates whether the msp_token is linked to a private mobile subscription or business mobile subscription.",
"example": "private",
"enum": ["private", "business", "unknown"]
},
"customerGroup": {
"type": "string",
"description": "An optional identifier which can be used to group customers.",
"example": "Market_Germany"
},
"iv": {
"type": "string",
"description": "Base64 encoded initialization vector which is required for the phone number decryption.",
"example": "5FRF9spUpq/25/gr"
}
}
}
},
"securitySchemes": {
"apiKey": {
"type": "apiKey",
"name": "api-key",
"in": "header"
},

```
      "basicAuth": {
        "type": "http",
        "in": "header",
        "scheme": "basic"
      }
    }
  }
}
```


B.2 REST APIs provided by MSP and called by CMP


```
{
  "openapi": "3.0.3",
  "info": {
    "title": "MSP CESIM CMP API",
    "description": "REST APIs provided by MSP and called by CMP.",
    "version": "1.0.0"
  },
  "servers": [
    {
      "url": "https"
    }
  ],
  "paths": {
    "/v1/activation-code-requests/{msp_token}": {
      "post": {
        "tags" : [ "Activation code request" ],
        "summary": "Request activation code",
        "description": "The CMP calls this interface to request an AC for a specific user. The MSP can send back the AC either in the response body (synchronous mode) or with the REST API endpoint 'Send Activation Code' (asynchronous mode). The AC will then be sent to a vehicle where it will be used to download an OP.",
        "operationId": "requestActivationCode",
        "parameters": [
          {
```

"name": "CMP-request-id",
"in": "header",
"required": true,
"example": "d534080b-9794-4351-88e7-8af9e079ca1e",
"description": "UUID identifying a request.",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "correlation-id",
"in": "header",
"required": true,
"example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
"description": "UUID identifying a transaction.",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "client-id",
"in": "header",
"required": true,
"example": "dk3kdwkef1",
"description": "This parameter identifies the CMP at the MSP.",
"schema": {
"type": "string"
}
},
{

"name": "target-id",

"in": "header",

"required": true,

"example": "gKl32ko4",

"description" : "This parameter identifies the MSP which is called.",

"schema": {

"type": "string"

}

},

{

"name": "msp_token",

"in": "path",

"required": true,

"example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",

"description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",

"schema": {

"type": "string",

"format": "uuid"

}

}

],

"requestBody": {

"description": "Request body",

"content": {

"application/json": {

"schema": {

"$ref": *"#/components/schemas/RequestActivationCodeBody"*

}

}

},

"required": true

},
"responses": {
"200": {
"description": "Success. The activation code is included in response body.",
"content": {
"application/json": {
"schema": {
"$ref": *"#/components/schemas/ActivationCodeResponse"*
}
}
}
},
"201": {
"description": "The request has been stored and will be processed, i.e. the activation code will be sent to CMP in asynchronous mode via the REST API endpoint 'Send Activation Code'."
},
"401": {
"description": "Unauthorized.",
"content": {
"application/json": {
"schema": {
"$ref": *"#/components/schemas/ApiMspUnauthorizedErrorResponse"*
}
}
}
},
"404": {
"description": "The user identified by the msp_token could not be found.",
"content": {
"application/json": {
"schema": {
"$ref": *"#/components/schemas/ApiMspTokenErrorResponse"*
}

```
          }
         }
        },
       "422": {
         "description": "No activation code can be created.",
         "content": {
          "application/json": {
           "schema": {
            "$ref": "#/components/schemas/ApiMspActivationCodeErrorResponse"
           }
          }
         }
        },
        "500": {
         "description": "Internal Server Error.",
         "content": {
          "application/json": {
           "schema": {
            "$ref": "#/components/schemas/ApiMspServerErrorResponse"
           }
          }
         }
        }
       }
      }
     },
    "/v1/statuses/{msp_token}": {
     "post": {
      "tags": [ "Profile status notification" ],
      "summary": "Send CSIM status",
      "description": "This interface is called by the CMP to notify the MSP of OP status updates. In addition to the notifications set in the SGP.22, a backend-to-backend profile status notification is also sent to MSPs.",
```

"operationId": "sendCsimStatus",
"parameters": [
{
"name": "CMP-request-id",
"in": "header",
"required": true,
"example": "d534080b-9794-4351-88e7-8af9e079ca1e",
"description": "UUID identifying a request.",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "correlation-id",
"in": "header",
"required": true,
"example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
"description": "UUID identifying a transaction.",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "client-id",
"in": "header",
"required": true,
"example": "dk3kdwkef1",
"description": "This parameter identifies the CMP at the MSP.",
"schema": {
"type": "string"

```
          }
        },
    {
        "name": "target-id",
        "in": "header",
        "required": true,
        "example": "gKl32ko4",
    "description" : "This parameter identifies the MSP which is called.",
        "schema": {
          "type": "string"
        }
      },
      {
        "name": "msp_token",
        "in": "path",
        "required": true,
        "example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",
        "description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",
        "schema": {
          "type": "string",
      "format": "uuid"
        }
      }
      ],
      "requestBody": {
        "description": "Request body",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/SendCsimStatusBody"
            }
```

```
        }
       },
       "required": true
      },
      "responses": {
       "201": {
        "description": "The request has been stored and will be processed."
       },
       "401": {
        "description": "Unauthorized.",
    "content": {
          "application/json": {
           "schema": {
            "$ref": "#/components/schemas/ApiMspUnauthorizedErrorResponse"
           }
          }
         }
       },
       "404": {
        "description": "The user identified by the msp_token could not be found.",
        "content": {
         "application/json": {
          "schema": {
           "$ref": "#/components/schemas/ApiMspTokenErrorResponse"
          }
         }
        }
       },
       "422": {
        "description": "The request cannot be processed.",
        "content": {
         "application/json": {
```

```
            "schema": {
              "$ref": "#/components/schemas/ApiMspClientErrorResponse"
            }
          }
        }
      },
      "500": {
        "description": "Internal Server Error.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiMspServerErrorResponse"
            }
          }
        }
      }
    }
  },
  "/v1/users/{msp_token}": {
    "delete": {
      "tags": ["MSP token invalidation"],
      "summary": "Request invalidate MSP token",
      "description": "This interface is called by the CMP to request the invalidation of a MSP token. This call happens after the user chooses to cancel the link between CMP account and MSP account. The MSP token cannot be used to request any new activation codes any longer and all operational profiles of the user will be deleted as a consequence. Note: The MSP must deactivate all profiles immediately after receiving 'Request invalidate MSP token' and not wait until CSIM status with profile deleted is sent from the CMP.",
      "operationId": "requestInvalidateFederatedID",
      "parameters": [
        {
          "name": "CMP-request-id",
          "in": "header",
```

"required": true,
"example": "d534080b-9794-4351-88e7-8af9e079ca1e",
"description": "UUID identifying a request.",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "correlation-id",
"in": "header",
"required": true,
"example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
"description": "UUID identifying a transaction.",
"schema": {
"type": "string",
"format": "uuid"
}
},
{
"name": "client-id",
"in": "header",
"required": true,
"example": "dk3kdwkef1",
"description": "This parameter identifies the CMP at the MSP.",
"schema": {
"type": "string"
}
},
{
"name": "target-id",
"in": "header",

"required": true,

"example": "gKl32ko4",

"description": "This parameter identifies the MSP which is called.",

"schema": {

"type": "string"

}

},

{

"name": "msp_token",

"in": "path",

"required": true,

"example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",

"description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",

"schema": {

"type": "string",

"format": "uuid"

}

}

],

"responses": {

"204": {

"description": "The msp_token has been invalidated successfully."

},

"401": {

"description": "Unauthorized.",

"content": {

"application/json": {

"schema": {

"$ref": *"#/components/schemas/ApiMspUnauthorizedErrorResponse"*

}

}

```
      }
    },
    "404": {
      "description": "The user identified by the msp_token could not be found.",
      "content": {
        "application/json": {
          "schema": {
            "$ref": "#/components/schemas/ApiMspTokenErrorResponse"
          }
        }
      }
    },
    "422": {
      "description": "The request cannot be processed.",
      "content": {
        "application/json": {
          "schema": {
            "$ref": "#/components/schemas/ApiMspClientErrorResponse"
          }
        }
      }
    },
    "500": {
      "description": "Internal Server Error.",
      "content": {
        "application/json": {
          "schema": {
            "$ref": "#/components/schemas/ApiMspServerErrorResponse"
          }
        }
      }
    }
```

```
      }
    }},
  "/v1/profiles/status/{msp_token}": {
    "get": {
      "tags": ["Profile status request"],
      "summary": "Request Profile Status",
      "description": "This interface is called by the CMP to request the status of all profiles or a specific profile that are not deleted for a federated id on MSP side. Using a specific iccid in the parameters can be added to get the status of a single iccid.",
      "operationId": "requestProfileStatus",
      "parameters": [
        {
          "name": "CMP-request-id",
          "in": "header",
          "required": true,
          "example": "d534080b-9794-4351-88e7-8af9e079ca1e",
    "description": "UUID identifying a request.",
          "schema": {
            "type": "string",
            "format": "uuid"
          }
        },
        {
          "name": "correlation-id",
          "in": "header",
          "required": true,
          "example": "77096b5d-0f42-4c35-9102-3b3a8d7d57e7",
    "description": "UUID identifying a transaction.",
          "schema": {
            "type": "string",
      "format": "uuid"
          }
        },
```

{
"name": "client-id",
"in": "header",
"required": true,
"example": "dk3kdwkef1",
"description": "This parameter identifies the CMP at the MSP.",
"schema": {
"type": "string"
}
},
{
"name": "target-id",
"in": "header",
"required": true,
"example": "gKl32ko4",
"description": "This parameter identifies the MSP which is called.",
"schema": {
"type": "string"
}
},
{
"name": "msp_token",
"in": "path",
"required": true,
"example": "3d8db5e5-dd84-40bf-836c-bb686f9f707e",
"description": "The federated ID (UUID) which was generated by the MSP and sent to CMP via REST API endpoint 'Send MSP Token' to be used by both the MSP and the CMP to refer to the same user.",
"schema": {
"type": "string",
"format": "uuid"
}
},

```
        {
          "name": "iccid",
          "in": "query",
          "example": "8944500805172032953",
          "description": "The ICCID for which the current status shall be retrieved.",
          "required": false,
          "schema": {
            "type": "string"
          }
        }
      ],
      "responses": {
        "200": {
          "description": "Success. The status of the profile(s) is included in the response body.",
    "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/RequestProfileStatusResponseBody"
            }
          }
        }
      },
      "401": {
        "description": "Unauthorized.",
    "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiMspUnauthorizedErrorResponse"
            }
          }
        }
      },
```

```
      "404": {
        "description": "The user identified by the msp_token could not be found.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiMspTokenErrorResponse"
            }
          }
        }
      },
      "422": {
        "description": "The request cannot be processed.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiMspClientErrorResponse"
            }
          }
        }
      },
      "500": {
        "description": "Internal Server Error.",
        "content": {
          "application/json": {
            "schema": {
              "$ref": "#/components/schemas/ApiMspServerErrorResponse"
            }
          }
        }
      }
    }}
```

```
 },
 "components": {
  "schemas": {
   "ApiMspTokenErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
     "code": {
      "type": "string",
      "description": "Error code",
      "example": "20"
     },
     "error": {
      "type": "string",
      "description": "Error description ",
      "example": "The MSP token could not be found."
     }
    }
   },
  "ApiMspUnauthorizedErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
     "code": {
      "type": "string",
      "description": "Error code",
      "example": "104"
     },
     "error": {
      "type": "string",
      "description": "Error description ",
      "example": "The autorization failed."
```

```
        }
       }
     },
    "ApiMspActivationCodeErrorResponse": {
      "required": [ "code", "error" ],
      "type": "object",
      "properties": {
        "code": {
          "type": "string",
          "description": "Error code",
          "example": "49"
        },
        "error": {
          "type": "string",
          "description": "Error description ",
          "example": "Failed to generate an activation code for the MSP Token."
        }
      }
    },
   "ApiMspServerErrorResponse": {
      "required": [ "code", "error" ],
      "type": "object",
      "properties": {
        "code": {
          "type": "string",
          "description": "Error code",
          "example": "90"
        },
        "error": {
          "type": "string",
          "description": "Error description ",
          "example": "An internal server occurred."
```

```
        }
      }
    },
  "ApiMspClientErrorResponse": {
    "required": [ "code", "error" ],
    "type": "object",
    "properties": {
      "code": {
        "type": "string",
        "description": "Error code",
        "example": "91"
      },
      "error": {
        "type": "string",
        "description": "Error description ",
        "example": "Failed to process the request."
      }
    }
  },
  "RequestProfileStatusResponseBody": {
    "required": ["profiles"],
    "type": "object",
    "properties": {
      "profiles": {
        "type": "array",
  "example": [
    {
          "iccid": "28918293818491828371",
          "status": "enabled"
        }, {
          "iccid": "28918293818491828372",
          "status": "disabled"
```

```
            }
        ],
        "items" : {
          "required": ["iccid", "status"],
          "type": "object",
          "properties": {
          "iccid": {
            "type": "string",
            "description": "The ICCID from the profile.",
            "example": "28918293818491828371"
            },
            "status": {
            "type": "string",
            "description": "The current status of the profile.",
            "example": "enabled",
            "enum": [ "enabled", "disabled", "installed", "prepared" ]
            }
          }
             },
              "description": "The type of the profile, e.g. 'personal'."
            }
          }
        },
        "RequestActivationCodeBody": {
          "required": ["profileType", "deviceType", "eid", "activationCodeRequestID",
"replaceIccid"],
          "type": "object",
          "properties": {
            "profileType": {
              "type": "string",
              "example": "personal",
              "description": "The type of the profile, e.g. 'personal'."
            },
```

"deviceType": {
"type": "string",
"example": "tcu",
"default": "tcu",
"description": "A type identifier for the eSIM device in the vehicle."
},
"imei": {
"type": "string",
"example": "880000862471845",
"description": "International Mobile Equipment Identity."
},
"eid": {
"type": "string",
"example": "89049032000001000000000831934057",
"description": "Embedded Integrated Circuit Card ID."
},
"activationCodeRequestID": {
"type": "string",
"format": "uuid",
"example": "14abc1e2-338f-11e6-ac61-9e71128cae77",
"description": "A UUID identifying the request for a new activation code. This activationCodeRequestID has to be returned to CMP in case the activation code is sent via the asynchronous mode (using the REST API endpoint 'Send Activation Code')."
},
"replaceIccid": {
"type": "string",
"example": "89310410106543780000",
"description": "The profile with this ICCID is a suggested replace candidate when all available eSIM profile slots of the customer are already used. Set value to null if no replace ICCID can be given. It is the MSP's decision if a profile swap is required and if the suggested profile or a different one shall be swapped."
}
}
},

"ActivationCodeResponse": {
"required": ["activationCode", "profileType", "swappedIccid"],
"type": "object",
"properties": {
"iccid": {
"type": "string",
"example": "89310410106543780000",
"description": "The Integrated Circuit Card ID of the profile which will be downloaded with the activation code."
},
"activationCode": {
"type": "string",
"example": "1$VR-01-43-CUST.SC.MY-ESIM.COM$DEF30A27E3CEFD34FA24B4A38D9681A5",
"description": "The activation code which is used to download an operational profile from the SM-DP+ server."
},
"profileType": {
"type": "string",
"description": "The type of the profile. Currently, there is only the profile type 'personal'.",
"example": "personal",
"enum": ["personal"]
},
"swappedIccid": {
"type": "string",
"example": "98341201501601380129",
"description": "Integrated Circuit Card ID, unique profile identifier of the profile which the MSP deactivated to be able to send out the activation code for the new profile. In case no profile swap was done, set this value to null."
}
}
},

"SendCsimStatusBody": {
"required": ["iccid", "status", "eid"],
"type": "object",
"properties": {
"iccid": {

```
          "type": "string",
          "example": "89310410106543780000",
          "description": "The Integrated Circuit Card ID."
        },
        "eid": {
          "type": "string",
          "description": "The Embedded Integrated Circuit Card ID.",
          "example": "89049032000001000000000831934057"
        },
        "status": {
          "type": "string",
          "example": "deleted",
          "description": "The new status of the profile identified by the ICCID.",
          "enum": [ "deleted", "enabled", "disabled", "installed", "installation_failed"]
        }
      }
    }
  }
}
```

Annex C **Protobuf Code (Google Protocol Buffers)**

C.1 CMP Message Types

```
syntax = "proto3";

package cmp_types;

message RequestId {
  string vehicleID = 1;
  uint64 timestamp = 2;
}

message Profile {
  string iccid = 1;
  string cmpUserId = 2;
  ProfileStatus status = 3;
  ProfileType type = 4;
  string mcc = 5;
  string mnc = 6;
}

enum ProfileStatus {
  DELETED = 0;
  ENABLED = 1;
  DISABLED = 2;
}

enum ProfileType {
  DEFAULT = 0;
  PERSONAL = 1;
  PRIVATE = 2;
  BUSINESS = 3;
}

enum RequestActivationCodeResult {
  UNKNOWN_VIN = 0;
  UNKNOWN_EID = 1;
  UNKNOWN_IMEI = 2;
  UNKNOWN_USER = 3;
  NO_SUBSCRIPTION_FOUND = 4;
  NO_ACTIVATION_CODE_AVAILABLE = 5;
  TERMS_NOT_ACCEPTED = 6;
}

enum DownloadInstallResult {
```

```
  PROFILE_INSTALLATION_SUCCESS = 0;
  PROFILE_INSTALLATION_UNKNOWN_ERROR = 1;
  PROFILE_DOWNLOAD_FAILED_NO_NETWORK_CONNECTION = 2;
  PROFILE_DOWNLOAD_FAILED_PROXY_COMMUNICATION_ERROR = 3;
  PROFILE_DOWNLOAD_FAILED_SMDP_COMMUNICATION_ERROR = 4;
  PROFILE_DOWNLOAD_FAILED_SMDP_DOWNLOAD_ERROR = 5;
  PROFILE_INSTALLATION_FAILED = 6;
}

enum RequestDeleteProfileResult {
  DELETE_PROFILE_SUCCESS = 0;
  DELETE_PROFILE_FAILED = 1;
  DELETE_PROFILE_NOT_FOUND = 2;
  DELETE_PROFILE_USER_NOT_FOUND = 3;
}

enum RequestCSIMInfoResult {
  CSIM_INFO_SUCCESS = 0;
  CSIM_INFO_GENERAL_ERROR = 1;
}

enum RequestDeleteAllProfilesResult {
  DELETE_ALL_PROFILES_SUCCESS = 0;
  DELETE_ALL_PROFILES_FAILED = 1;
  DELETE_ALL_PROFILES_NO_PROFILES_FOUND = 2;
}
```

C.2 APIs provided by the CMP backend to be called by the vehicle

```
syntax = "proto3";

package cmp_backend;

import "cmp_types.proto";

message RequestActivationCode {
  cmp_types.RequestId requestId = 1;
  string cmpUserId = 2;
  string imei = 3;
  string eid = 4;
  message Response {
        cmp_types.RequestActivationCodeResult result = 1;
  }
}

message SendProfileDownloadInstallInfo {
  cmp_types.RequestId requestId = 1;
```

```
  string cmpUserId = 2;
  string iccid = 3;
  string eid = 4;
  string activationCode = 5;
  cmp_types.DownloadInstallResult result = 6;
  string functionExecutionStatus = 7;
  uint32 csimErrorCode = 8;
}

message SendCSIMStatus {
  cmp_types.RequestId requestId = 1;
  string eid = 2;
  string imei = 3;
  string csimNetwork = 4;
  string psimNetwork = 5;
  repeated cmp_types.Profile profiles = 6;
}

message SendEnableProfileErrorReport {
  cmp_types.RequestId requestId = 1;
  cmp_types.Profile profile = 2;
  uint32 csimErrorCode = 3;
}

message SendDisableProfileErrorReport {
  cmp_types.RequestId requestId = 1;
  cmp_types.Profile profile = 2;
  uint32 csimErrorCode = 3;
}
```

C.3 APIs provided by the vehicle to be called by the CMP backend

```
syntax = "proto3";

package cmp_vehicle;

import "cmp_types.proto";

message SendActivationCode {
  cmp_types.RequestId requestId = 1;
  string cmpUserId = 2;
  cmp_types.ProfileType profileType = 3;
  string activationCode = 4;
}

message RequestDeleteProfile {
  cmp_types.RequestId requestId = 1;
```

```
  string cmpUserId = 2;
  string iccid = 3;

  message Response {
    cmp_types.RequestDeleteProfileResult result = 1;
    uint32 csimErrorCode = 2;
  }
}

message RequestCSIMInfo {
  cmp_types.RequestId requestId = 1;
  message Response {
    cmp_types.RequestCSIMInfoResult result = 1;
    uint32 csimErrorCode = 2;
  }
}

message RequestDeleteAllProfiles {
  cmp_types.RequestId requestId = 1;
  message Response {
   cmp_types.RequestDeleteAllProfilesResult result = 1;
   uint32 csimErrorCode = 2;
  }
}
```

Annex D Document Management

D.1 Document History

Version	Date	Brief Description of Change	Approval Authority	Editor / Company
AID.02 v1.0	18/05/2022	First Version of AID.02 contains the CRs below	ISAG	Yolanda Sanz/GSMA
		CR0001R00 – Introduction and Scope		
		CR0002R00- Actors		
		CR004r01 – Profile Download Use case		
		CR003R01 – Onboarding Use case		
		CR005R01 Introduction and Scope section		
		CR006R01 Account federation between MSP and CMP Car provisioning		
		CR007R01 Architecture		
		CR008R03 Message Flow Onboarding		
		CR009R02 Message Flow Car Provisioning		
		CR010R01 Message Flow Withdrawn Onboarding		
		CR0012R01 Message Flow Delete		
		CR0013R01 Message Sync		
		CR0014R00 Messagae Flow Onboarding		
		CR0015R01 Message flow car provisioning		
		CR0016R01 Message flow Withdraw Onboarding		
		CR0017R01 Message Flow Delte		
		CR0018R01 Message Synch		
		CR0019R00 Terminology Aligment OP		
		CR0020R03 Onboarding Data and Functions		
		CR022R00 Editorial Correction in Section 1.1 (Scope		

		CR024R01 CMP Status update Interfaces		
		CR0021R04 Request and Send Activation Code		
		CR0023R01 Status Update Interfaces provided by MSP		
		CR0025R00 New http header format		
		CR0011R02 Profile Swap Concept		
		CR0026R01 Error Codes		
		CR0027R01 Naming and Typo		
		CR0028R01 Request AC Response		
		CR0029R01 Account ID Format CR0030 Onboarding via MSP touchpoint		
		CR0031R01 Remark on Invalidate MSP Token		
		CR0032R00 Send Profile Information in data format		
AID.02 v2.0	04 May 2023	Draft 0 of AID.02 v2.0 with the based line of AID.02 v1.0	ISAG	Yolanda Sanz, GSMA
		CR2001R00 Scope for AID2 interface		
		CR2002R01 New functions for Car Provisioning AiD2		
		CR2003R02 New functions for Car Provisioning AiD2		
		CR2004R00 New functions for Car Provisioning AiD2		
		CR2005R01 Add binding of profile to eid as additional step into procedure “Car Provisioning”		
		CR2007R01 Response Code invalid eid		
		CR2008R02 Enable and Disable Profile		
		CR2009R01 Request CSIM Info		
		CR2010R03 Delete all profiles		
		CR2011R00 Send CSIM status in Car Provisioning		
		CR2014R00 Varius user comments		

		CR2012R00 Car Connectivity CR2013R00 Update Car Provisioning Figure Fix some editorial issuess		
AID.02 v2.1	26 January 2024	CR21001R00 Optional and Default Parameters	ISAG	Yolanda Sanz, GSMA
		CR21002R01 Optional and Default Parameters		
		CR21003R00 Additional minor comments regarding AID1 interface		
		CR21004R00 json code for AID1 interface		
		CR21005R00 protobuf code for AID2 interface		
		CR21006R00 Parameter changes in AID2 interface		
		CR21007R00 References to common data elements		
		CR21008R00 Changes in json code		
		CR21009R00 Mandatory/optional header parameters		
		CR21010R00 Revised texts in Procedures		
		CR21011R00 Changes in Profile Swap Procedure		
		CR21012R00 Changes in Functions		

Other Information

Type	Description
Document Owner	AID Group
Editor / Company	Yolanda Sanz/GSMA

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@gsm.org

Your comments or suggestions & questions are always welcome.