ETSI GS MEC 016 V3.1.1 (2024-03)

Multi-access Edge Computing (MEC); Device application interface

Disclaimer

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.

Reference
RGS/MEC-0016v311 DevAppInterfac

Keywords
API, MEC

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 contains the API definition for the lifecycle management of user applications over the Mx2 reference point between the device application and the User Application LifeCycle Management Proxy (UALCMP) in the MEC system.

The present document covers the following lifecycle management operations: user application look-up, instantiation and termination. In addition, a mechanism is specified for the exchange of lifecycle management related information between the MEC system and the device application.

The intended key audience of the present document are the application developers for the MEC system, since this API provides them with a method to manage their applications.

NOTE: User application mobility related lifecycle management operations are not covered by the present document.

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

[2] Void.

[3] Void.

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

[5] IETF RFC 6750: "The OAuth 2.0 Authorization Framework: Bearer Token Usage".

[6] IETF RFC 4776: "Dynamic Host Configuration Protocol (DHCPv4 and DHCPv6) Option for Civic Addresses Configuration Information".

[7] ISO 3166: "Codes for the representation of names of countries and their subdivisions".

[8] IETF RFC 7946: "The GeoJSON Format".

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

[10] IETF RFC 8705: "OAuth 2.0 Mutual-TLS Client Authentication and Certificate-Bound Access Tokens".

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

[i.2] ETSI GS MEC 002: "Multi-access Edge Computing (MEC); Use Cases and Requirements".

[i.3] OpenAPI™ Specification.

NOTE: OpenAPI Specification and OpenAPI Initiative and their respective logos, are trademarks of the Linux Foundation.

[i.4] Void.

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GR MEC 001 [i.1] and the following apply:

user application lifecycle management proxy: system level functional element that allows specific and authorized requests from the device application for the user application lifecycle management

3.2 Symbols

Void.

3.3 Abbreviations

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

AA	Authentication and Authorization
TLS	Transport Layer Security
UALCMP	User Application LifeCycle Management Proxy

4 Overview

The present document specifies the API for the Mx2 reference point to support the corresponding requirements defined for the Multi-access Edge Computing in ETSI GS MEC 002 [i.2].

Clause 5 describes how the API can be used by the device application and by the MEC system. It describes the information flows for the procedures over the Mx2 reference point. It also describes the relevant API endpoints such as the UALCMP and the API gateway.

The information that is exchanged over the Mx2 reference point is described in clause 6, providing detailed description of all information elements available on that interface.

Clause 7 describes the actual API, providing detailed information how the information elements map into the RESTful API design of the interface.

Clause 8 describes the authentication, authorization and access control for the API.

5 Description of the service (informative)

5.1 Sequence diagrams

5.1.1 Introduction

The following clauses describe how the device application interacts with the UALCMP over the Mx2 reference point. The sequence diagrams that are relevant for the API are presented.

The device application presents the access token to the UALCMP with every request in order to assert that it is allowed to access the resource with the particular method it invokes. The access token is included in the "Authorization" request header field as a bearer token according to IETF RFC 6750 [5].

5.1.2 User application look-up

The user application look-up is the procedure for requesting the list of available user applications in the MEC system to the requesting device application. The user application look-up procedure is illustrated in figure 5.1.2-1.

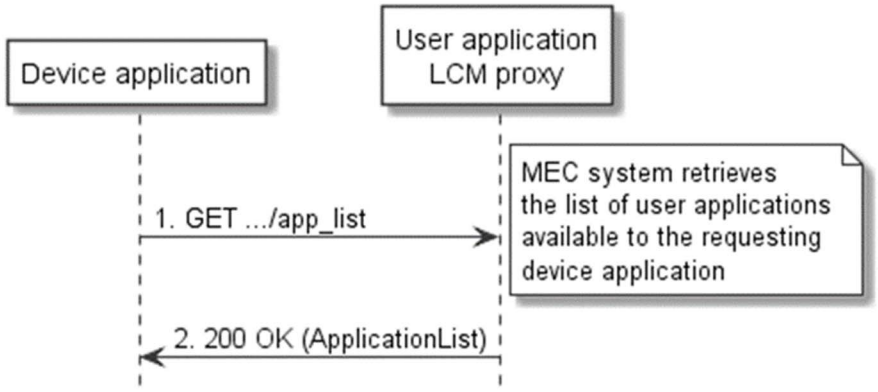

Figure 5.1.2-1: User application look-up

1) The device application submits the GET request to the UALCMP. The UALCMP authorizes the request from device application. The MEC system retrieves the list of user applications available to the requesting device application.

2) The UALCMP returns the 200 OK response to the device application, with the message body containing the data structure for the list of available user applications.

5.1.3 Application context create

The application context create is the procedure to request either to join with an available user application or to instantiate a new user application. The application context create procedure is illustrated in figure 5.1.3-1.

As part of the user application instantiation, the MEC system will create an associated application context that the MEC system maintains for the lifetime of the user application. The application context contains information specific to the instance(s) of the user application such as unique identifiers within the MEC system and the address(es) (URI) provided for clients that are external to the MEC system to interact with the user application.

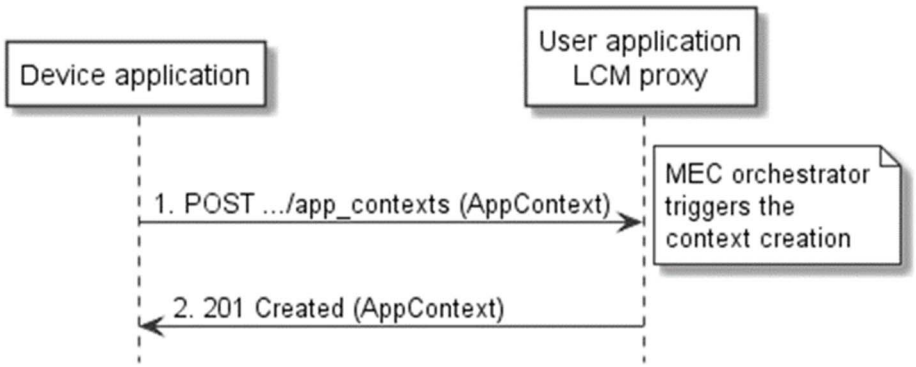

Figure 5.1.3-1: Application context create

1) The device application submits the POST request to the UALCMP. The message body contains the data structure for the application context to be created.

2) The UALCMP authorizes the request from the device application. The request is forwarded to the OSS. The OSS makes the decision on granting the context creation request. The MEC orchestrator triggers the creation of the application context in the MEC system.

3) The UALCMP returns the 201 Created response to the device application with the message body containing the data structure of the created application context, which includes the address(es) (reference URIs) provided for clients that are external to the MEC system to interact with the user application. The response message header contains the address of the resource relating to the application instance context created and maintained by the MEC system.

5.1.4 Application context delete

The application context delete is a procedure in which the device application requests the deletion of the application context. The application context delete procedure is illustrated in figure 5.1.4-1.

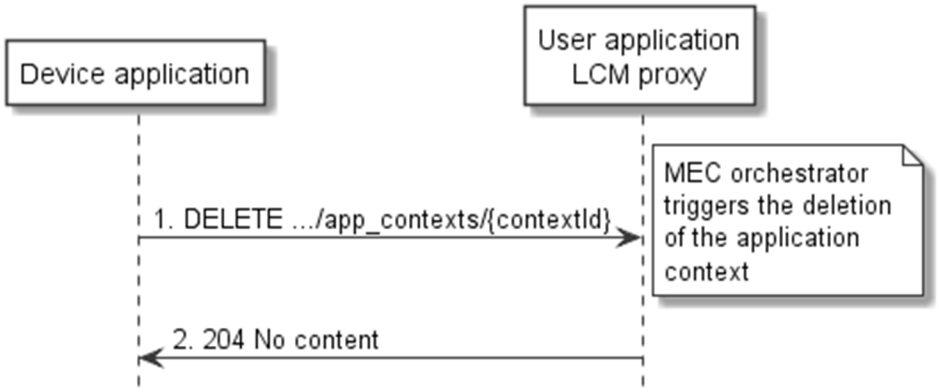

Figure 5.1.4-1: Application context delete

1) The device application submits the DELETE request to the UALCMP for the application context to be deleted.

2) The UALCMP authorizes the request from device application. The request is forwarded to the OSS. The OSS makes the decision on granting the deletion. The MEC orchestrator triggers the deletion of the application context, including deletion of the resource maintained by the MEC system that represents it.

3) The UALCMP returns "204 No content" response.

5.1.5 Application context update

The UALCMP is provided with an update of the application context. The procedure is illustrated in figure 5.1.5-1.

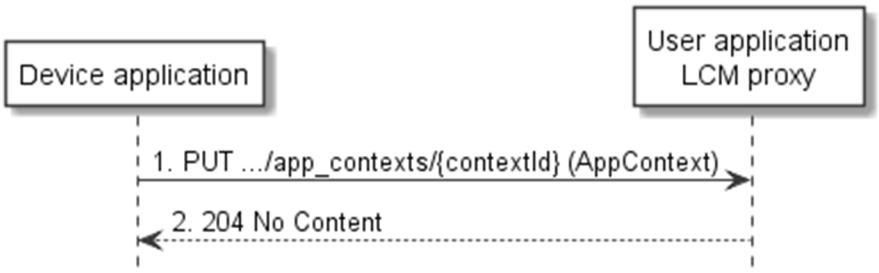

Figure 5.1.5-1: Application context update

1) The device application updates a specific application context by sending a PUT request to the resource within the MEC system that represents it, with the message body containing the modified data structure of AppContext in which only the callback reference and/or application location constraints, may be updated.

2) The UALCMP returns a "204 No Content" response.

5.1.6 Receiving notification events

Figure 5.1.6-1 presents the scenario where the UALCMP sends notification events to the device application.

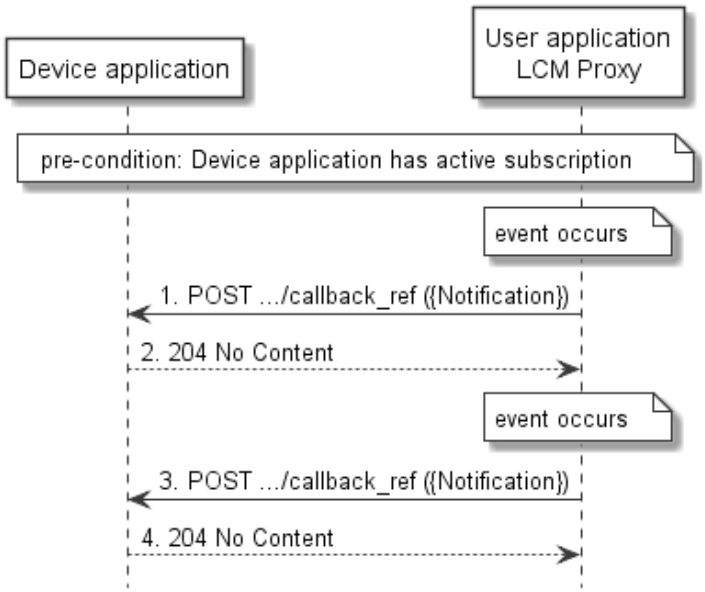

Figure 5.1.6-1: Flow of receiving notification events

Receiving notification events, as illustrated in figure 5.1.6-1, consists of the following steps:

1) The UALCMP sends a POST message to the callback reference address provided by the device application as part of application context creation, with the message body containing the {Notification} data structure. The variable {Notification} is replaced with the data type specified for different notification events as specified in clauses 6.4.2 through 6.4.4, indicating for instance a modification to the address of the user application instance.

2) The device application sends a "204 No Content" response to the UALCMP.

5.1.7 Location constraint look-up

The location constraint look-up is the procedure for a device application to obtain a list of locations available for instantiation of a specific user application in the MEC system. The location constraint look-up procedure is illustrated in figure 5.1.7-1.

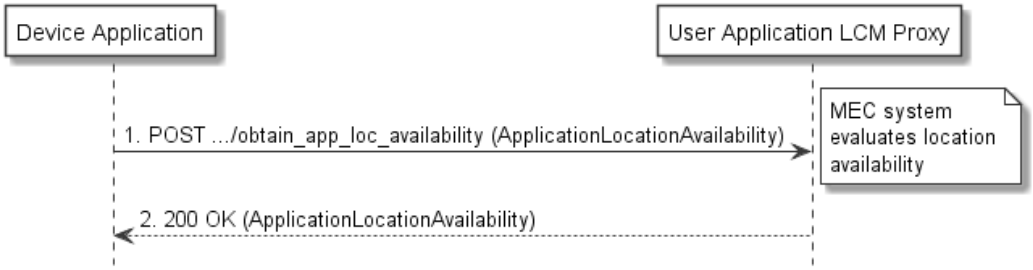

Figure 5.1.7-1: User application location constraint look-up

1) The device application submits a POST request to the UALCMP. The message body contains the data structure identifying the application. The UALCMP authorizes the request from the device application. The locations are then evaluated by the MEC system.

2) The UALCMP returns the 200 OK response to the device application, with the message body containing the data structure for the list of available user application locations as evaluated by the MEC system.

5.2 Considerations on API endpoints

The User Application LifeCycle Management Proxy (UALCMP) exposes the Mx2 APIs to the device application. Using these APIs, a device application can request the application lifecycle management operations as outlined in clause 5.1.

In the context of the present document, the device has both a device application and a client application running. The device application interacts with the UALCMP; the client application interacts with a MEC application on the MEC host, and a MEC Operator may employ an API gateway functionality for API requests from such client applications.

The API gateway is an operator managed entity that is a logical single point of entry for all API requests, including MEC system APIs. It can encompass multiple functions including, but not limited to, security, access control, load balancing/routing, protocol translation, as well monitoring API requests for auditing, billing, and charging. Depending on operator deployments, the API gateway functionality can accompany the MEC platform, and be used to front API requests from client applications to access MEC services.

6 Data model

6.1 Introduction

The following clauses provide the description of the data model.

6.2 Resource data types

6.2.1 Introduction

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

6.2.2 Type: ApplicationList

This type represents the information on available applications. The device application can acquire this information by user application look-up procedure described in clause 5.1.2.

The elements of the ApplicationList shall follow the notations provided in table 6.2.2-1.

Table 6.2.2-1: Definition of type ApplicationList

Attribute name	Data type	Cardinality	Description
appList	Array (Structure (inlined))	0..N	List of user applications available to the device application. As defined below.
>appInfo	Structure (inlined)	1	
>>appDId	String	1	Identifier of this MEC application descriptor. It is equivalent to the appDId defined in clause 6.2.1.2 of ETSI GS MEC 010-2 [1]. This attribute shall be globally unique.
>>appName	String	1	Name of the MEC application. The length of the value shall not exceed 32 characters.
>>appProvider	String	1	Provider of the MEC application. The length of the value shall not exceed 32 characters.
>>appSoftVersion	String	1	Software version of the MEC application. The length of the value shall not exceed 32 characters.
>>appDVersion	String	1	Identifies the version of the application descriptor. It is equivalent to the appDVersion defined in clause 6.2.1.2 of ETSI GS MEC 010-2 [1].
>>appDescription	String	1	Human readable description of the MEC application (see note 2).
>>appLocation	LocationConstraints	0..N	Identifies the locations of the MEC application.
>>appCharcs	Structure (inlined)	0..1	Characteristics of the application. As defined below. The application characteristics relate to the system resources consumed by the application. A device application can use this information e.g. for estimating the cost of use of the application or for the expected user experience.
>>>memory	uint32	0..1	The maximum size in Mbytes of the memory resource expected to be used by the MEC application instance in the MEC system.
>>>storage	uint32	0..1	The maximum size in Mbytes of the storage resource expected to be used by the MEC application instance in the MEC system.
>>>latency	uint32	0..1	The target round trip time in milliseconds supported by the MEC system for the MEC application instance.
>>>bandwidth	uint32	0..1	The required connection bandwidth in kbit/s for the use of the MEC application instance.
>>>serviceCont	Enum	0..1	Required service continuity mode for this application. Permitted values: 0 = SERVICE_CONTINUITY_NOT_REQUIRED. 1 = SERVICE_CONTINUITY_REQUIRED.
>vendorSpecificExt	Structure (inlined)	0..1	Extension for vendor specific information (see note 1).
>>vendorId	String	1	Vendor identifier. The length of the value shall not exceed 32 characters. The rest of the structure of vendor specific extension is not defined.
NOTE 1: The vendor specific extension allows submitting information on the application lists that have been made available to the device application of the corresponding vendor. NOTE 2: The language support may be limited. The length of the value shall not exceed 128 characters.			

6.2.3 Type: AppContext

This type represents the information on application context created by the MEC system.

The elements of the AppContext shall follow the notations provided in table 6.2.3-1.

Table 6.2.3-1: Definition of type AppContext

Attribute name	Data type	Cardinality	Description
contextId	String	0..1	Uniquely identifies the application context in the MEC system. Assigned by the MEC system and shall be present other than in a create request. The length of the value shall not exceed 32 characters.
associateDevAppId	String	1	Uniquely identifies the device application. The length of the value shall not exceed 32 characters.
callbackReference	URI	0..1	URI assigned by the device application to receive application lifecycle related notifications. Inclusion in the request implies the client supports the pub/sub mechanism and is capable of receiving notifications. This endpoint shall be maintained for the lifetime of the application context.
appLocationUpdates	Boolean	0..1	Used by the device application to request to receive notifications at the callbackReference URI relating to location availability for user application instantiation.
appAutoInstantiation	Boolean	0..1	Provides indication to the MEC system that instantiation of the requested application is desired should a requested appLocation become available that was not at the time of the request.
appInfo	Structure (inlined)	1	
>appDId	String	0..1	Identifier of this MEC application descriptor. This attribute shall be globally unique. It is equivalent to the appDId defined in clause 6.2.1.2 of ETSI GS MEC 010-2 [1]. It shall be present if the application is one in the ApplicationList.
>appName	String	1	Name of the MEC application. The length of the value shall not exceed 32 characters.
>appProvider	String	1	Provider of the MEC application. The length of the value shall not exceed 32 characters.
>appSoftVersion	String	0..1	Software version of the MEC application. The length of the value shall not exceed 32 characters.
>appDVersion	String	1	Identifies the version of the application descriptor. It is equivalent to the appDVersion defined in clause 6.2.1.2 of ETSI GS MEC 010-2 [1].
>appDescription	String	0..1	Human readable description of the MEC application. The length of the value shall not exceed 128 characters.
>userAppInstanceInfo	array (Structure inlined)	1..N	List of user application instance information.
>>appInstanceId	String	0..1	Identifier of the user application instance. It shall only be included in the response.
>>referenceURI	URI	0..1	Address of the user application instance. It shall only be included in the response.
>>appLocation	LocationConstraints	0..1	Location of the user application instance. For a user application not provided by the requesting device application it shall match one of the appLocations in ApplicationList.

Attribute name	Data type	Cardinality	Description
>appPackageSource	URI	0..1	URI of the application package. Included in the request if the application is not one in the ApplicationList. appPackageSource enables on-boarding of the application package into the MEC system. The application package shall comply with the definitions in clause 6.2.1.2 of ETSI GS MEC 010-2 [1].
NOTE 1: If a value of the attribute is included in the request, the same value shall be included in the response. NOTE 2: The design of the current operation with callback reference assumes no web proxy between the entity that originates the notification and the entity that receives it. NOTE 3: The language support for the application description may be limited. NOTE 4: Attribute appLocationUpdates and appAutoInstantiation shall not both be set to TRUE.			

6.2.4 Type: ApplicationLocationAvailability

This type represents the information on locations available instantiation of new user application instances.

The elements of the ApplicationLocationAvailability shall follow the notations provided in table 6.2.4-1.

Table 6.2.4-1: Definition of type ApplicationLocationAvailability

Attribute name	Data type	Cardinality	Description
associateDevAppId	String	1	Uniquely identifies the device application. The length of the value shall not exceed 32 characters.
appInfo	Structure (inlined)	1	
>appName	String	1	Name of the MEC application. The length of the value shall not exceed 32 characters.
>appProvider	String	1	Provider of the MEC application. The length of the value shall not exceed 32 characters.
>appSoftVersion	String	0..1	Software version of the MEC application. The length of the value shall not exceed 32 characters.
>appDVersion	String	1	Identifies the version of the application descriptor. It is equivalent to the appDVersion defined in clause 6.2.1.2 of ETSI GS MEC 010-2 [1].
>appDescription	String	0..1	Human readable description of the MEC application. The length of the value shall not exceed 128 characters.
>availableLocations	array (Structure (inline))	0..N	MEC application location constraints.
>>appLocation	LocationConstraints	0..1	Shall only be included in the response, where it indicates a location constraint available in the MEC system.
>appPackageSource	URI	0..1	URI of the application package. Shall be included in the request. The application package shall comply with the definitions in clause 6.2.1.2 of ETSI GS MEC 010-2 [1].

6.3 Subscription data types

In the present document, no subscription data types are defined.

6.4 Notification data types

6.4.1 Introduction

This clause defines data structures for notifications.

6.4.2 Type: AddressChangeNotification

This type represents a notification from the UALCMP regarding a change in address of a user application instance. For the notification procedure reference the method "Receiving notification events" described in clause 5.1.6.

Table 6.4.2-1: Definition of type AddressChangeNotification

Attribute name	Data type	Cardinality	Description
notificationType	String	1	Shall be set to "AddressChangeNotification".
contextId	String	1	Uniquely identifies the application context in the MEC system.
appInstanceId	String	1	Identifier of the user application instance.
referenceURI	URI	1	Address of the user application. Used as the reference URI for the application. Assigned by the MEC system.

6.4.3 Type: ApplicationContextDeleteNotification

This type represents a notification from the UALCMP regarding the deletion of an application context by the MEC system. For the notification procedure reference the method "Receiving notification events" described in clause 5.1.6.

Table 6.4.3-1: Definition of type ApplicationContextDeleteNotification

Attribute name	Data type	Cardinality	Description
notificationType	String	1	Shall be set to "ApplicationContextDeleteNotification".
contextId	String	1	Uniquely identifies the application context that has been deleted from the MEC system.

6.4.4 Type: ApplicationContextUpdateNotification

This type represents a notification from the UALCMP regarding an update to an application change by the MEC system and in particular changes in user application instance location availability. For the notification procedure reference the method "Receiving notification events" described in clause 5.1.6.

This type represents a notification from the UALCMP regarding an update to an application context by the MEC system and in particular changes in user application instance availability. For the notification procedure reference the method "Receiving notification events" described in clause 5.1.6.

Table 6.4.4-1: Definition of type ApplicationContextUpdateNotification

Attribute name	Data type	Cardinality	Description
notificationType	String	1	Shall be set to "ApplicationContextUpdateNotification".
contextId	String	1	Uniquely identifies the application context in the MEC system.
userAppInstanceInfo	array (Structure inlined)	1..N	List of user application instance information.
>appInstanceId	String	1	Identifier of the user application instance.
>referenceURI	URI	1	Address of the user application instance.
>appLocation	LocationConstraints	0..1	Location of the user application instance.

6.4.5 Type: ApplicationLocationAvailabilityNotification

This type represents a notification from the UALCMP regarding the availability of a location that was requested in the Application context create that could not be fulfilled at the time of the request. For the notification procedure reference the method "Receiving notification events" described in clause 5.1.6.

Table 6.4.5-1: Definition of type ApplicationLocationAvailabilityNotification

Attribute name	Data type	Cardinality	Description
notificationType	String	1	Shall be set to "ApplicationLocationAvailabilityNotification".
contextId	String	0..1	Uniquely identifies the application context in the MEC system.
availableLocations	array (Structure (inline))	1..N	Locations available to the MEC application.
>appLocation	LocationConstraints	1	It shall match one of the appLocations in the AppContext sent in the Application context create

6.5 Referenced structured data types

6.5.1 Introduction

This clause defines data structures that may be referenced from data structures defined in clauses 6.2 to 6.4, but may neither be resource representations nor notifications.

6.5.2 Type: LocationConstraints

6.5.2.1 Description

The LocationConstraints data type supports the specification of MEC application requirements related to MEC application deployment location constraints.

The location constraints can be represented as follows:

- as a country code;
- as a civic address combined with a country code;
- as an area, conditionally combined with a country code.

6.5.2.2 Attributes

The attributes of LocationConstraints are shown in table 6.5.2.2-1.

Table 6.5.2.2-1: Attributes of the LocationConstraints information element

Attribute name	Data type	Cardinality	Description
countryCode	String	0..1	The two-letter ISO 3166 [7] country code in capital letters. Shall be present in case the "area" attribute is absent. May be absent if the "area" attribute is present (see note).
civicAddressElement	array (Structure inlined)	0..N	Zero or more elements comprising the civic address. Shall be absent if the "area" attribute is present.
>caType	Integer	1	Describe the content type of caValue. The value of caType shall comply with section 3.4 of IETF RFC 4776 [6].
>caValue	String	1	Content of civic address element corresponding to the caType. The format caValue shall comply with section 3.4 of IETF RFC 4776 [6].
area	Polygon (see [8])	0..1	Geographic area. Shall be absent if the "civicAddressElement" attribute is present. The content of this attribute shall follow the provisions for the "Polygon" geometry object as defined in IETF RFC 7946 [8], for which the "type" member shall be set to the value "Polygon". See note.
NOTE: If both "countryCode" and "area" are present, no conflicts should exist between the values of these two attributes. In case of conflicts, the API producer (e.g. MEO, MEAO) shall disregard parts of the geographic area signalled by "area" that are outside the boundaries of the country signalled by "countryCode". If "countryCode" is absent, it is solely the "area" attribute that defines the location constraint.			

7 API definition

7.1 Introduction

This clause defines the resources and operations of the user application lifecycle management API on the Mx2 reference point.

7.2 Global definitions and resource structure

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

{apiRoot}/{apiName}/{apiVersion}/

The "apiName" shall be set to "dev_app" and the "apiVersion" shall be set to "v1" for the present document. It includes the scheme ("https"), host and optional port, and an optional prefix string. The API shall support HTTP over TLS as defined in clause 6.22 of ETSI GS MEC 009 [9]. All resource URIs in the clauses below are defined relative to the above root URI.

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

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 [9] for more information.

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

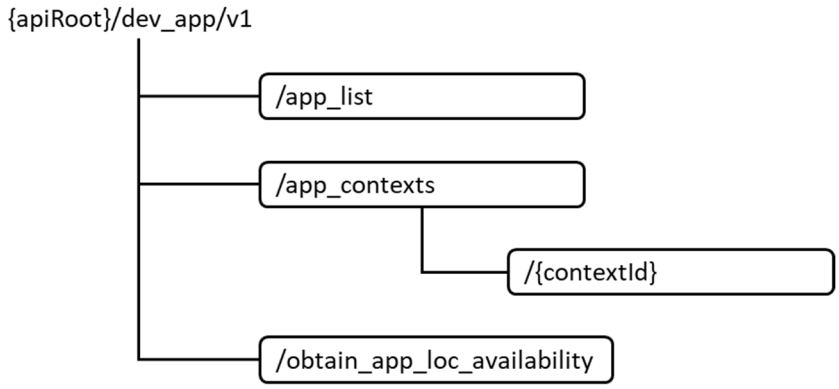

Figure 7.2-1: Resource URI structure of the user application lifecycle management API

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

Table 7.2-1: Resources and methods overview

Resource name	Resource URI	HTTP method	Meaning
meAppList	/app_list	GET	Retrieve available application information.
Parent resource of all devAppContexts	/app_contexts	POST	For requesting the creation of a new application context.
Individual devAppContext	/app_contexts/{contextId}	PUT	For updating the callbackReference and/or appLocation of an existing application context.
		DELETE	For requesting the deletion of an existing application context.
Obtain application location availability task	/obtain_app_loc_availability	POST	Used to obtain the location constraints for a new application context.

7.3 Resource: meAppList

7.3.1 Description

This resource can be queried to retrieve information on available application information.

7.3.2 Resource definition

Resource URI: **{apiRoot}/dev_app/v1/app_list**

Resource URI variables for this resource are defined in table 7.3.2-1.

Table 7.3.2-1: Resource URI Variables for resource "meAppList"

Name	Definition
apiRoot	See clause 8.2

7.3.3 Resource Methods

7.3.3.1 GET

The GET method is used to query information about the available MEC applications.

This method shall comply with 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

Name	Data type	Cardinality	Remarks
appName	String	0..N	Name to identify the MEC application.
appProvider	String	0..N	Provider of the MEC application.
appSoftVersion	String	0..N	Software version of the MEC application.
serviceCont	Enum (inlined)	0..1	Required service continuity mode for this application. Permitted values: 0 = SERVICE_CONTINUITY_NOT_REQUIRED. 1 = SERVICE_CONTINUITY_REQUIRED.
vendorId	String	0..N	Vendor identifier
NOTE: The value of the attribute of the type String shall not exceed the length of 32 characters.			

Table 7.3.3.1-2: 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**
	ApplicationList	1	200 OK	The response body contains an array of the user applications available to the querying device application.
	ProblemDetails	0..1	400 Bad Request	Incorrect parameters were passed in the request. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	401 Unauthorized	An erroneous or missing bearer token. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	The client provided a URI that cannot be mapped to a valid resource URI. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	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.

7.3.3.2 PUT

Not applicable.

7.3.3.3 PATCH

Not applicable.

7.3.3.4 POST

Not applicable.

7.3.3.5 DELETE

Not applicable.

7.4 Resource: all devAppContexts

7.4.1 Description

This resource represents the parent for all individual application contexts.

7.4.2 Resource definition

Resource URI: **{apiRoot}/dev_app/v1/app_contexts**

Resource URI variables for this resource are defined in table 7.4.2-1.

Table 7.4.2-1: Resource URI Variables for resource "all devAppContexts"

Name	Definition
apiRoot	See clause 7.2

7.4.3 Resource Methods

7.4.3.1 GET

Not applicable.

7.4.3.2 PUT

Not applicable.

7.4.3.3 PATCH

Not applicable.

7.4.3.4 POST

The POST method is used to create a new application context. Upon success, the response contains entity body describing the created application context.

This method shall comply with the URI query parameters, the request and response data structures, and response codes, as specified in tables 7.4.3.4-1 and 7.4.3.4-2.

Table 7.4.3.4-1: URI query parameters supported by the POST method on this resource

Name	Data type	Cardinality	Remarks
n/a			

Table 7.4.3.4-2: Data structures supported by the POST request/response on this resource

	Data type	Cardinality	Remarks	
Request body	AppContext	1	Entity body in the request contains the Application Context as requested by the device application.	
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	AppContext	1	201 Created	The response body contains the Application Context as it was created by the MEC system, which includes the reference URI(s) of the associated user application instance(s). The URI of the resource created within the MEC system associated with the request, with its specific application context ID, shall be included in the "Location" HTTP header of the response.
	ProblemDetails	0..1	400 Bad Request	Incorrect parameters were passed in the request. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	401 Unauthorized	An erroneous or missing bearer token. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	The client provided a URI that cannot be mapped to a valid resource URI. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	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.

7.4.3.5 DELETE

Not applicable.

7.5 Resource: individual devAppContext

7.5.1 Description

This resource represents one application context the MEC system has created.

7.5.2 Resource definition

Resource URI: **{apiRoot}/dev_app/v1/app_contexts/{contextId}**

Resource URI variables for this resource are defined in table 7.5.2-1.

Table 7.5.2-1: Resource URI Variables for resource "individual devAppContext"

Name	Definition
apiRoot	See clause 7.2.
contextId	Uniquely identifies the application context in the MEC system. It is assigned by the MEC system.

7.5.3 Resource Methods

7.5.3.1 GET

Not applicable.

7.5.3.2 PUT

The PUT method is used to update the callback reference and/or application location constraints of an existing application context. Upon successful operation, the target resource is updated with the new application context information.

This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in tables 7.5.3.2-1 and 7.5.3.2-2.

Table 7.5.3.2-1: URI query parameters supported by the PUT method on this resource

Name	Data type	Cardinality	Remarks
n/a			

Table 7.5.3.2-2: Data structures supported by the PUT request/response on this resource

	Data type	Cardinality	Response codes	Remarks
Request body	AppContext	1		Only the callbackReference and/or appLocation attribute values are allowed to be updated. Other attributes and their values shall remain untouched.
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	n/a		204 No Content	Upon success, a response 204 No Content without any response body is returned.
	ProblemDetails	0..1	400 Bad Request	Incorrect parameters were passed in the request. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	401 Unauthorized	An erroneous or missing bearer token. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	The client provided a URI that cannot be mapped to a valid resource URI. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	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.

7.5.3.3 PATCH

Not applicable.

7.5.3.4 POST

Not applicable.

7.5.3.5 DELETE

The DELETE method is used to delete the resource that represents the existing application context.

This method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in tables 7.5.3.5-1 and 7.5.3.5-2.

Table 7.5.3.5-1: URI query parameters supported by the DELETE method on this resource

Name	Data type	Cardinality	Remarks
n/a			

Table 7.5.3.5-2: Data structures supported by the 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	Upon success, a response 204 No Content without any response body is returned.
	ProblemDetails	0..1	400 Bad Request	Incorrect parameters were passed in the request. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	401 Unauthorized	An erroneous or missing bearer token. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	The client provided a URI that cannot be mapped to a valid resource URI. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	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.

7.6 Resource: obtain application location availability task

7.6.1 Description

This task resource allows a device application to obtain information on locations available for instantiation of a specific user application in the MEC system.

7.6.2 Resource definition

Resource URI: **{apiRoot}/dev_app/v1/obtain_app_loc_availability**

Resource URI variables for this resource are defined in table 7.6.2-1.

Table 7.6.2-1: Resource URI Variables for resource "obtain application location availability task"

Name	Definition
apiRoot	See clause 7.2

7.6.3 Resource Methods

7.6.3.1 GET

Not applicable.

7.6.3.2 PUT

Not applicable.

7.6.3.3 PATCH

Not applicable.

7.6.3.4 POST

The POST method is used to obtain the locations available for instantiation of a specific user application in the MEC system.

This method shall comply with the URI query parameters, the request and response data structures, and response codes, as specified in tables 7.6.3.4-1 and 7.6.3.4-2.

Table 7.6.3.4-1: URI query parameters supported by the POST method on this resource

Name	Data type	Cardinality	Remarks
n/a			

Table 7.6.3.4-2: Data structures supported by the POST request/response on this resource

	Data type	Cardinality	Remarks	
Request body	ApplicationLocationAvailability	1	Entity body in the request contains the user application information for the MEC system to evaluate the locations available for instantiation of that application.	
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	ApplicationLocationAvailability	1	200 OK	The response body contains the locations available for instantiation of the requested user application in the MEC system.
	ProblemDetails	0..1	400 Bad Request	Incorrect parameters were passed in the request. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	401 Unauthorized	An erroneous or missing bearer token. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	ProblemDetails	0..1	404 Not Found	The client provided a URI that cannot be mapped to a valid resource URI. More information should be provided in the "detail" attribute of the "ProblemDetails" structure.
	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.

7.6.3.5 DELETE

Not applicable.

8 Authentication, authorization and access control

8.1 Introduction

The UALCMP authorizes requests from the device application. The following clause describes this process, whereby OAuth 2.0 is applied on the device application interface.

8.2 Description

An Authentication and Authorization (AA) entity is assumed to be available for both the REST client, i.e. the device application, and the REST server represented by the UALCMP. This AA entity performs the authentication for the credentials of the device application and the UALCMP. TLS is used between the device application and the AA entity to protect subsequent authorization transactions, in support of OAuth 2.0 [4].

The AA entity exposes the token endpoint for the device application as defined by OAuth 2.0. In the present document, the client credentials flow of OAuth 2.0 (see IETF RFC 6749 [4]) shall be supported by the AA entity, and it may be used by the device application to obtain the access token. The AA entity performs the authentication for the credentials of the device application. Other means for the device application to obtain the access token are outside the scope of the present document. The AA entity, and the communication between it and the UALCMP are out of scope of the present document.

It is assumed that the AA entity is configured by a trusted manager entity with the appropriate credentials and access rights information. This configuration information is exchanged between the AA entity and the UALCMP in an appropriate manner to allow the UALCMP to enforce the access rights. The trusted manager entity and the actual way of performing the exchange of this information are out of scope. The device application shall present the access token to the UALCMP with every request in order to assert that it is allowed to access the resource with the particular method it invokes. Examples of access tokens are a certificate-bound access token according to IETF RFC 8705 [10], a bearer token according to IETF RFC 6750 [5]. The OAuth 2.0 flow occurs after a TLS connection is established between the two parties.

NOTE: The process/guidelines for choosing the type of access token is out of scope of the present document.

On the device application interface the access rights of the device application are bound to its access token. Additional policies can also be bound to access token, such as the maximum frequency of API calls. An access token has a lifetime after which it is invalid. An AA entity may revoke an access token before it expires.

The UALCMP forwards to the OSS the requests it receives from the device application. The OSS makes the decision of granting the application context create and delete. The OSS may contact the AA entity to assist in making these decisions; the interactions with the AA entity are out of scope of the present document.

8.3 Provisioning considerations

The basic security configuration for both TLS and OAuth, is done as follows:

The device application needs to be provisioned with certificates. These procedures are based on the use of a Public Key Infrastructure (PKI) and they are out of scope of the present document.

Annex A (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 user application lifecycle management API a supplementary description file compliant to the OpenAPI Specification [i.3].

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

The supplementary description file, relating to the present document, is located at https://forge.etsi.org/rep/mec/gs016-dev-app-api.

History

Document history		
V1.1.1	September 2017	Publication
V2.1.1	April 2019	Publication
V2.2.1	April 2020	Publication
V3.1.1	March 2024	Publication