Draft ETSI GS MEC 040 V3.1.10 (2024-02)

Multi-access Edge Computing (MEC); Federation enablement APIs

Disclaimer: This DRAFT is a working document of ETSI ISG MEC. It is provided for information only and is still under development within ETSI ISG MEC. ETSI and its Members accept no liability for any further use/implementation of this Specification.

Non-published MEC drafts stored in the "Open Area" are working documents, these may be updated, replaced, or removed at any time

Do not use as reference material.

Disclaimer

Do not cite this document other than as "work in progress".

The present document has been produced and approved by the Multi-access Edge Computing (MEC) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Approved and published Specifications and reports for implementation of the MEC system shall be obtained via the ETSI Standards search page at: http://www.etsi.org/standards-search

Reference
RGS/MEC-0040v321FederationAPI

Keywords
API, interworking, MEC, service

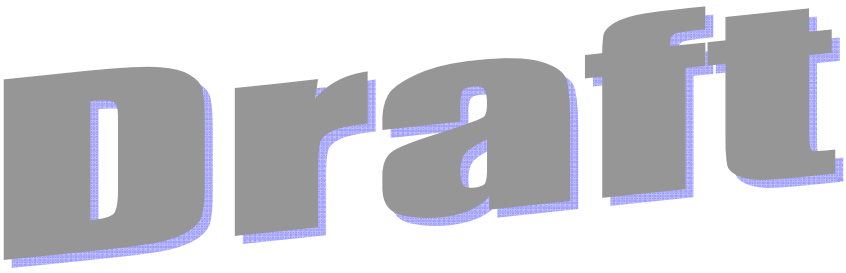
ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents



Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Multi-access Edge Computing (MEC).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document focuses on the functionalities enabled over the relevant reference points (i.e. Mfm and Mff) to support MEC federation. It describes the information flows, required information, and specifies the necessary operations, data models and API definitions. The present document carefully considers the relevant work of other industry bodies relating to MEC federation (e.g. GSMA OPG, 5GAA, etc.) and all relevant work done in ETSI.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI GS MEC 001: "Multi-access Edge Computing (MEC); Terminology".

[2] void

[3] void

[4] void

[5] GSMA Permanent Reference Document OPG.04: "Operator Platform - East-Westbound Interface APIs", v2.0, March 2023.

[6] 3GPP TS 23.502: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Procedures for the 5G System (5GS); Stage 2 (Release 18)".

[7] ETSI GS MEC 010-2: "Multi-access Edge Computing (MEC); MEC Management; Part 2: Application lifecycle, rules and requirements management".

[8] void.

[9] IETF RFC 4122: "A Universally Unique IDentifier (UUID) URN Namespace".

NOTE: Available at https://www.rfc-editor.org/info/rfc4122.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] GSMA Permanent Reference Document: "Operator Platform Telco Edge Requirements", v1.0, Jun. 2021.

NOTE: Available at https://www.gsma.com/futurenetworks/wp-content/uploads/2021/06/OPG-Telco-Edge-Requirements-2021.pdf.

[i.2] ETSI GR MEC 035: "Multi-access Edge Computing (MEC); Study on Inter-MEC systems and MEC-Cloud systems coordination".

[i.3] ETSI GS MEC 002: "Multi-access Edge Computing (MEC); Use Cases and Requirements".

[i.4] ETSI GS MEC 003: "Multi-access Edge Computing (MEC); Framework and Reference Architecture".

[i.5] Void.

[i.6] Void.

[i.7] OpenAPI™ Specification.

NOTE: Available at https://github.com/OAI/OpenAPI-Specification.

[i.8] ETSI GS MEC 009: "Multi-access Edge Computing (MEC); General principles, patterns and common aspects of MEC Service APIs".

[i.9] Void.

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI GS MEC 001 [1] and the following apply:

App	Application
CHF	Charging Functions
CR	Cloud Resources
E/WBI	East/West Bound Interface
GSMA	GSM Association
MEF	MEC Federator
MNO	Mobile Network Operator
NBI	NorthBound Interface
NR	Network Resources
OEM	Original Equipment Manufacturer
OP	Operator Platform
SBI	SouthBound Interface
UNI	User Network Interface

4 Overview

4.1 Introduction

The present document specifies Federation enablement APIs that enable the shared usage of MEC services and applications across different systems (e.g. MEC system, Cloud system).

Clause 4 introduces the relevant work of other industry bodies e.g. GSMA OPG.

Clause 5 presents the reference scenarios for the MEC federation, and introduces the functionalities enabled via the relevant reference points (i.e. Mfm and Mff). It provides the high-level information flows and describes the necessary operations.

Clause 6 describes the data models that can be exchanged over the Federation enablement APIs, which provide detailed descriptions of all information elements used for MEC federation.

Clause 7 defines the actual Federation enablement APIs providing detailed information of how information elements are mapped into a RESTful API design.

4.2 GSMA Operator Platform and its interfaces

According to the GSMA Permanent Reference Document (PRD), "Operator Platform Telco Edge Requirements" [i.1], an Operator Platform (OP) is a facilitator of subscribers' seamless access to edge applications instantiated within a federation of edge networks involving multiple owners. Such seamless access is needed either when subscribers roam to visited networks or when a partner network is a better choice for edge application instantiation.

The objective of the OP concept is to guide the industry ecosystem, i.e. MNOs, vendors, OEMs and service providers towards shaping a common solution for the exposure of network capabilities. As an initial step, [i.1] provides both an end-to-end definition and requirements of the OP for the support of edge computing. In further details, the GSMA defines OP requirements as well as OP architecture and functional modules. Therefore, aim of GSMA is to engage with standardization and open source communities that will undertake the standard definition of the OP. As depicted in Figure 4.2-1, the following OP interfaces have been defined in [i.1].

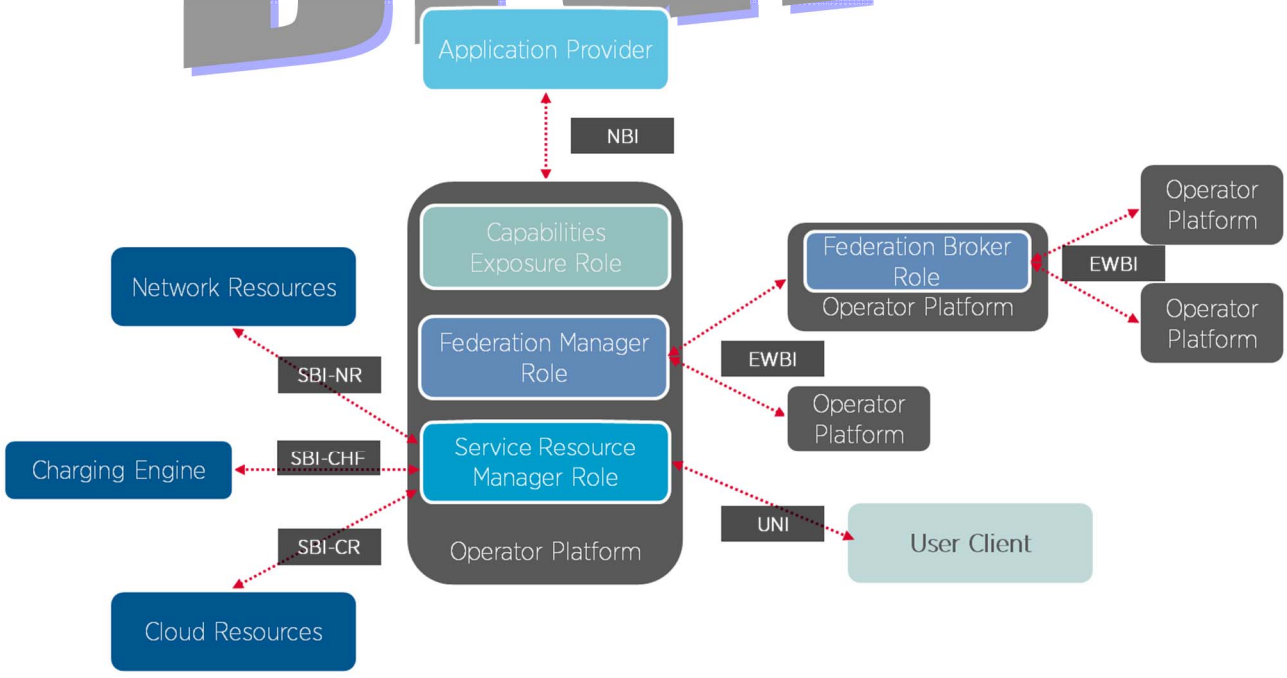


Figure 4.2-1: High-level OP reference architecture (source: [i.1])

- Northbound Interface (NBI);

- Southbound Interface (SBI); Cloud Resources (SBI-CR);
- Southbound Interface (SBI); Network Resources (SBI-NR);
- Southbound Interface (SBI); Charging Functions (SBI-CHF);
- User Network Interface (UNI);
- East/West Bound Interface (E/WBI).

5 Description of the services (informative)

5.1 Federation enablement service introduction

Federation enablement APIs offers services such as discovery, information exchange and application life cycle management to enable the inter-work of one MEC system with another MEC system. The related requirements were carefully studied and extracted from various use cases in ETSI GR MEC 035 [i.2] including V2X services scenario, multi-operator environment, Application instance transfer between a MEC system and a MEC/Cloud system, connecting different services, immersive AR game scenario, edge service delivery through visited network and edge node sharing.

The extracted requirements are listed as follows, summarized from ETSI GS MEC 002 [i.3].

- MEC system discovery ([Federation-02])
- MEC platform discovery ([Federation-03])
- Information exchange between MEC systems ([Federation-04])
- Information exchange between MEC platforms ([Federation-05])
- Support handling direct/indirect MEC system communication ([Federation-06])
- MEC application discovery ([Federation-07])
- MEC application on-boarding/instantiation ([Federation-08])
- Information exchange among MEC applications ([Federation-09])
- MEC service discovery ([Federation-10])

NOTE: Reusing the data models and APIs for MEC-Cloud coordination is considered if applicable, but its information flow is out of scope of the present document.

5.2 Sequence diagrams

5.2.1 Introduction

The rest of clause 5.2 introduces the following sequence diagrams based on the extracted requirements.

- Registration of MEC system(s) to the federation (clause 5.2.2.1)
- Discovery:
  - MEC system discovery (clause 5.2.2.2)
  - MEC application discovery (clause 5.2.2.3)
  - MEC service discovery (clause 5.2.2.4)
- Application package management and Application instance lifecycle management (clause 5.2.2.5)

NOTE 1: Support handling direct/indirect MEC system communication is satisfied by MEC Federator as defined in ETSI GS MEC 003 [i.4].

NOTE 2: The requirement for registration is based on the premise that multiple MEOs can register to a single MEF.

5.2.2 Request/Response model

5.2.2.1 Registration/Update/Deregistration of MEC system to the MEC federator

5.2.2.1.1 Registration

The registration information flow is used for enabling a MEO to register its MEC system information with a MEC Federator over Mfm reference point, as depicted in Figure 5.2.2.1.1-1.

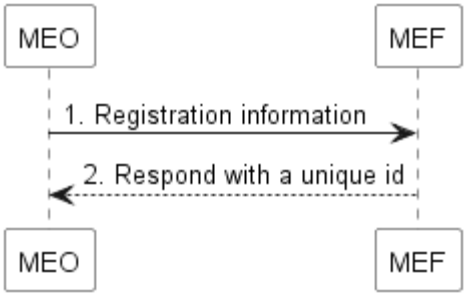


Figure 5.2.2.1.1-1: Information flow of Registration

Registration procedure consists of the following steps:

1) The MEO sends a registration request to the MEC federator.
2) The MEC federator responds with a unique ID among federation members.

5.2.2.1.2 Update

Information flow of update of MEC system(s) to the federation is depicted in Figure 5.2.2.1.2-1.

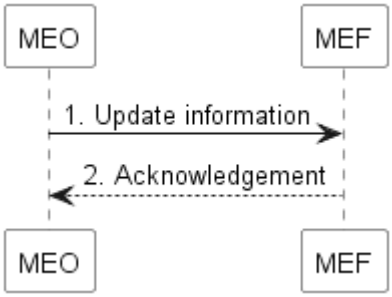


Figure 5.2.2.1.2-1: Information flow of Update

Update procedure consists of the following steps:

1) The MEO sends an update request to MEC Federator.
2) MEC Federator returns an acknowledgement to MEO.

5.2.2.1.3 Deregistration

Information flow of deregistration of MEC system(s) from the federation is depicted in Figure 5.2.2.1.3-1.

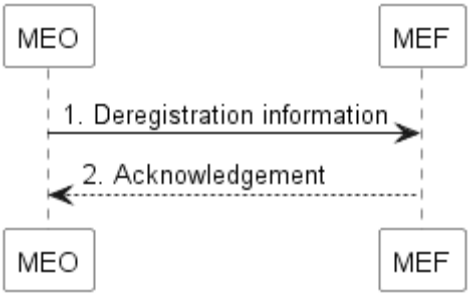


Figure 5.2.2.1.3-1: Information flow of Deregistration

Deregistration procedure consists of the following steps:

1) The MEO sends an update request to MEC Federator.

2) MEC Federator returns an acknowledgement to MEO.

5.2.2.2 MEC system discovery

Information flow of MEC system discovery is used for enabling MEO to be aware of another MEC system. MEC system discovery is the primitive and essential procedure for enabling the other functionalities relating to Feature MECFederation. The information flow is depicted in Figure 5.2.2.2-1.

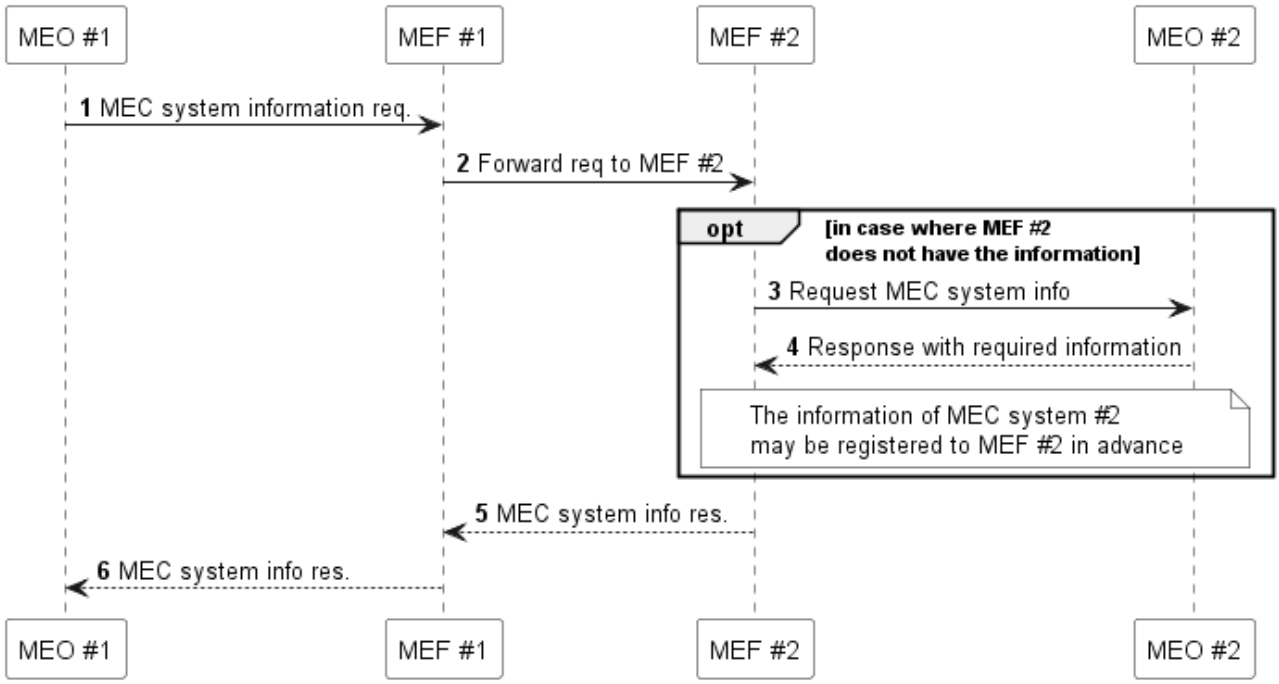


Figure 5.2.2.2-1: Information flow of MEC system discovery

As a prerequisite of this flow, MEC Federator Discovery is conducted among MEC Federators, which means MEC Federators are aware of each other in advance.

1) The MEO #1 sends a MEC system information request to MEC Federator #1 over Mfm reference point. This request is triggered by MEC platform or MEC Application instance.

2) MEC Federator #1 forwards the request to MEC Federator #2.

3) In case where MEC Federator #2 does not have the desired information (which means MEO #2 does not register its own information in advance), MEC Federator #2 sends a MEC system information request to MEO #2 over Mfm reference point.

4) MEO #2 responds with the information of its own system to MEC Federator #2.

5) MEC Federator #2 forwards the response to MEC Federator #1.

6) MEC Federator #1 forwards the response to MEO #1.

5.2.2.3 MEC application instance discovery

MEC application instance discovery refers to a process triggered by a MEC application instance or the MEC management (MEPM or MEO), which discovers one or more MEC application instances in the MEC federation of the application from which it was instantiated. For example, the discovery may be based on information of a specific MEC application instance or of the corresponding application descriptor , may be with restriction information on target location. This process can be triggered, for instance, in the cases calling for MEC application instance-to-instance communication (e.g. neighbouring vehicles communicating with different MEC application instances may need to cooperate via those MEC application instances. grouped users communicating with different MEC application instances may need to communicate with each other via those MEC application instances, or grouped users may be gathered from different MEC systems and served by a single MEC application instance). This process can be triggered by the MEC management, for instance, in the cases the MEC management decides to find a target application instance in other MEC system after receiving information about the user device associated to a MEC application instance moving out of the service area of the current MEC system (e.g. the AF receives notification about User Plane change as described in clause 4.3.6.3 of 3GPP TS 23.502 [6]). The information flow is depicted in Figure 5.2.2.3-1.

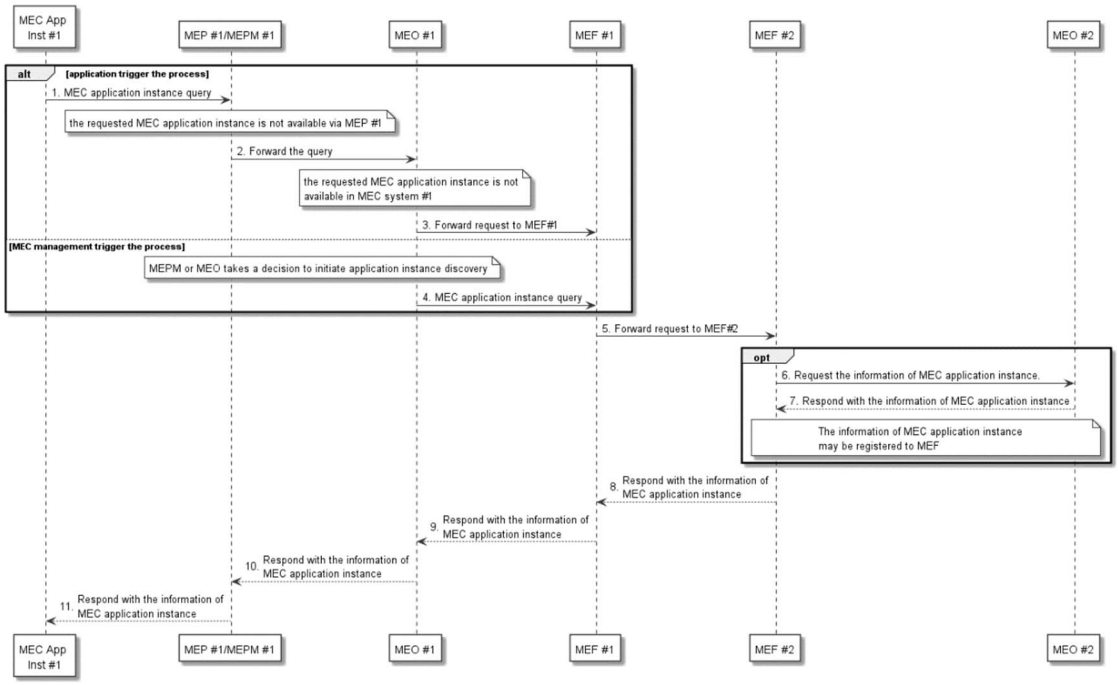


Figure 5.2.2.3-1: Information flow of MEC application discovery

Procedure of MEC application instance discovery consists of the following steps:

1) MEC application instance #1 sends a query to MEP #1 to discover a MEC application instance.

NOTE 1: MEC application instance #1 may know either the identifier of the requested application instance or identifier of application descriptor.

2) In the case where the desired MEC application instance is not available via MEP #1, MEP #1 forwards the query to MEO #1 via MEPM #1.

NOTE 2: How to handle the query between MEP #1 and MEO #1 is not further specified in the present document.

3) MEO #1 examines if the requested MEC application instance is available in MEC system #1. In case where the MEC application instance is not available in MEC system #1, MEO #1 forwards the query to MEF #1. Otherwise, go to step 10.

Alternatively, MEC management triggers application discovery, the following step 4 is executed:

4) If the MEC management takes a decision to find application from other MEC system, MEO#1 sends a query to MEF #1 to discover a MEC application instance.

NOTE 3: The decision mechanism in the MEC management is not further specified in the present document. The trigger reason could be e.g. MEC management subscribes to event notification based on 3GPP standards.

5) MEF #1 forwards the query to MEF #2.

6) If the information of MEC application instance, i.e. the list of active MEC application instances, is not registered to MEF #2, MEF #2 forwards the query to MEO #2.

7) MEO #2 responds with the information of MEC application instance(s). If no available MEC application is discovered, void would return.

NOTE 4: In the case where multiple MEC application instances are found, the information of a list of MEC application instances is returned. In this case, the information of the MEC platform associated with each instance might be useful for the selection of MEC application instances. However, for security reasons, the information of MEC platform should be hidden between federated MEC systems. Based on the agreement among federated MEC systems, the information of MEC platform, e.g. available MEC services, can be included in the response.

8) MEF #2 responds with the information of MEC application instance(s) to MEF #1.

9) MEF #1 responds with the information of MEC application instance(s) to MEO #1.

10) MEO #1 responds with the information of MEC application instance(s) to MEP #1 via MEPM #1.

11) MEP #1 responds with the information of MEC application instance(s) to MEC application instance #1.

5.2.2.4 MEC service discovery

MEC service discovery in a MEC federation can be performed when a MEC system of the MEC federation wants to obtain MEC service availability. This process could be triggered in the case where the service consumer (e.g. a MEC application or a MEC platform of a MEC system the service discovery query originates from) needs the specific MEC service that is not available at the collocated MEC platform. The information flow is depicted in Figure 5.2.2.4-1.

NOTE 1: The desired MEC service could be provided by multiple MEC platforms. The service consumer that triggers the discovery may select one service instance.

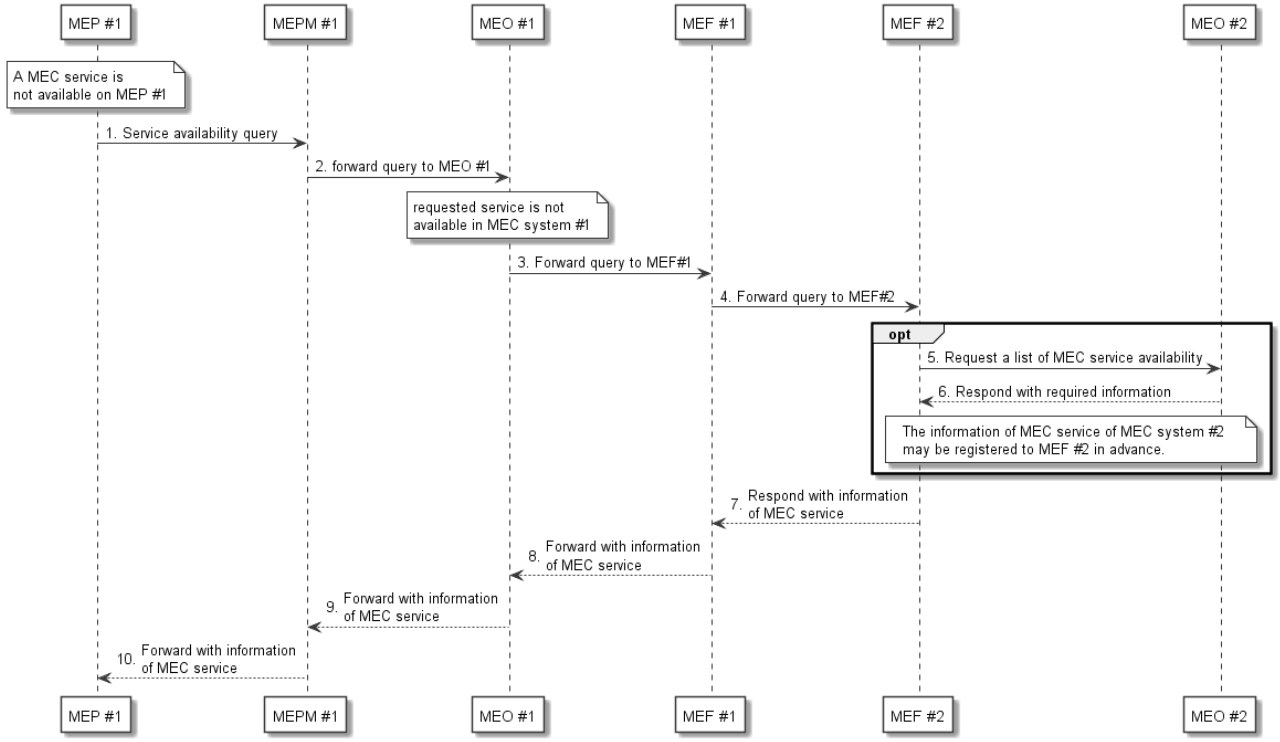


Figure 5.2.2.4-1: Information flow of MEC service discovery

Procedure of MEC service discovery consists of the following steps:

1) MEP#1 sends a request to query the availability of a MEC service or a list of MEC services.

2) MEPM #1 forwards the query to MEO #1.

NOTE 2: How to handle the request between MEP, MEPM and MEO is not further specified in the present document.

3) MEO #1 examines if the requested service(s) are available in MEC system #1. In case where the service(s) are not available, MEO #1 forwards the query to MEF #1. Otherwise, MEO#1 responds with the information of MEC service(s) to MEPM #1.

NOTE 3: The case where the requested service(s) are available in the MEC system is out of scope of the present document.

4) Subject to federation agreements and operator policies, MEF #1 forwards the query to MEF #2.

5) Optionally, MEF #2 forwards the query to MEO #2. Otherwise, go to step 7.

NOTE 4: MEF is assumed to subscribe MEC service availability.

NOTE 5: The case where MEF is not able to subscribe MEC service availability is not considered in the present document.

6) MEO #2 responds with the information of MEC service(s). If the requested MEC service(s) are not available in MEC system #2, MEO #2 returns void.

NOTE 6: For security reasons, the information of the corresponding MEC platform information should be hidden between the federated MEC systems.

7) MEF #2 responds with the information of MEC service(s) to MEF #1.

8) MEF #1 forwards the information of MEC service(s) to MEO #1.

9) MEO #1 forwards the information of MEC service(s) to MEPM #1.

10) MEPM #1 forwards the information of MEC service(s) to MEP #1.

NOTE 7: MEC service discovery using a pub/sub mechanism is not further specified in the present document.

5.2.2.5 Application package management and Application instance lifecycle management

5.2.2.5.1 Introduction

The overall procedures for Application package management and Application instance lifecycle management follow clauses 5.2 and 5.3 of ETSI GS MEC 010-2 [7]. The difference is that all requests are forwarded to the external MEC system in the MEC federation through MEC federator with the information obtained in clauses 5.2.2.1 and 5.2.2.2. This clause describes onboarding an application package and application instantiation, and it can be assumed that the rest of other procedures are also supported through Mfm, Mff interface, and MEC federator, similar to those introduced here. These requests can be triggered at an entity (e.g. OSS) that is connected to application provider.

Application package management:

- On-board application package.
- Query application package information.
- Disable application package.
- Enable application package.
- Delete application package.
- Fetch application package.

Application instance lifecycle management:

- Application instantiation.
- Application termination.
- Application operation.

5.2.2.5.2 On-boarding application package

The message flow of on-boarding application package is used to make application package available to the MEC system in the MEC federation. On-boarding request is triggered by Client (e.g. OSS) of MEC system #1 and forwarded to federated MEC system. The detailed description of this flow is depicted in Figure 5.2.2.5.2-1

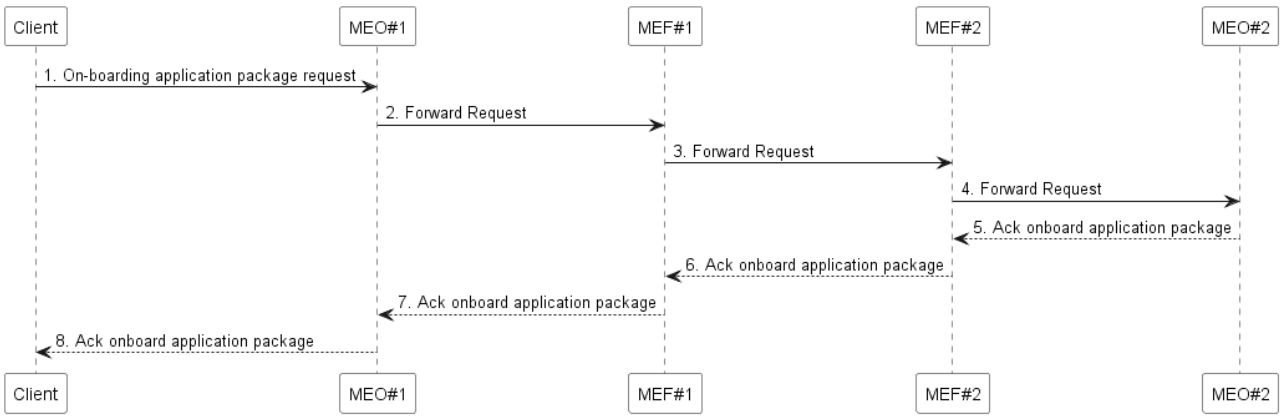


Figure 5.2.2.5.2-1: On-boarding application package in MEC federation

1) Client sends an on-boarding application package request to the MEO #1 with the information for MEC federation. The information may be based on the result of clause 5.2.2.2 MEC system discovery.

2) If information for MEC federation is included, MEO #1 forwards the request to MEC federator #1 and continues with the rest of the steps. Otherwise, follow the procedures of clause 5.2.2 in ETSI GS MEC 010-2 [7].

3) MEC federator #1 forwards the request to MEC federator#2.

4) MEC federator #2 forwards the request to MEO #2 based on included MEC system information.

5) MEO #2 performs the actions described in step 1 of clause 5.2.2 of the ETSI GS MEC 010-2 [7] and MEO #2 returns an acknowledgement to MEC federator #2. The MEC federator #2 keeps track of resources that represent onboarded application packages in the MEC federation (i.e. it is not intended to create any API resource locally in the MEC federator #2).

6) The acknowledgement is forwarded to MEC federator #1. The MEC federator #1 keeps track of resources that represent onboarded application packages in the MEC federation (i.e. it is not intended to create any API resource locally in the MEC federator #1).

7) The acknowledgement is forwarded to MEO #1.

8) The acknowledgement is forwarded to the Client.

5.2.2.5.3 Application Instantiation

The message flow of application instantiation is used to instantiate an application instance in the MEC system.

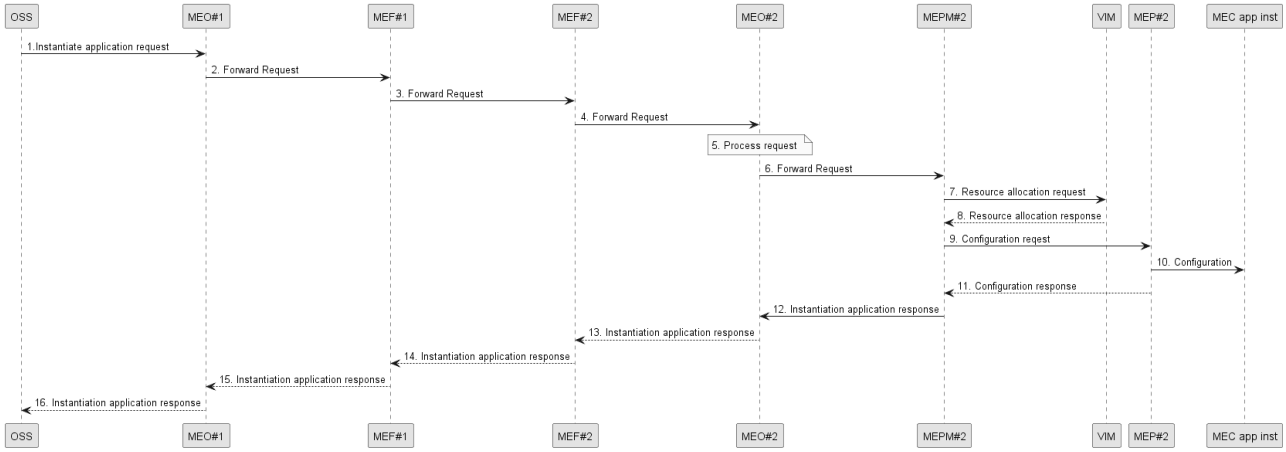


Figure 5.2.2.5.3-1: Application instantiation flow in MEC federation

1) Client sends an instantiate application request to the MEO #1 with the information for MEC federation. The information may be based on the result of clause 5.2.2.2 MEC system discovery.

2) If information for MEC federation is included, MEO #1 forwards the request to MEC federator #1 and continues with the rest of the steps. Otherwise, follow the procedures of clause 5.3.2 in ETSI GS MEC 010-2 [7].

3) MEC federator #1 forwards the request to MEC federator #2.

4) MEC federator#2 forwards the request to MEO #2.

5) MEO #2 checks the application instance configuration data, and authorizes the request and selects the MEC host (and corresponding MEC platform manager). If necessary, MEO #2 performs the actions described in step 3 of clause 5.3.1 of ETSI GS MEC 010-2 [7].

Steps 6 to 12 follow the same procedures as described in steps 4-10 of clause 5.3.1 of ETSI GS MEC 010-2 [7].

13) MEO #2 forwards the response to MEC federator#2 including the results of the instantiation operation and the application instance ID if there is. The MEC federator #2 keeps track of resources that represent instantiated application in the MEC federation (i.e. it is not intended to create any API resource locally in the MEC federator #2).

14) The response is forwarded to MEC federator #1. The MEC federator #1 keeps track of resources that represent instantiated application packages in the MEC federation (i.e. it is not intended to create any API resource locally in the MEC federator #1).

15) The response is forwarded to MEO #1.

16) The response is forwarded to the Client.

5.2.2.6 Providing MEC system-wide MEC application instance information updates to MEF

Figure 5.2.2.6-1 explains the procedure of providing MEC application instance information updates to the MEF within the MEC system. This procedure can be triggered by the instantiation of a MEC application or a new or updated MEC application instance registration.

NOTE 1: The mechanism MEO obtains the information of an MEC application instance registration to the MEC platform is not further specified in the present document.

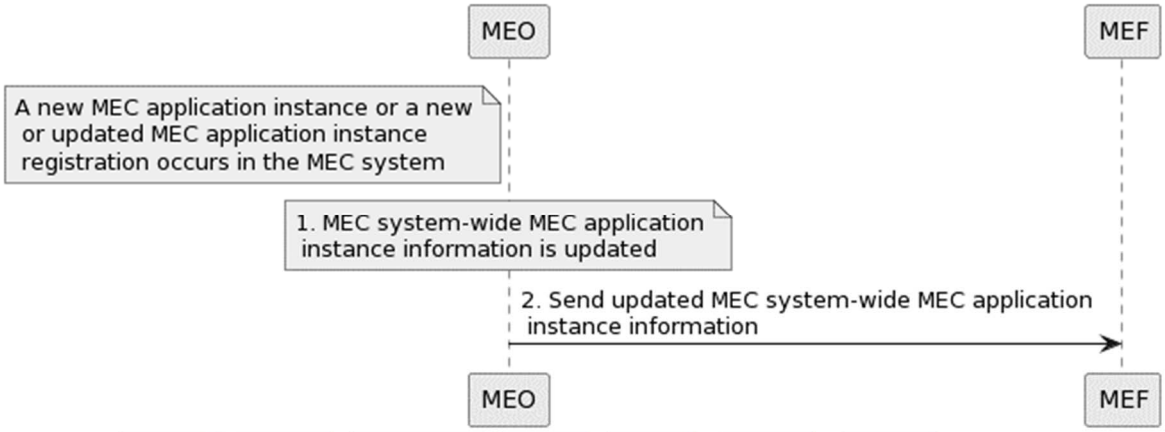


Figure 5.2.2.6-1: Information flow of providing MEC application instance information updates to MEF

The procedure of a MEO providing MEC system-wide MEC application instance information updates to MEF when, within the MEC system, there is a MEC application instantiation by the MEC system, or a MEC application instance registers with a MEC platform or updates its registration information, consists of the following steps:

1) MEC system-wide MEC application instance information is updated.

2) The MEO forwards the updated MEC system-wide MEC application instance information to the MEF.

NOTE 2: What MEC application instance information would be shared in the MEC federation is not further specified in the present document.

NOTE 3: The mechanism MEC application instance information masking is not further specified in the present document.

NOTE 4: The definition of specific authorization, communication periodicity and filtering policies, that are, instead, related to business agreements among federating entities, is outside the scope of the present document.

5.2.2.7 Forwarding MEC system-wide MEC application instance information updates to another MEC system of MEC federation

Figure 5.2.2.7-1 illustrates the inter-MEF communication needed within a MEC federation to share the permissible (e.g. per MEC system authorization policy) MEC system-wide MEC application instance information updates.

Figure 5.2.2.7-1: Information flow of providing permissible MEC system-wide MEC application instance information updates to another MEC system of the MEC federation

The procedure of forwarding MEC system-wide MEC application instance information updates to another MEC system of MEC federation consists of the following steps:

1) MEF 1 sends the updated MEC system-wide MEC application instance information to MEF 2.

NOTE 1: What MEC application instance information is shared in the MEC federation is not further specified in the present document.

NOTE 2: The definition of specific authorization, communication periodicity and filtering policies, that are, instead, related to business agreements among federating entities, is outside the scope of the present document.

5.2.3 REST based subscribe-notify model

5.2.3.1 Subscribing to federation event notifications

To receive a notification of federation event, the Client creates a subscription to the event as depicted in Figure 5.2.3.1-1.

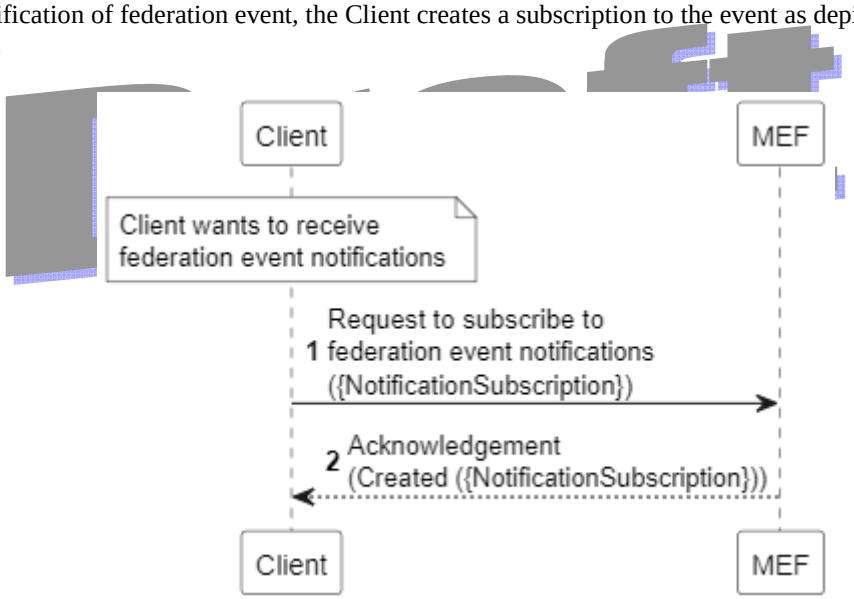


Figure 5.2.3.1-1: Information flow of subscribing to federation event notifications

Subscribing to federation event notification consists of the following steps.

When the Client wants to receive notifications about MEC federation event, it creates a subscription to federation event:

1) The Client sends a request for subscribing to federation event notifications. This request containing the {NotificationSubscription} data structure to the resource representing the federation subscription.

2) Upon success, the MEF generates a unique URI among other subscriptions, which can act as an identifier of subscription (subscriptionId), and returns an acknowledgment message including this URI to the Client.

NOTE: The set of federation events is not further specified in the present document.

5.2.3.2 Receiving notification on expiry of federation event subscription

An expiry time for a federation event subscription may be defined. In case expiry time is used, prior to the expiry, a notification is sent to the subscriber MEO. The scenario where the MEC Federator sends the notification is depicted in Figure 5.2.3.2-1.

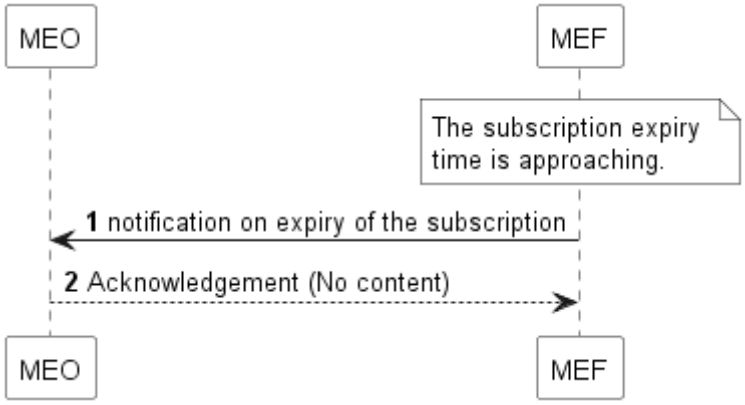


Figure 5.2.3.2-1: Information flow of receiving notification on expiry of federation event subscription

Receiving notification on expiry of federation event subscription consists of the following steps:

1) The MEC Federator sends a notification on expiry of the subscription when the subscription expiry time is approaching.

2) The MEO returns an acknowledgement to the MEC Federator.

5.2.3.3 Updating subscription for federation event notifications

Figure 5.2.3.3-1 presents the scenario where the MEO needs to update an existing subscription for federation event notification.

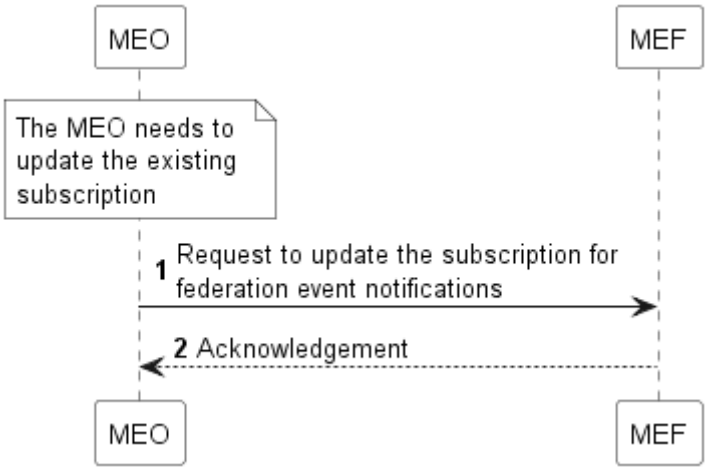


Figure 5.2.3.3-1: Information flow of updating subscription for federation event notifications

Updating subscription for federation event notification consists of the following steps:

1) The MEO sends a request for updating subscription for federation event notifications.

2) The MEC Federator returns an acknowledgement to the MEO.

5.2.3.4 Unsubscribing from federation event notifications

Figure 5.2.3.4-1 shows the scenario of deleting the subscription when the MEO does not want to receive federation event notification after subscribing to the event.

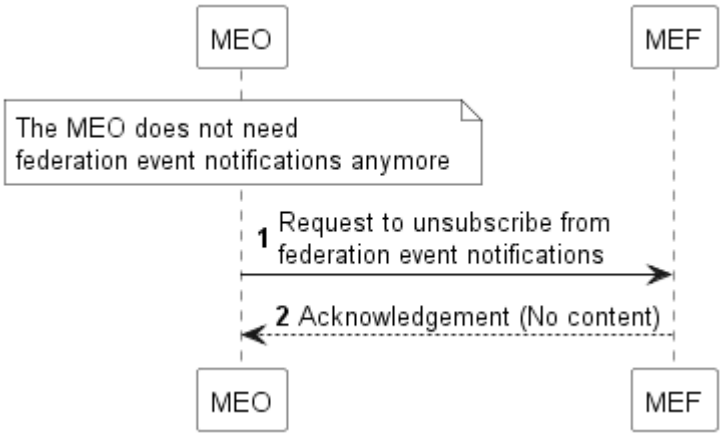


Figure 5.2.3.4-1: Information flow of unsubscribing from federation event notifications

Unsubscribing federation event notification consists of the following steps:

1) The MEO sends a request for unsubscribing from federation event notifications.

2) The MEC Federator returns an acknowledgement to the MEO.

5.2.3.5 Receiving MEC system registration/update notifications

Figure 5.2.3.5-1 presents the scenario of MEC system registration/update notification.

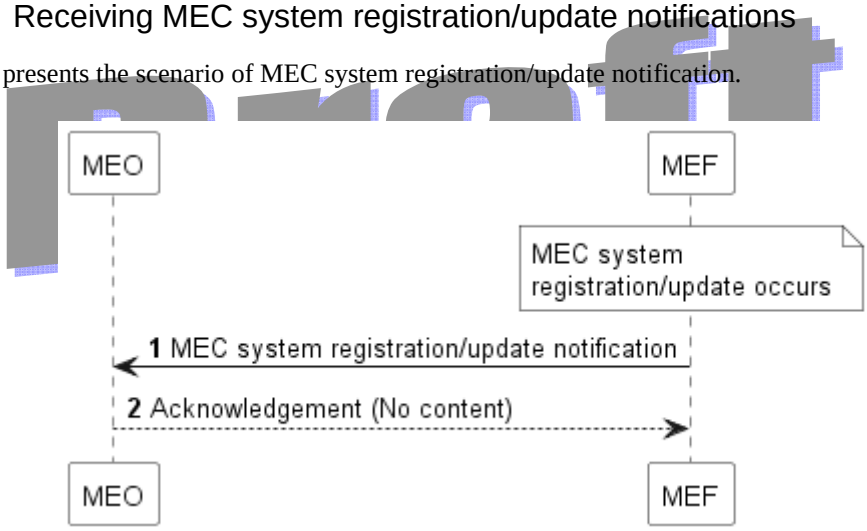


Figure 5.2.3.5-1: Information flow of receiving MEC system registration/update notification

Receiving MEC system registration/update notification consists of the following steps:

1) When MEC system registration/update occurs, the MEC Federator sends a notification of registration/update of other MEC system to the subscriber MEO.

2) The MEO returns an acknowledgement to MEC Federator.

5.2.3.6 Receiving MEC application instance registration/ registration update/ deregistration notifications

Figure 5.2.3.6-1 presents the scenario of MEC application registration/registration update/deregistration notifications.

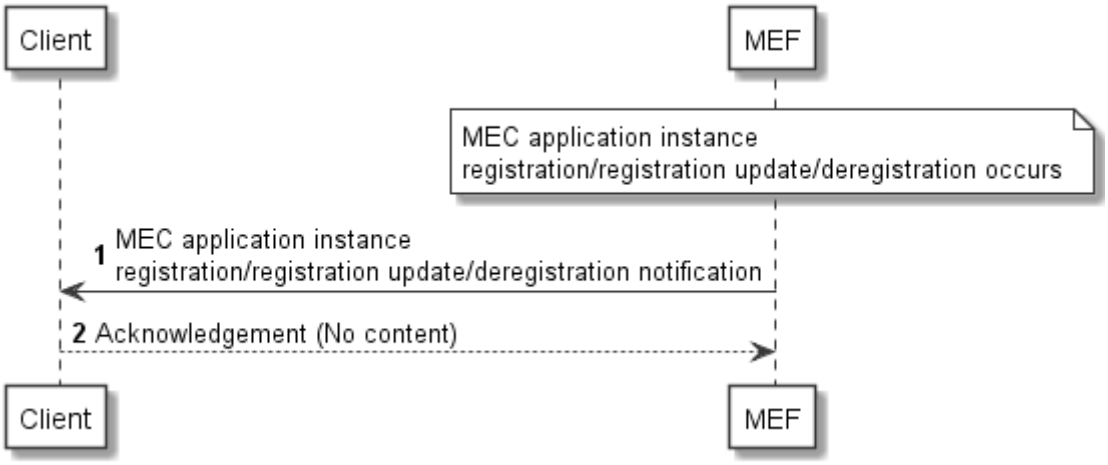


Figure 5.2.3.6-1: Information flow of receiving MEC application instance registration/registration update/deregistration notification

Receiving MEC application instance registration/registration update/deregistration notification consists of the following steps:

1) When MEC application instance registration/registration update/deregistration occurs, the MEF sends a notification of registration/registration update/deregistration of other MEC system to the subscriber Client.

2) The Client returns an acknowledgement to MEF.

NOTE: The mechanism by which the MEF detects the registration/update/deregistration of MEC application instance is not further specified in the present document.

5.2.4 Creation/Update/Removal of MEC federation between two partners

5.2.4.1 Introduction

These procedures provide key functionalities to establish MEC federation between two partners, which are based on the GSMA PRD OPG.04 [5], clause 2.2.1.

Basic functionalities include:

- Create MEC federation between partners;
- Update an already establish MEC federation between partners;
- Remove a MEC federation establishment between partners.

NOTE: In the present document, it is assumed the interaction takes place between the MEFs representing the MEC systems in the federation. MEF (MEC Federator) as defined in ETSI GS MEC 003 [i.4] includes MEC Federation Broker (MEFB) and MEC Federation Manager (MEFM) functionalities.

5.2.4.2 Create MEC Federation between partners

The Create MEC Federation Operation is initiated by the Originating MEC system towards the Partner MEC system to establish a directed federation relationship between the two partners.

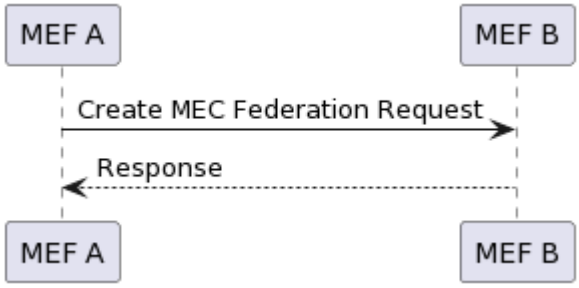


Figure 5.2.4.2-1: Information flow of Create MEC Federation

The message flow for creating a MEC federation relationship is as follows:

1) A MEC Federation create request is sent by the MEF A representing the originating MEC system to the MEF B representing the partner MEC system.

3) The MEF B sends a response to the MEF A to inform about the result of the operation:

   - On success, a 200 OK message is sent.
   - On failure, an appropriate error code is returned.

5.2.4.3 Update MEC Federation between partners

To make an update of a federation partnership the request initiator i.e. the Originating MEC system sends a message to partner OP to update modifiable parameters which were earlier exchanged during the create federation request flow.

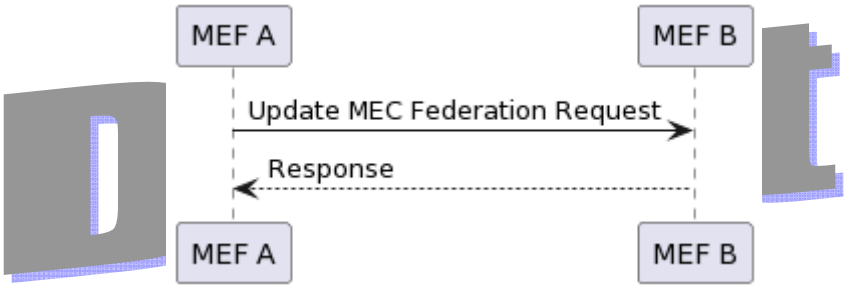


Figure 5.2.4.3-1: Information flow of Update

The message flow for updating a MEC federation relationship is as follows:

1) A MEC Federation update request is sent by the MEF A representing the originating MEC system to the MEF B representing the partner MEC system.

4) The MEF B sends a response to the MEF A to inform about the result of the operation:

   - On success, a 200 OK message is sent.
   - On failure, an appropriate error code is returned.

5.2.4.4 Remove MEC Federation between partners

This procedure is intended to remove existing federation information within a partner MEC system. By Remove Federation Operation, the initiator MEC system sends a request to the partner MEC system to terminate the existing federation configuration.

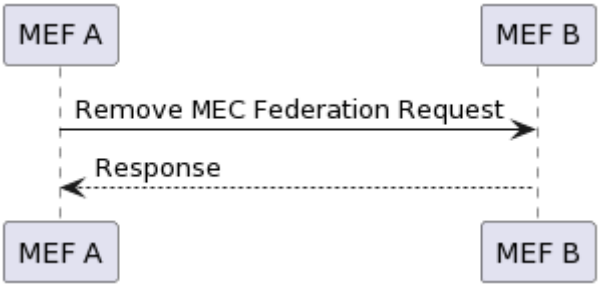


Figure 5.2.4.4-1: Information flow of Deregistration

The message flow for removing an existing MEC federation relationship is as follows:

1) A MEC Federation remove request is sent by MEF A representing the originating MEC system to MEF B representing the partner MEC system.

2) The MEF B sends a response to the MEF A to inform about the result of the operation:

   - On success, a 200 OK message is sent.
   - On failure, an appropriate error code is returned.

6 Data model

6.1 Introduction

The following clauses define the data types common to the APIs specified in the present document.

6.2 Resource data types

6.2.1 Introduction

This clause defines data structures to be used in resource representations.

6.2.2 Type: SystemInfo

This type represents an information provided by the MEC orchestrator as a part of the "Registration of MEC system to the federation" introduced in clause 5.2.2.1.1.

The attributes of the SystemInfo shall follow the indications provided in Table 6.2.2-1.

Table 6.2.2-1: Attributes of SystemInfo

Attribute name	Data type	Cardinality	Description
systemId	SystemId	1	Identifier of the MEC system. Shall be absent in POST request, and present otherwise.
systemName	String	1	The name of the MEC system. This is how the MEC system identifies other MEC systems
systemProvider	String	1	Provider of the MEC system.

6.2.3 Type: SystemInfoUpdate

This type represents an information provided by MEC orchestrator as a part of the "Update of MEC system(s) to the federation" introduced in clause 5.2.2.1.2.

Table 6.2.3-1: Attributes of SystemInfoUpdate

Attribute name	Data type	Cardinality	Description
systemName	String	0..1	The name of the MEC system. This is how the MEC system identifies other MEC systems.
endpoint	EndPointInfo	0..1	Endpoint information (e.g. URI, FQDN, IP address) of MEC federator.
NOTE: At least one attribute shall exist.			

6.2.4 Type: FedServiceInfo

This data type represents the general information of a MEC service in a MEC federation. The attributes of the FedServiceInfo shall follow the indications provided in Table 6.2.4-1.

Table 6.2.4-1: Attributes of FedServiceInfo

Attribute name	Data type	Cardinality	Description
systemId	SystemId	1	Identifier of the MEC system.
mecHostInformation	MecHostInformation	1	Represent the information of the MEC host. See note 1.
serviceInfo	ServiceInfo	1	Represent the information of a MEC service. See note 2.
NOTE 1: The MecHostInformation data type is defined in the Table 6.2.2.17.2-1 of MEC010-2 [7]. NOTE 2: The ServiceInfo data type is defined in the Table 8.1.2.2-1 of ETSI GS MEC 010-2 [7].			

6.2.5 Referenced simple data types

6.2.5.1 Introduction

This clause defines simple data types.

6.2.5.2 Simple data types

The simple data type shall follow the indications provided in Table 6.2.5.2-1.

Table 6.2.5.2-1: Simple data types

Type name	Description
SystemId	String representing identifier of the MEC system. For the uniqueness of the identifier across the federated MEC systems, UUID format [9] is recommended.

6.3 Subscription data types

6.3.1 Introduction

This clause defines data structures for subscriptions.

6.3.2 Type: SystemUpdateNotificationSubscription

This type represents a subscription to the notifications from the MEC federator related to information update of the MEC systems in the MEC federation.

The attributes of the SystemUpdateNotificationSubscription shall follow the indications provided in Table 6.3.2-1.

Table 6.3.2-1: Attributes of SystemUpdateNotificationSubscription

Attribute name	Data type	Cardinality	Description
subscriptionType	String	1	Shall be set to "SystemUpdateNotificationSubscription".
callbackReference	Uri	1	URI selected by the MEC orchestrator to receive notifications on the subscribed MEC system information updates in the MEC federation. This shall be included in both the request and the response.
_links	Structure (inlined)	0..1	Object containing hyperlinks related to the resource. This shall only be included in the HTTP responses.
>self	LinkType	1	Self-referring URI. The URI shall be unique within the MEC Federation Enablement API as it acts as an ID for the subscription (SubscriptionId).
systemId	SystemId	0..N	Identifier of the MEC system. If absent, the subscription should include all MEC systems in the MEC federation.
expiryDeadline	TimeStamp	0..1	The expiration time of the subscription determined by the MEC Federation Enablement Service.

6.3.3 Type: SubscriptionLinkList

This type represents a list of links related to existing subscriptions for subscription service consumers of the MEC federation. This information is provided by the MEC federator when it receives a request to retrieve current subscriptions. The attributes of this data type shall follow the indications provided in Table 6.3.3-1.

Table 6.3.3-1: Attributes of SubscriptionLinkList

Attribute name	Data type	Cardinality	Description
_links	Structure (inlined)	1	List of hyperlinks related to the resource.
>self	LinkType	1	
>subscription	Array(Structure (inlined))	0..N	A link to a subscription.
>>href	URI	1	The URI referring to the subscription.
>>subscriptionType	String	1	Type of the subscription. The string shall be set according to the "subscriptionType" attribute of the associated subscription data type event defined in clause 6.3.

6.4 Notifications data types

6.4.1 Introduction

This clause defines data structures that define notifications.

6.4.2 Type: SystemUpdateNotification

This type represents the information that the MEC federator notifies the subscribed MEC orchestrator about the information update of the MEC systems in the MEC federation.

The attributes of the SystemUpdateNotification shall follow the indications provided in Table 6.4.2-1.

Table 6.4.2-1: Attributes of SystemUpdateNotification

Attribute name	Data type	Cardinality	Description
notificationType	String	1	Shall be set to "SystemUpdateNotification".
updatedSystemInfo	SystemInfo	1..N	Updated information of the MEC system(s) in the MEC federation.
_links	Structure (inlined)	1	Object containing hyperlinks related to the resource.
>subscription	LinkType	1	A link to the related subscription.

6.5 Referenced structured data types

6.5.1 Introduction

This clause defines data structures that are referenced from data structures defined in the previous clauses, but are neither resource representations nor bound to any pub/sub mechanism.

6.5.2 Type: TimeStamp

This type represents a time stamp.

Table 6.5.2-1: Attributes of the TimeStamp

Attribute name	Data type	Cardinality	Description
seconds	Uint32	1	The seconds part of the time. Time is defined as Unix-time since January 1, 1970, 00:00:00 UTC.
nanoSeconds	Uint32	1	The nanoseconds part of the time. Time is defined as Unix-time since January 1, 1970, 00:00:00 UTC.

7 API definition

7.1 Introduction

This clause defines the resources and operations of the MEC federation enablement APIs over the Mfm reference point.

7.2 Global definitions and resource structure

All resource URIs of this API shall have the following root:

{apiRoot}/{apiName}/{apiVersion}

The "apiRoot" includes the scheme ("https"), host and optional port, and an optional prefix string. The "apiName" shall be set to "fed_enablement" and "apiVersion" shall be set to "v1" for the current version of the present document. The "apiRoot", "apiName" and "apiVersion" can be discovered using the service registry. All resource URIs in clauses 7.3 to 7.7 are defined relative to the above root URI.

The API shall support HTTP over TLS as defined in clause 6.22 of ETSI GS MEC 009 [10]

The content format of JSON shall be supported. The JSON format is signalled by the content type "application/json".

This API shall use OAuth 2.0, as defined in clause 6.16 of ETSI GS MEC 009 [10]. This OAuth 2.0 authorization procedure shall occur only on TLS-protected connections.

This API supports additional application-related error information to be provided in the HTTP response when an error occurs. See clause 6.15 of ETSI GS MEC 009 [10] for more information.

Figure 7.2-1 illustrates the resource URI structure of this API.

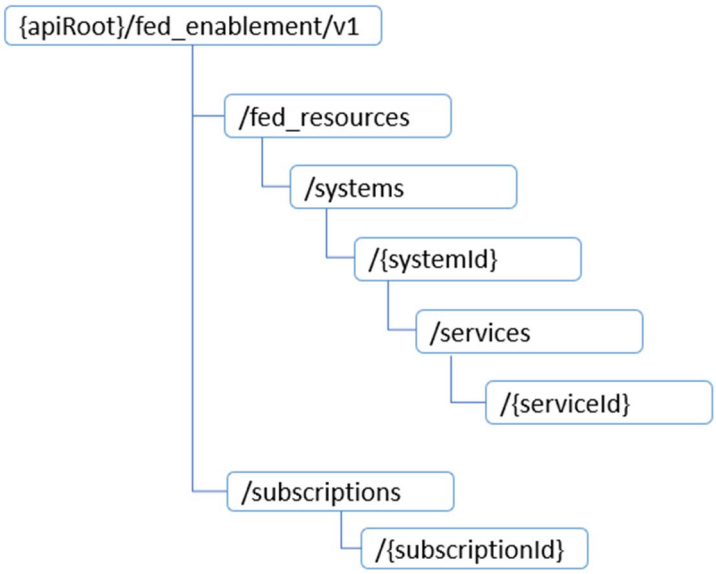


Figure 7.2-1: Resource URI structure of the Federation enablement API

Table 7.2-1 provides an overview of the resources defined by the present document, and the applicable HTTP methods.

Table 7.2-1: Overview of the resources and methods

Resource name	Resource URI	HTTP method	Meaning
A list of MEC system information	/fed_resources/systems	GET	Retrieve a list of system resources (see clause 6 for data model) of federation members.
		POST	Create new system resource for a given MEC system.
Individual MEC system information	/fed_resources/systems/{systemId}	GET	Retrieve the system resource of the federation member with systemId as its system identifier.
		PATCH	Update the system resource for a given MEC system.
		DELETE	Delete the system resource for a given MEC system.
A list of MEC services for federation	/fed_resources/systems/{systemId}/services	GET	Retrieve the information of the federated MEC services hosted by the MEC system associated with the systemId.
Individual MEC service for federation	/fed_resources/systems/{systemId}/services/{serviceId}	GET	Retrieves the information of a specific federated MEC service hosted by the MEC system associated with the systemId and serviceId.

7.3 Resource: A list of system_info

7.3.1 Description

This resource can be used through the MEC system discovery sequence. See clause 5.2 for the details of the sequence.

7.3.2 Resource definition

Resource URI: {apiRoot}/fed_enablement/v1/fed_resources/systems

This resource shall support the resource URI variables defined in Table 7.3.2-1.

Table 7.3.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 7.2

7.3.3 Resource methods

7.3.3.1 GET

The GET method retrieves the information of a list of systems resources of federation members. This method is typically used in the sequence of "MEC system discovery" as described in clause 5.2.2.2. The method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.3.3.1-1 and 7.3.3.1-2.

Table 7.3.3.1-1: URI query parameters supported by the GET method on this resource

Attribute name	Data type	Cardinality	Description
systemId	SystemId	0..N	Identifier of the MEC system.
systemName	String	0..N	The name of the MEC system.
systemProvider	String	0..N	Provider of the MEC system.
NOTE: One or more query parameter attributes may be provided, or none. If multiple query parameters exist, a list including MEC system information which match all query parameters will be returned. If none, a list including all MEC system information registered to the MEF will be returned.			

Table 7.3.3.1-2: Data structures supported by GET request/response on this resource

	Data type	Cardinality	Response codes	Remarks
Request body	n/a			
Response body	SystemInfo	1..N	200 OK	It is used to indicate that the query for retrieving systems resource(s) is successful. Response body containing one or multiple systems resources shall be returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.3.3.2 PUT

Not supported.

7.3.3.3 PATCH

Not supported.

7.3.3.4 POST

The POST method creates the information of systems resources to the MEC federator. This method is typically used in the sequence of "Registration of MEC system to the federation" as described in clause 5.2.2.1. The method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.3.3.4-1 and 7.3.3.4-2.

Table 7.3.3.4-1: URI query parameters supported by the POST method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.3.3.4-2: Data structures supported by POST request/response on this resource

	Data type	Cardinality	Response codes	Remarks
Request body	SystemInfo	1	Entity body in the request contains SystemInfo to be created. The attribute "systemId" shall be absent.	
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	SystemInfo	1	201 Created	It is used to indicate that the systems resource is successfully created. The HTTP response includes a "Location" HTTP header that contains the URI of the created resource.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.3.3.5 DELETE

Not supported.

7.4 Resource: Individual system

7.4.1 Description

This resource can be used for the Registration/Update/Deregistration sequence for manipulating the system resources. In addition, the system resource can also be queried through this resource in the MEC system discovery sequence. See clause 5.2 for the details of sequence.

7.4.2 Resource definition

Resource URI: {apiRoot}/fed_enablement/v1/fed_resources/systems/{systemId}.

This resource shall support the resource URI variables defined in Table 7.4.2-1.

Table 7.4.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 7.2
systemId	Identifier of MEC system.

7.4.3 Resource methods

7.4.3.1 GET

The GET method retrieves the system resource information. This method is typically used in the sequence of "MEC system discovery" as described in clause 5.2.2.2. The method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.4.3.1-1 and 7.4.3.1-2.

Table 7.4.3.1-1: URI query parameters supported by the GET method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.4.3.1-2: Data structures supported by GET request/response on this resource

Request body	**Data type**	**Cardinality**	**Remarks**	
	n/a			
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	SystemInfo	1	200 OK	It is used to indicate that the query for retrieving system resource is successful. Response body containing one system resource shall be returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.4.3.2 PUT

Not supported.

7.4.3.3 PATCH

The PATCH method updates the information of system resources stored in the MEC federator through previous registration. This method is typically used in the sequence of "Update of MEC system to the federation" as described in clause 5.2.2.2. The method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.4.3.3-1 and 7.4.3.3-2.

Table 7.4.3.3-1: URI query parameters supported by the PATCH method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.4.3.3-2: Data structures supported by PATCH request/response on this resource

Request body	Data type	Cardinality	Remarks	
	SystemInfoUpdate	1	It contains attributes to be update.	
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	SystemInfo	1	200 OK	It is used to indicate that the system resource is successfully updated.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.4.3.4 POST

Not supported.

7.4.3.5 DELETE

The DELETE method deletes the information of system resources stored in the MEF. This method is typically used in the sequence of "Deregistration of MEC system to the federation" as described in clause 5.2.2.1.1. The method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.4.3.5-1 and 7.4.3.5-2.

Table 7.4.3.5-1: URI query parameters supported by the DELETE method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.4.3.5-2: Data structures supported by DELETE request/response on this resource

Request body	Data type	Cardinality	Remarks	
	n/a			
Response body	Data type	Cardinality	Response codes	Remarks
	n/a		204 No Content	The operation has been successful. The response body shall be empty.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.5 Resource: subscriptions

7.5.1 Description

This resource represents subscriptions for notifications from the MEC federator to the MEOs.

7.5.2 Resource definition

Resource URI: {apiRoot}/fed_enablement/v1/subscriptions/

This resource shall support the resource URI variables defined in Table 7.5.2-1.

Table 7.5.2-1: Resource URI variables for the resource "subscriptions"

Name	Definition
apiRoot	See clause 7.2

7.5.3 Resource methods

7.5.3.1 GET

The GET method retrieves the information about the subscriptions for the requestor (a MEO). Upon success, the response contains entity body with the list of links to the subscriptions that are present for the requester. This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.5.3.1-1 and 7.5.3.1-2.

Table 7.5.3.1-1: URI query parameters supported by the GET method on this resource

Attribute name	Data type	Cardinality	Description
subscriptionType	String	0..1	Query parameter is used to filter a specific subscription type. Permitted values: • system_update_notification_subscription
systemId	SystemId	0..1	Identifier of the MEC system.

Table 7.5.3.1-2: Data structures supported by the GET request/response on this resource

Request body	Data type	Cardinality	Remarks	
	n/a			
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	SubscriptionLinkList	1	200 OK	Upon success, a response body containing the list of links to requestor's subscriptions is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	406 Not Acceptable	It is used to indicate that the server cannot provide the any of the content formats supported by the client. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.5.3.2 PUT

Not supported.

7.5.3.3 PATCH

Not supported.

7.5.3.4 POST

The POST method is used to create a new subscription to MEF. Upon success, the response contains entity body describing the created subscription. This method shall comply with the request and response data structures, and response codes, as specified in Table 7.5.3.4-1.

Table 7.5.3.4-1: Data structures supported by POST request/response on this resource

	Data type	Cardinality	Response codes	Remarks
Request body	{NotificationSubscription}	1		The request body contains specific subscription data type that is to be created. The option is listed below and defined in clause 6.3.2: • SystemUpdateNotificationSubscription
Response body	{NotificationSubscription}	1	201 Created	It is used to indicate that the subscription resource is successfully created. The HTTP response includes a "Location" HTTP header that contains the URI of the created resource. In the returned NotificationSubscription structure, the created subscription is described using the appropriate data type from the list below and as defined in clause 6.3.2: • SystemUpdateNotificationSubscription
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	406 Not Acceptable	It is used to indicate that the server cannot provide the any of the content formats supported by the client. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	415 Unsupported Media Type	It is used to indicate that the server or the client does not support the content type of the entity body. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error
	ProblemDetails	0..1	422 Unprocessable Entity	It is used to indicate that the server understands the content type of the request entity and that the syntax of the request entity is correct but that the server is unable to process the contained instructions. This error condition can occur if an JSON request body is syntactically correct but semantically incorrect, for example if the target area for the request is considered too large. This error condition can also occur if the capabilities required by the request are not supported. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.5.3.5 DELETE

Not supported.

7.6 Resource: existing subscription

7.6.1 Description

This resource represents subscriptions that the client has created to receive the notifications event of the MEC federation.

7.6.2 Resource definition

Resource URI: {apiRoot}/fed_enablement/v1/subscriptions/{subscriptionId}

This resource shall support the resource URI variables defined in table 7.6.2-1.

Table 7.6.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 7.2
subscriptionId	Refers to created subscription, where the MEF assigns a unique resource name among federation members for this subscription. The resource name can also be used to identify the resource.

7.6.3 Resource methods

7.6.3.1 GET

The GET method retrieves the information about the subscriptions for the requestor. Upon success, the response contains entity body with the data type describing the subscription. This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.6.3.1-1 and 7.6.3.1-2.

Table 7.6.3.1-1: URI query parameters supported by the GET method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.6.3.1-2: Data structures supported by the GET request/response on this resource

Request body	Data type	Cardinality	Remarks	
	n/a			
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	{NotificationSubscription}	1	200 OK	Upon success, a response body containing data type describing the specific subscription data type is returned. The allowed data type for subscription is defined in clauses 6.3.2 and is as follows: • SystemUpdateNotificationSubscription
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	406 Not Acceptable	It is used to indicate that the server cannot provide the any of the content formats supported by the client. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.6.3.2 PUT

The PUT method, in this case, has "replace" semantics. Upon success, the target resource is to be updated with the new Data Type received within the message body of the request. This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.6.3.2-1 and 7.6.3.2-2.

Table 7.6.3.2-1: URI query parameters supported by the PUT method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.6.3.2-2: Data structures supported by PUT request/response on this resource

	Data type	Cardinality	Remarks	
Request body	{NotificationSubscription}	1	The request body contains specific subscription data type that is to be created. The option is listed below and defined in clause 6.3.2: • SystemUpdateNotificationSubscription	
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	{NotificationSubscription}	1	200 OK	Upon success, a response cody containing data type describing the updated subscription is returned. The allowed subscription data type is described using the appropriate data type from the list below and as defined in clause 6.3.2: • SystemUpdateNotificationSubscription
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	406 Not Acceptable	It is used to indicate that the server cannot provide the any of the content formats supported by the client. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	412 Precondition Failed	It is used when a condition has failed during conditional requests. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	422 Unprocessable Entity	It is used to indicate that the server understands the content type of the request entity and that the syntax of the request entity is correct but that the server is unable to process the contained instructions. This error condition can occur if a JSON request body is syntactically correct but semantically incorrect, for example if the target area for the request is considered too large. This error condition can also occur if the capabilities required by the request are not supported. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.6.3.3 PATCH

Not supported.

7.6.3.4 POST

Not supported.

7.6.3.5 DELETE

The DELETE method is used to cancel the existing subscription. Cancellation can be made by deleting the resource that represents existing subscription. This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.6.3.5-1 and 7.6.3.5-2.

Table 7.6.3.5-1: URI query parameters supported by the DELETE method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.6.3.5-2: Data structures supported by DELETE request/response on this resource

Request body	Data type	Cardinality	Remarks	
	n/a			
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	n/a	0	204 No Content	Upon success, a response 204 No Content without any response body is returned.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.7 Resource: A list of MEC services for federation on the Mfm and Mff reference points

7.7.1 Description

This resource is used to represent a list of MEC services for MEC federation.

7.7.2 Resource definition

Resource URI: {apiRoot}/fed_enablement/v1/fed_resources/systems/{systemId}/services

This resource shall support the resource URI variables defined in Table 7.7.2-1.

Table 7.7.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 7.2.
systemId	Identifier of MEC system.

7.7.3 Resource methods

7.7.3.1 GET

The GET method retrieves the information of all MEC services hosted by the MEC system associated with the systemId. Upon success, the response shall contain entity body with the list of FedServiceInfo that are present for the requester. This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.7.3.1-1 and 7.7.3.1-2.

Table 7.7.3.1-1: URI query parameters supported by the GET method on this resource

Attribute name	Data type	Cardinality	Description
seInstancerId	SerInstanceId	0..1	Indicator of a MEC service offered for MEC federation.
serName	SerName	0..1	The name of the service.
serCategory	CategoryRef	0..1	A Category reference. For the serCategory, the example values include: 1. "RNI" 2. "Location" 3. "Bandwidth Management". See note.
NOTE: If using this parameter in MEC federation, it is necessary to define a common setting policy among federation members in advance.			

Table 7.7.3.1-2: Data structures supported by the GET request/response on this resource

Request body	Data type	Cardinality	Remarks	
	n/a			
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	FedServiceInfo	0..N	200 OK	Upon success, a response body containing an array of FedServiceInfo is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	406 Not Acceptable	It is used to indicate that the server cannot provide the any of the content formats supported by the client. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	414 URI Too Long	It is used to indicate that the server is refusing to process the request because the request URI is longer than the server is willing or able to process.

7.7.3.2 PUT

Not supported.

7.7.3.3 PATCH

Not supported.

7.7.3.4 POST

Not supported.

7.7.3.5 DELETE

Not supported.

7.8 Resource: Individual MEC service for federation on the Mfm and Mff reference points

7.8.1 Description

This resource is used to represent individual MEC service for MEC federation.

7.8.2 Resource definition

Resource URI: {apiRoot}/fed_enablement/v1/systems/{systemId}/services/{serviceId}

This resource shall support the resource URI variables defined in Table 7.8.2-1.

Table 7.8.2-1: Resource URI variables for the resource

Name	Definition
apiRoot	See clause 7.2
systemId	Identifier of MEC system.
serviceId	Indicator of a MEC service for MEC federation.

7.8.3 Resource methods

7.8.3.1 GET

The GET method retrieves the information of a specific MEC service hosted by the MEC system associated with the systemId and serviceId. Upon success, the response shall contain entity body with the FedServiceInfo of the MEC service that is present for the requester. This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in Tables 7.8.3.1-1 and 7.8.3.1-2.

Table 7.8.3.1-1 URI query parameters supported by the GET method on this resource

Attribute name	Data type	Cardinality	Description
n/a			

Table 7.8.3.1-2: Data structures supported by the GET request/response on this resource

Request body	Data type	Cardinality	Remarks	
	n/a			
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	FedServiceInfo	1	200 OK	Upon success, a response body containing a FedServiceInfo is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed in the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	401 Unauthorized	It is used when the client did not submit credentials. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	1	403 Forbidden	The operation is not allowed given the current status of the resource. More information shall be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	It is used when a client provided a URI that cannot be mapped to a valid resource URI. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	406 Not Acceptable	It is used to indicate that the server cannot provide the any of the content formats supported by the client. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.8.3.2 PUT

Not supported.

7.8.3.3 PATCH

Not supported.

7.8.3.4 POST

Not supported.

7.8.3.5 DELETE

Not supported.

Annex A (informative): Enabling MEC App providers to access MEC federation services

A.1 Introduction

According to ETSI GR MEC 035 [i.2], a "*MEC federation is a federated model of MEC systems enabling shared usage of MEC services and applications*". This definition is based on the need to standardize features taking into account the GSMA Operator Platform Group (OPG) OP Telco Edge requirements contained in [i.1].

In further detail, in [i.1], a number of interfaces are described, among which:

1) the NBI connecting an application provider to an OP instance; and

2) the E/WBI connecting two OP instances.

Definitions of these two particular interfaces, as documented in [i.1], need to be considered by ETSI MEC when specifying the federation enablement services. Correspondences of NBI and E/WBI interfaces to MEC federation functional entities and reference points are shown in the exemplary Figure A-1, where it is assumed that each OP instance refers to a federated MEC system and there is no MEFB entity present.

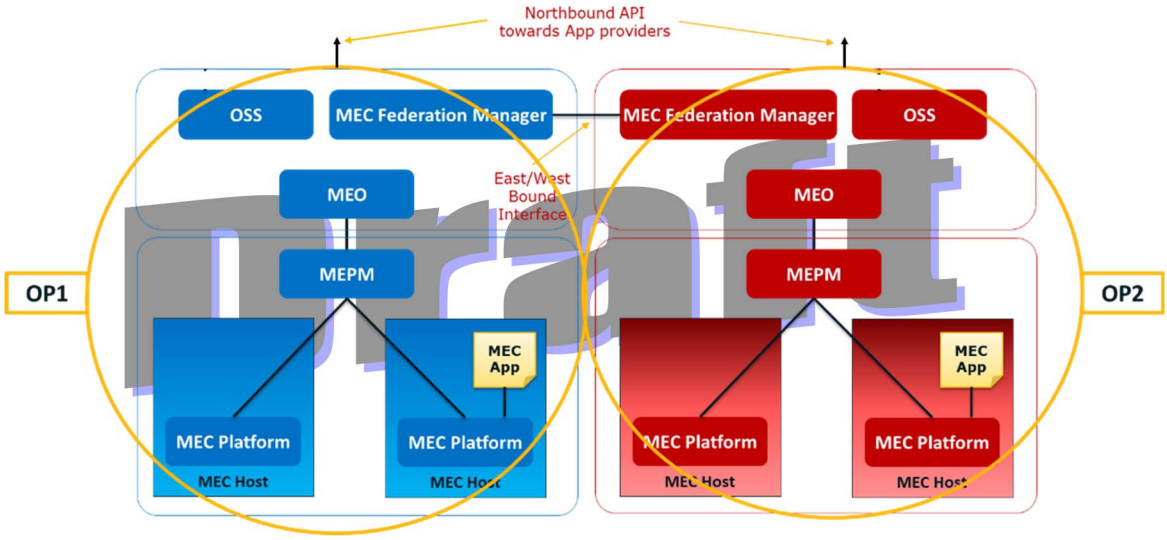


Figure A-1: MEC federation and exemplary correspondence to GSMA OP interfaces NBI and E/WBI

NOTE: In the context of MEC federation, in the view of an implementation with 5G networks, both ETSI MEC and 3GPP standards should be considered, in a harmonized way.

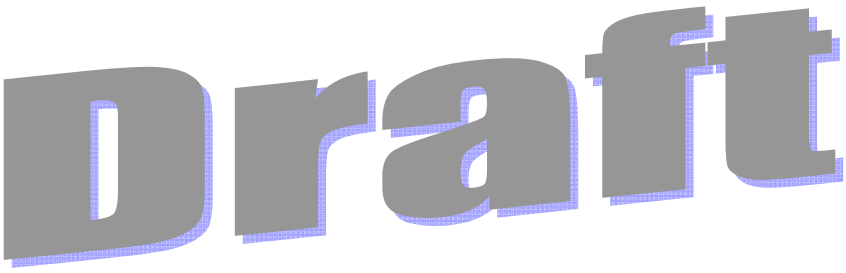
In this scenario, enablement of MEC App providers to access the MEC federation in an interoperable manner is needed. This would imply the possibility for MEC App providers to offer MEC applications instantiated across the MEC federation, benefiting from all the resources (including MEC services) shared in the MEC federation, according to the authorization policies defined by the multi-party agreements among the business entities forming the MEC federation. In detail, what is needed is enabling an interoperable interface to MEC App providers accessing a MEC federation, with a list of available MEC services. Privacy and security of each MEC system the MEC federation is composed of needs to be ensured, considering that each MEC system which is part of the MEC federation has its own policy and need to expose information, customize access policies and capability to offer customized tools to MEC App providers. The E/WBI interface (along with the NBI interface) is an enabler for the access of MEC federation services by MEC App providers. In particular, the E/WBI enables MEC systems forming a MEC federation to be informed about the MEC App provider's requirement, information initially provided by the MEC App provider to a MEC system of a MEC federation through the NBI interface.

Annex B (informative): Complementary material for API utilization

To complement the definitions for each method and resource defined in the interface clauses of the present document, ETSI MEC ISG is providing for the Application Mobility Service API a supplementary description file compliant to the OpenAPI Specification [i.7].

In case of discrepancies between the supplementary files and the related data structure definitions in the present document, the data structure definitions take precedence.

The supplementary files, relating to the present document, are located at https://forge.etsi.org/rep/mec/gs040-fed-enablement-api.



Annex C (informative): Data models of EWBI defined by GSMA OPG

C.1 Introduction

This clause captures the related data models as defined in GSMA PRD OPG.0x [x].

C.2 Interface Management API

The following table summarizes the data model supported by the interface management API as defined in clause 3.1.1.7 of GSMA PRD OPG.04 [5]. It is applicable for Mff Interface Management API in clause 7.7.2.

Table C.2-1: Data model for Interface Management APIs

Data Type	Description	Remarks
Structured Data Types		
offeredAvailabilityZones	Information about the availability zones which the Partner MEC System offers to the Originating System	Defined in clause 3.1.1.7.2.1 of [5]
availabilityZone	The equivalent of an Availability Zone on Public Cloud	Defined in clause 3.1.1.7.2.2 of [5]
edgeDiscoveryServiceEndPoint	End Point information of the Edge Discovery service	Defined in clause 3.1.1.7.2.3 of [5]
mncChangeInfo	The network code update information to notify change in the supported network codes by the Partner MEC System	Defined in clause 3.1.1.7.2.4 of [5]
zoneStatus	Availability status of zones	Defined in clause 3.1.1.7.2.7 of [5]
Enumeration		
objectType	The attribute being updated by the Partner MEC System on existing federation	Defined in clause 3.1.1.7.3.2 of [5]
operationType		Defined in clause 3.1.1.7.3.3 of [5]
Status		Defined in clause 3.1.1.7.3.4 of [5]

History

Document history		
V3.1.2	2022-12	The first early draft based on ETSI GS MEC 040 v3.0.15.
V3.1.3	2022-12	Implements documents MEC(22)000432r2 and MEC(22)000481r2.
V3.1.4	2023-03	Implements documents MEC(22)000613r1, MEC(22)000612r5, MEC(23)000079r1, and MEC(23)000120r1.
V3.1.5	2023-08	Implements documents MEC(23)000126r2, MEC(23)000226r2, and MEC(23)000227r1.
V3.1.6	2023-11	Fixes format error on editor's note in Clause 7.2.1. Implements documents MEC(23)000098r7 and MEC(23)000345.
V3.1.7	2023-12	Implements document MEC(23)000506r1.
V3.1.8	2023-12	Implements documents MEC(23)000573r1 and MEC(23)000580. Fixes format errors.
V3.1.8	January 2024	Clean-up done by ***editHelp!*** E-mail: mailto:edithelp@etsi.org
V3.1.9	2024-01	Implements a document MEC(24)000045.
V3.1.10	2024-01	Final draft V3.1.10 is similar to Stable draft V3.1.9 and ready to be sent to Remote Consensus for MEC review.