ETSI GS MEC 033 V3.1.1 (2022-12)

GROUP SPECIFICATION

Multi-access Edge Computing (MEC); IoT API

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

DGS/MEC-0033IoTAPI

Keywords

API, IoT, 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

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

If you find a security vulnerability in the present document, please report it through our Coordinated Vulnerability Disclosure Program:
https://www.etsi.org/standards/coordinated-vulnerability-disclosure

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

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 defines the IoT API to assist the deployment and usage of devices that require additional support in a MEC environment, e.g. due to security constraints, limited power, compute and communication capabilities, such as IoT and MTC devices. The API enables the device provisioning and configuration of the associated components and applications requiring connection to these devices.

The present document describes the information flows and the required information. It also specifies the RESTful binding with the data model.

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] IETF RFC 2818: "HTTP Over TLS".

NOTE: Available at https://www.rfc-editor.org/info/rfc2818.

[3] IETF RFC 5246: "The Transport Layer Security (TLS) Protocol Version 1.2".

NOTE: Available at https://www.rfc-editor.org/info/rfc5246.

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

NOTE: Available at https://www.rfc-editor.org/info/rfc6749.

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

NOTE: Available at https://www.rfc-editor.org/info/rfc6750.

[6] IETF RFC 8446: "The Transport Layer Security (TLS) Protocol Version 1.3".

NOTE: Available at https://www.rfc-editor.org/info/rfc8446.

[7] ETSI TS 133 210: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G; Network Domain Security (NDS); IP network layer security (3GPP TS 33.210)".

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

[i.2] ETSI GS MEC 011: "Multi-access Edge Computing (MEC); Edge Platform Application Enablement".

[i.3] OpenAPI™ Specification.

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

[i.4] ETSI TS 129 561: "5G; 5G System; Interworking between 5G Network and external Data Networks; Stage 3 (3GPP TS 29.561 Release 16)".

[i.5] ETSI TS 129 061: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G; Interworking between the Public Land Mobile Network (PLMN) supporting packet based services and Packet Data Networks (PDN) (3GPP TS 29.061 Release 16)".

[i.6] ETSI GS MEC 010-2: "Multi-access Edge Computing (MEC); MEC Management; Part 2: Application lifecycle, rules and requirements management".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GS MEC 001 [1] apply.

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:

API	Application Programming Interface
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
IoT	Internet of Things
IoTS	IoT Service
MAC	Media Access Control
MQTT	Message Queue Telemetry Transport
NB-IoT	Narrowband IoT
NIDD	Non-IP Data Delivery
PEI	Permanent Equipment Identifier
SUPI	Subscriber Permanent Identifier
USIM	Universal Subscriber Identity Module

4 Overview

The present document specifies the IoT API to support the deployment in a MEC environment of an IoT system based on components that may or may not support other ETSI MEC functions and interfaces.

Clause 5 introduces how the IoT API may be used by an IoT system administrator to execute configuration, provisioning and enablement tasks for the IoT devices and for the other IoT components in order to properly run in a MEC environment.

The information that can be exchanged over the IoT API is described in clause 6 which provides detailed description on all necessary information elements.

Clause 7 defines the actual IoT API providing detailed information how information elements are mapped into a RESTful API design.

5 Description of the service (informative)

5.1 IoT service introduction

The IoT ecosystem comprises a large number of proprietary and standard architectures, aiming at creating an appropriate service layer for the IoT deployment. The service layer comprises a feature-rich set of functions to enable the communication between the IoT devices and the IoT applications, such functions focusing on device management, security aspects, interconnection of applications, etc. For example, proprietary IoT platforms typically offer a message bus (user transport) where IoT devices and IoT applications can publish on/subscribe to topics, a security framework, and additional features like, e.g. data analytics. On top of this, edge computing is deemed a key added value for IoT deployment, but the architecture of an IoT service, despite being arbitrarily complex, may not take into account MEC-related aspects.

In this context, the MEC IoT Service (IoTS) provides means to incorporate heterogeneous IoT platforms in the MEC system, and exposes IoT APIs to enable the configuration of the various components of the overall IoT system. Specifically, the IoT API provides means for IoT system administrators to easily integrate the MEC IoTS and the IoT platform, allowing for, e.g.:

- the discovery of IoT platforms;
- the provisioning of IoT devices into the MEC system;
- the routing of communications between the devices and the requested IoT platform;
- the enablement of discovery and usability of the IoT platform's native APIs.

Depending on the networking/computational capabilities of the IoT device provisioned via the IoT API, the MEC IoTS may be in charge of:

1) either associating the IoT device and the IoT platform's message bus, in case of fully capable IoT devices; or
2) interacting with the message bus client on behalf of the IoT device.

Figure 5.1-1 illustrates a usage example in which an edge facility exists, comprising a MEC host served by a data plane provided by a 3GPP mobile network. First of all, an IoT system administrator needs to configure the following components of the IoT system in an out-of-band fashion (see dashed lines):

1) a (not MEC-compliant) IoT platform, comprising a dedicated message bus (i.e. a user transport not provided by the MEC platform);
2) the IoT devices, e.g. the smart meters of an enterprise. Each IoT device is equipped with a mobile network transceiver and a USIM. The device may also be difficult to reach after the initial deployment;
3) the IoT applications which interact with the IoT platform.

A MEC IoTS exposing IoT API is registered on the MEC platform's service registry, thus discoverable over Mp1 by MEC apps. The IoT API is used by a MEC application featuring a frontend to receive inputs directly from the IoT system administrator, who aims at integrating the IoT platform with the MEC host in order to leverage its mobile-oriented trait to facilitate the usage of 3GPP networks with cellular-agnostic IoT platforms and applications.

In other cases, the IoT API may be used by a MEC application to perform automated operations; such MEC application can be:

i) the IoT application;

ii) the IoT platform; or

iii) a dedicated MEC application, e.g. to complement the functionalities of existing not MEC-compliant components of the IoT platform.

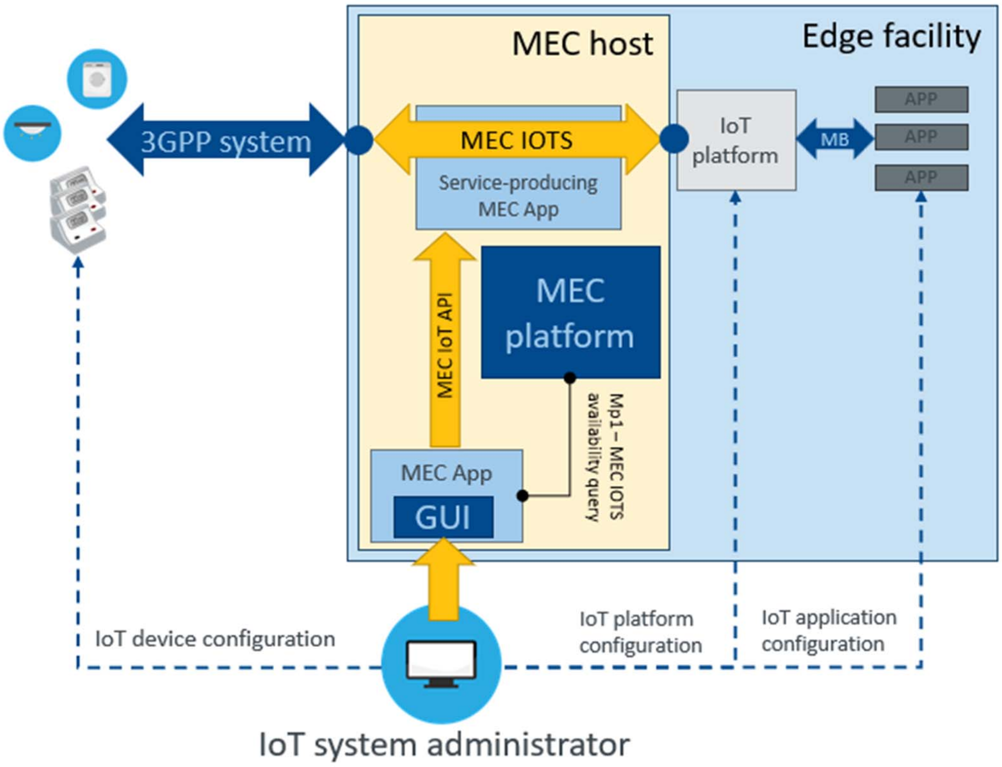

Figure 5.1-1: Usage example of IoT API

The IoT API offers the following services to service consumers:

- IoT platform discovery;
- Device provisioning;
- Transport configuration.

Figure 5.1-2 provides a high-level sequence diagram explaining how the IoT API works. Specifically:

- Step 1 comprises the messages needed to register the MEC IoTS on the MEC platform's service registry leveraging Mp1 [i.2] when the MEC IoTS is provided by a service-producing MEC App.
- Step 2 comprises the messages needed by a service consumer to query the MEC platform's service registry about the availability of the MEC IoTS leveraging Mp1 [i.2].
- Step 3 comprises the messages needed for regular usage of the IoT API by an authorized service consumer, i.e. for IoT platform discovery, device provisioning, and transport configuration. The access rights control is left to the MEC IoTS as per clause 6.16 of ETSI GS MEC 009 [i.1].

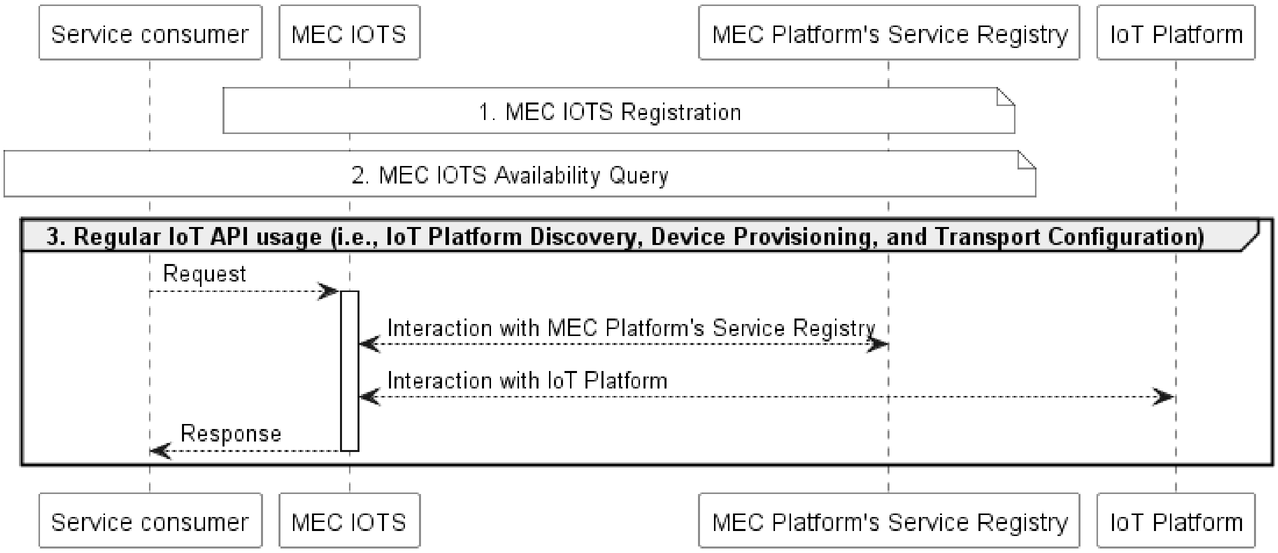
Service consumer
MEC IOTS
MEC Platform's Service Registry
IoT Platform
1. MEC IOTS Registration
2. MEC IOTS Availability Query
3. Regular IoT API usage (i.e., IoT Platform Discovery, Device Provisioning, and Transport Configuration)
Request
Interaction with MEC Platform's Service Registry
Interaction with IoT Platform
Response

Figure 5.1-2: High-level sequence diagram upon IoT API bootstrap and regular operation

5.2 Device provisioning

5.2.1 General

The device provisioning interface enables an IoT system administrator to register, deregister, update information about an IoT device including, for example, managing the association of such a device to a particular traffic rule. The device is referred to by using one or a combination of its identifiers, be they human-defined names or network layer identifiers, e.g. serial number, MAC address, IMEI, etc., whereas the traffic rule can be a MEC traffic rule, defined in ETSI GS MEC 010-2 [i.6], or an alternative traffic rule (i.e. a user transport offered by a IoT platform).

This service enables a constrained device to send (receive) a message to (from) an application even if a protocol translation is necessary (provided that the MEC platform supports a similar traffic rule). As an example, this service permits a NB-IoT device in Non-IP Data Delivery (NIDD) mode to exchange messages with a MEC application that operates as an MQTT client.

Clauses 5.2.2 to 5.2.6 describe the operations available for device provisioning. The related sequence diagrams are presented.

5.2.2 Registered devices query

This operation allows a service consumer to retrieve the information of all the currently registered devices in the MEC platform with a valid traffic rule association. The call flow for this operation is depicted in Figure 5.2.2-1.

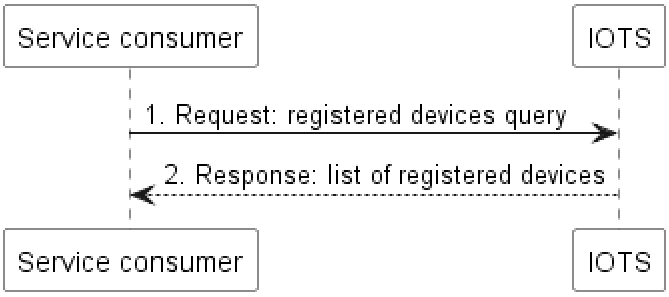
Service consumer
IOTS
1. Request: registered devices query
2. Response: list of registered devices

Figure 5.2.2-1: Flow of Registered devices query

Registered devices query consists of the following steps:

1) The service consumer requests the IoTS to send the list of and the information associated to the registered devices in the MEC platform with a valid traffic rule association.

2) The IoTS replies with the list of registered devices (if any) and their associated information.

5.2.3 Device registration

This operation allows a service consumer to register a new device. The initial registration of the device consists in provisioning a mandatory combination of its identifiers and information for authenticating it to the data network. The registration may also include the association of a traffic rule to the device. The call flow for this operation is depicted in Figure 5.2.3-1.

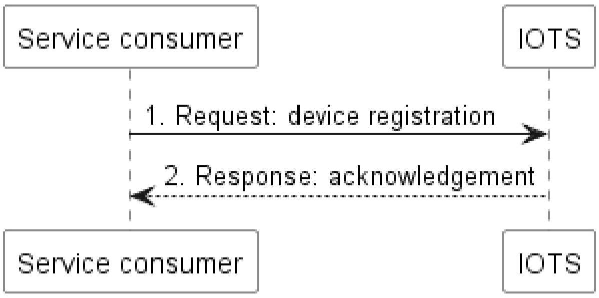

Figure 5.2.3-1: Flow of Device registration

Device registration consists of the following steps:

1) The service consumer requests the IoTS to register the device on the MEC platform.

2) The IoTS acknowledges the operation.

5.2.4 Device registration query

This operation allows a service consumer to obtain information about a registered device. The call flow for this operation is depicted in Figure 5.2.4-1.

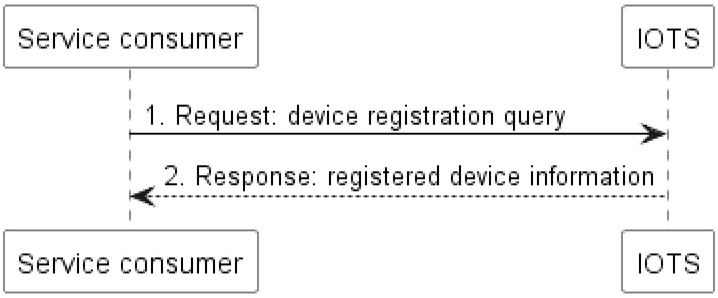
Service consumer
IOTS
1. Request: device registration query
2. Response: registered device information
Service consumer
IOTS

Figure 5.2.4-1: Flow of Device registration query

Device registration query consists of the following steps:

1) The service consumer requests the IoTS to send information about a registered device.

2) The IoTS replies with the information of the registered device (if it exists).

5.2.5 Device registration update

This operation allows a service consumer to update the information about a registered device. The registration update consists in provisioning new types of device identifiers as well as a MEC traffic rule to be associated to the device if these pieces of information were not provided upon initial registration (see clause 5.2.3). The call flow for this operation is depicted in Figure 5.2.5-1.

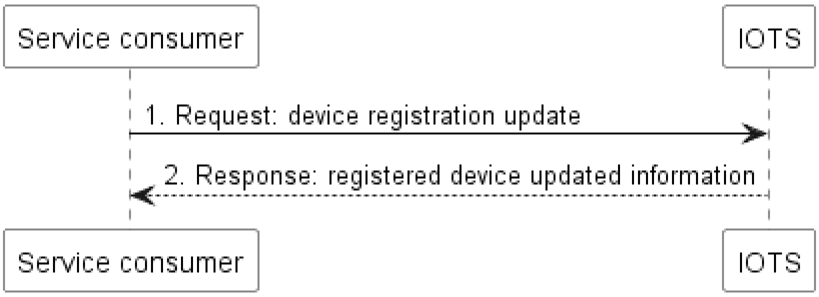
Service consumer
IOTS
1. Request: device registration update
2. Response: registered device updated information
Service consumer
IOTS

Figure 5.2.5-1: Flow of Device registration update

Device registration update consists of the following steps:

1) The service consumer requests the IoTS to update the information about a registered device.

2) The IoTS replies with the updated information of the registered device (if it exists).

5.2.6 Device deregistration

This operation allows a service consumer to deregister a device, i.e. to decommission it from the MEC platform. The call flow for this operation is depicted in Figure 5.2.6-1.

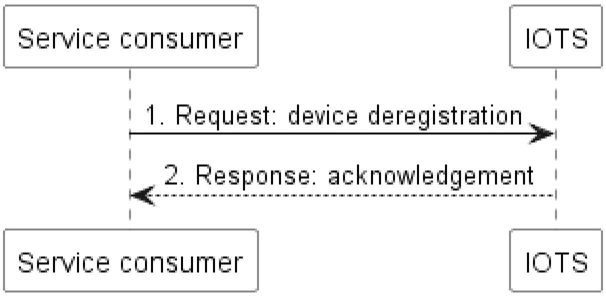
Service consumer
IOTS
1. Request: device deregistration
2. Response: acknowledgement
Service consumer
IOTS

Figure 5.2.6-1: Flow of Device deregistration

Device deregistration consists of the following steps:

1) The service consumer requests the IoTS to deregister the device.

2) The IoTS acknowledges the request.

5.3 IoT platform discovery

5.3.1 General

The IoT platform discovery interface enables an IoT system administrator to configure an IoT platform into the MEC platform and an authorized IoTS consumer to retrieve information about the registered IoT platform(s). This service, as an example, enables an IoT system administrator to register/deregister/update additional IoT gateways and an authorized MEC application to obtain the IoT platform reference so as to use native APIs exposed by the registered IoT platform.

Clauses 5.3.2 to 5.3.6 describe the operations available for IoT platform discovery. The related sequence diagrams are presented.

5.3.2 Registered IoT platforms query

This operation allows a service consumer to retrieve the information of all the IoT platforms currently registered on the MEC platform. The call flow for this operation is depicted in Figure 5.3.2-1.

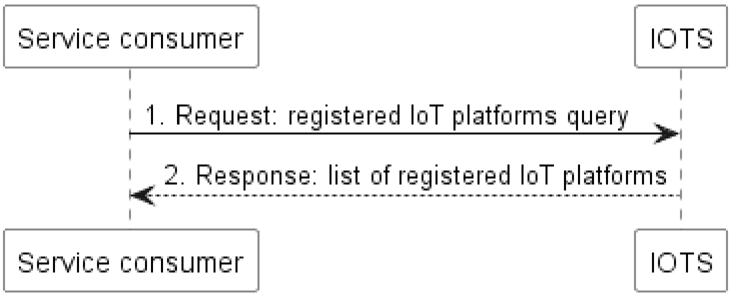
Service consumer
IOTS
1. Request: registered IoT platforms query
2. Response: list of registered IoT platforms
Service consumer
IOTS

Figure 5.3.2-1: Flow of Registered IoT platforms query

Registered IoT platforms query consists of the following steps:

1) The service consumer requests the IoTS to send the list of all registered IoT platforms in the MEC platform.

2) The IoTS replies with the list of registered IoT platforms (if any) and their associated references.

5.3.3 IoT platform information request

This operation allows a service consumer to discover native services running on a registered IoT platform. The call flow for this operation is depicted in Figure 5.3.3-1.

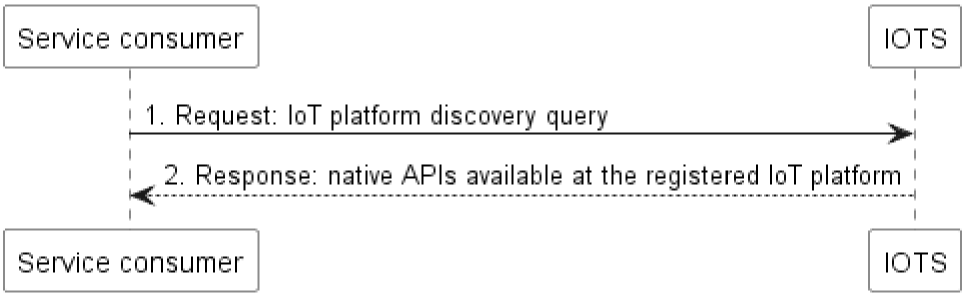
Service consumer
IOTS
1. Request: IoT platform discovery query
2. Response: native APIs available at the registered IoT platform
Service consumer
IOTS

Figure 5.3.3-1: Flow of IoT platform information request

IoT platform information request consists of the following steps:

1) The service consumer requests the IoTS to send information about a registered IoT platform.

2) The IoTS replies with the information of the native services available at the selected IoT platform (if any).

5.3.4 IoT platform registration

This operation allows a service consumer to register an IoT platform on the MEC platform. The call flow for this operation is depicted in Figure 5.3.4-1.

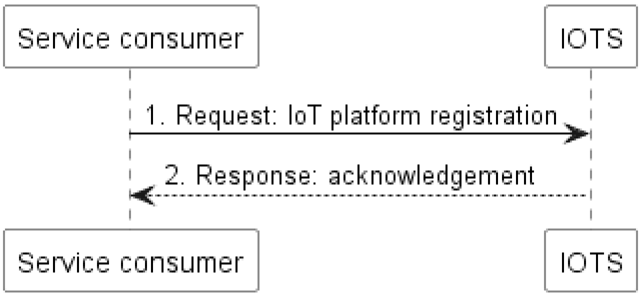
Service consumer
IOTS
1. Request: IoT platform registration
2. Response: acknowledgement
Service consumer
IOTS

Figure 5.3.4-1: Flow of IoT platform registration

IoT platform registration consists of the following steps:

1) The service consumer requests the IoTS to register the IoT platform.

2) The IoTS acknowledges the request.

5.3.5 IoT platform update

This operation allows a service consumer to update the information about a registered IoT platform. The call flow for this operation is depicted in Figure 5.3.5-1.

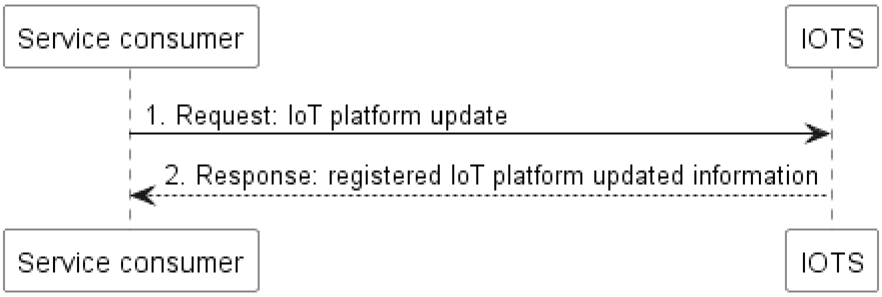
Service consumer
IOTS
1. Request: IoT platform update
2. Response: registered IoT platform updated information
Service consumer
IOTS

Figure 5.3.5-1: Flow of IoT platform update

IoT platform update consists of the following steps:

1) The service consumer requests the IoTS to update the information about a registered IoT platform.

2) The IoTS replies with the updated information of the registered IoT platform (if it exists).

5.3.6 IoT platform deregistration

This operation allows a service consumer to deregister the IoT platform, i.e. to decommission it from the MEC platform. The call flow for this operation is depicted in Figure 5.3.6-1.

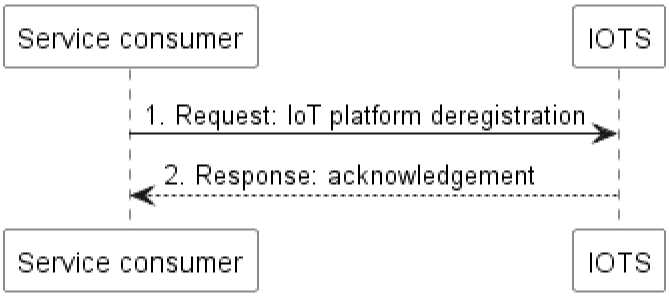
Service consumer
IOTS
1. Request: IoT platform deregistration
2. Response: acknowledgement
Service consumer
IOTS

Figure 5.3.6-1: Flow of IoT platform deregistration

IoT service platform deregistration consists of the following steps:

1) The service consumer requests the IoTS to deregister the IoT platform.

2) The IoTS acknowledges the request.

5.4 Transport configuration

5.4.1 General

The transport configuration interface enables the routing of communications from an IoT device to the intended IoT platform.

If the discovered IoT platform is a service-producing MEC app itself, BYOT is leveraged to register the provided user transport on the MEC platform. Otherwise, the registration of the user transport provided by the discovered IoT platform is performed through the IoT API via the IoT platform registration procedure. In both cases, the registered transport shall be properly labelled in the MEC platform registry, in order to identify it as the user transport provided by the discovered IoT platform instance.

As a result, the IoTS is able to route packets from the provisioned IoT devices to the correct user transport provided by the intended IoT platform.

Clauses 5.4.2 and 5.4.3 describe the operations available for transport configuration. The related sequence diagrams are presented.

5.4.2 User transport query

This operation permits to obtain information about user transports provided by a IoT platform which was discovered in the MEC platform. The call flow for this operation is depicted in Figure 5.4.2-1.

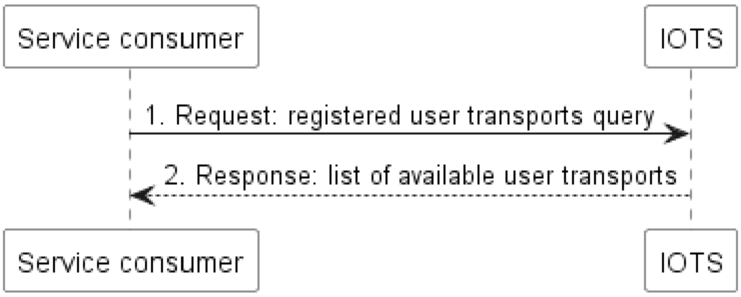
Service consumer
IOTS
1. Request: registered user transports query
2. Response: list of available user transports
Service consumer
IOTS

Figure 5.4.2-1: Flow of User transport query

User transport query consists of the following steps:

1) The service consumer requests the IoTS the list of registered user transports offered by a discovered IoT platform.

2) The IoTS replies with the updated information of the registered user transports (if at least one exists).

5.4.3 User transport assignment

This operation permits to associate a IoT device and an alternative traffic rule, that is, a user transport provided by a discovered IoT platform, if this information was not provided upon initial registration of the device (see clause 5.2.3). The call flow for this operation is depicted in Figure 5.4.3-1.

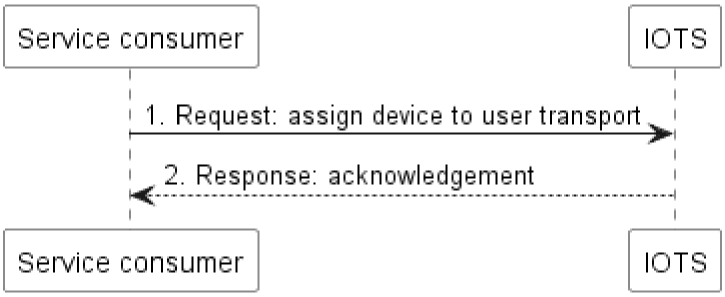
Service consumer
IOTS
1. Request: assign device to user transport
2. Response: acknowledgement
Service consumer
IOTS

Figure 5.4.3-1: Flow of User transport assignment

User transport assignment consists of the following steps:

1) The service consumer requests the IoTS to associate a device and a given user transport offered by a discovered IoT platform.

2) The IoTS acknowledges the request.

6 Data Model

6.1 Introduction

The following clauses specify the data types that are used to implement the IoT API, for which the relevant sequence diagrams are described in clauses 5.2.2 to 5.2.6.

6.2 Resource data types

6.2.1 Introduction

This clause defines data structures that shall be used in resource representations.

6.2.2 Type: DeviceInfo

This type represents the information associated to an IoT device.

Table 6.2.2-1: Definition of type DeviceInfo

Attribute name	Data type	Cardinality	Description
deviceAuthenticationInfo	Not specified	1	Information needed for secondary authentication of the IoT device to the data network - see ETSI TS 129 561 [i.4] and ETSI TS 129 061 [i.5] for 5G and LTE procedures, respectively. This attribute is implementation dependent and should be logically linked to the identifiers of the IoT device listed hereafter.
deviceMetadata	Not specified	0..N	Additional information about the IoT device. This attribute is implementation dependent and may be expressed as an array of key-value pairs.
gpsi	String	0..1	GPSI of the IoT device if 5G-enabled (see note 1).
pei	String	0..1	PEI of the IoT device if 5G-enabled (see note 1).
supi	String	0..1	SUPI of the IoT device if 5G-enabled (see note 1).
msisdn	String	0..1	MSISDN of the IoT device if LTE-enabled (see note 1).
imei	String	0..1	IMEI of the IoT device if LTE-enabled (see note 1).
imsi	String	0..1	IMSI of the IoT device if LTE-enabled (see note 1).
iccid	String	0..1	ICCID of the IoT device (see note 1).
deviceId	String	1	Human-readable identifier of the IoT device.
requestedMecTrafficRule	array(TrafficRuleDescriptor)	0..N	MEC traffic rules the IoT device is requested to be associated to (see note 2). The data type definition is as per ETSI GS MEC 010-2 [i.6].
requestedIotPlatformId	String	0..1	IoT platform to which the IoT device is requested to be associated to (see note 2).
requestedUserTransportId	String	0..1	User transport to which the IoT device is requested to be associated to (see note 2).
deviceSpecificMessageFormats	Structure (inlined)	0..1	Format of the messages to be published by the MEC IoTS on the user transport provided by the associated IoT platform in case the MEC IoTS acts on behalf of the IoT device.
>eventMsgFormat	EventMsg	0..1	Event message format configuration.
>uplinkMsgFormat	UplinkMsg	0..1	Uplink message format configuration.
downlinkInfo	Structure (inlined)	0..1	Downlink communication configuration of the user transport provided by the associated IoT platform in case the MEC IoTS acts on behalf of the IoT device.
>downlinkTopic	String	0..1	Topic associated to the IoT device. This topic should be used by an end IoT application to send downlink data to the IoT device.
>devicePort	Int	0..1	UDP port to be used by the MEC IoTS for the outgoing downlink packets towards the IoT device. In case a default value is used, this attribute is optional.

Attribute name	Data type	Cardinality	Description
clientCertificate	String	0..1	Client-side SSL/TLS certificate to be used by the MEC IoTS to interact with the user transport provided by the associated IoT platform in case the MEC IoTS acts on behalf of the IoT device.
enabled	Boolean	1	Indication whether the IoT device has a valid associated traffic rule (TRUE) or not (FALSE). See note 3.
NOTE 1: At least one attribute among gpsi, pei, supi, msisdn, imei, imsi, and iccid should be provided. Sufficient security measures shall be put in place when any attribute among PEI, SUPI, IMEI, and IMSI is disclosed over the API. NOTE 2: Until a valid traffic rule is not provided, the device will not be able to use the IoTS. A valid traffic rule is provided by one of the following options: • the requestedMecTrafficRule attribute; • the requestedIotPlatformId attribute when the IoT platform offers only one user transport; • the combination of requestedIotPlatformId and requestedUserTransportId. NOTE 3: Enabled is a pseudo-attribute which needs to be maintained by the IoTS based on the presence or not of a valid traffic rule associated to the device. It cannot be set directly by the service consumer.			

6.2.3 Type: IotPlatformInfo

This type represents the information associated to a IoT platform.

Table 6.2.3-1: Definition of type IotPlatformInfo

Attribute name	Data type	Cardinality	Description
iotPlatformId	String	1	Identifier of the IoT platform.
userTransportInfo	array(MBTransportInfo)	1..N	Information about the user transport(s) provided by the IoT platform.
customServicesTransportInfo	array(TransportInfo)	0..N	Transport enabling access to vendor-specific services provided by the IoT platform. The data type definition is as per ETSI GS MEC 011 [i.2].
enabled	Boolean	1	Indication whether the IoT platform is capable of providing user transports and vendor-specific services (TRUE) or not (FALSE).

6.3 Referenced structured data types

6.3.1 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.

6.3.2 Type: MBTransportInfo

This type defines a user transport based on a message bus. It extends the TransportInfo resource data type defined in ETSI GS MEC 011 [i.2], specializing its scope to a transport of MB_TOPIC_BASED type.

Table 6.3.2-1: Definition of type MBTransportInfo

Attribute name	Data type	Cardinality	Description
id	String	1	The identifier of this transport as per ETSI GS MEC 011 [i.2].
name	String	1	The name of this transport as per ETSI GS MEC 011 [i.2].
description	String	0..1	Human-readable description of this transport as per ETSI GS MEC 011 [i.2].
type	TransportType	1	Type of the transport. The attribute shall be set to "MB_TOPIC_BASED."
protocol	String	1	The name of the protocol used. Being the transport of MB_TOPIC_BASED type, this attribute should be typically set to "MQTT" or "AMQP."
version	String	1	The version of the protocol used as per ETSI GS MEC 011 [i.2].
endpoint	EndPointInfo	1	Information about the endpoint to access the transport as per ETSI GS MEC 011 [i.2].
security	SecurityInfo	1	Information about the security used by the transport as per ETSI GS MEC 011 [i.2].
implSpecificInfo	Structure (inlined)	1	Additional implementation specific details of the transport.
>eventTopics	array(String)	0..N	Topics used to publish events related to the established session between the IoT device(s) and the end IoT application(s) on the user transport.
>uplinkTopics	array(String)	0..N	Topics used to publish data generated by the IoT device(s) on the user transport, in order to be consumed by the end IoT application(s).
>downlinkTopics	array(String)	0..N	Topics used to publish data generated by the IoT applications(s) on the user transport, in order to be consumed by the end IoT device(s).

6.3.3 Type: EventMsg

This type defines the format of the messages to be published on the user transport in order to provide application-specific information about events related to the established session between the IoT device and the end IoT application(s).

Table 6.3.3-1: Definition of type EventMsg

Attribute name	Data type	Cardinality	Description
eventTopic	String	1	Topic where the message containing application-specific information should be published.
selectedSerializer	SerializerType	1	Type of serializer to be used for the topic as per ETSI GS MEC 011 [i.2].
includeDeviceAddr	Boolean	0..1	Indication whether to include the IP address of the IoT device (TRUE) or not (FALSE).
includeDeviceMetadata	Boolean	0..1	Indication whether to include the metadata about the IoT device (TRUE) or not (FALSE).
includePei	Boolean	0..1	Indication whether to include the PEI of the IoT device (TRUE) or not (FALSE).
includeSupi	Boolean	0..1	Indication whether to include the SUPI of the IoT device (TRUE) or not (FALSE).
includeImei	Boolean	0..1	Indication whether to include the IMEI of the IoT device (TRUE) or not (FALSE).
includeImsi	Boolean	0..1	Indication whether to include the IMSI of the IoT device (TRUE) or not (FALSE).
includeIccid	Boolean	0..1	Indication whether to include the ICCID of the IoT device (TRUE) or not (FALSE).
includeDeviceId	Boolean	0..1	Indication whether to include the human-readable identified of the IoT device (TRUE) or not (FALSE).

6.3.4 Type: UplinkMsg

This type defines the format of the messages containing uplink data generated by the IoT device to be published on the user transport.

Table 6.3.4-1: Definition of type UplinkMsg

Attribute name	Data type	Cardinality	Description
uplinkTopic	String	1	Topic where the message containing the data generated by the IoT device(s) should be published, in order to be consumed by the end IoT application(s).
selectedSerializer	SerializerType	1	Type of serializer to be used for the topic as per ETSI GS MEC 011 [i.2].
includeDevicePort	Boolean	0..1	Indication whether to include the UDP port of the remote IoT device (TRUE) or not (FALSE).
includeDeviceAddr	Boolean	0..1	Indication whether to include the IP address of the IoT device (TRUE) or not (FALSE).
includeDeviceMetadata	Boolean	0..1	Indication whether to include the metadata about the IoT device (TRUE) or not (FALSE).
includePei	Boolean	0..1	Indication whether to include the PEI of the IoT device (TRUE) or not (FALSE).
includeSupi	Boolean	0..1	Indication whether to include the SUPI of the IoT device (TRUE) or not (FALSE).
includeImei	Boolean	0..1	Indication whether to include the IMEI of the IoT device (TRUE) or not (FALSE).
includeImsi	Boolean	0..1	Indication whether to include the IMSI of the IoT device (TRUE) or not (FALSE).
includeIccid	Boolean	0..1	Indication whether to include the ICCID of the IoT device (TRUE) or not (FALSE).
includeDeviceId	Boolean	0..1	Indication whether to include the human-readable identified of the IoT device (TRUE) or not (FALSE).

7 API definition

7.1 Introduction

This clause defines the resources and operations of the MEC IoT API.

7.2 Global definitions and resource structure

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

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

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

The API shall support HTTP over TLS (also known as HTTPS defined in IETF RFC 2818 [2]). TLS version 1.2 as defined by IETF RFC 5246 [3] shall be supported; TLS 1.3 defined by IETF RFC 8446 [6] should be supported. HTTP without TLS shall neither be supported nor used. TLS implementations should meet or exceed the security algorithm, key length and strength requirements specified in clause 6.2.3 (if TLS version 1.2 is used) or clause 6.2.2 (if TLS version 1.3 is used) of ETSI TS 133 210 [7] (3GPP Release 16 or later).

The content format of JSON shall be supported.

The JSON format is signalled by the content type "application/json".

This API shall require the use of the OAuth 2.0 client credentials grant type according to IETF RFC 6749 [4] with bearer tokens according to IETF RFC 6750 [5]. See ETSI GS MEC 009 [i.1], clause 6.16 for more information. The token endpoint can be discovered as part of the service availability query procedure defined in ETSI GS MEC 011 [i.2]. How the client credentials are provisioned into the MEC application is out of scope of the present document.

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 [i.1] for more information.

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

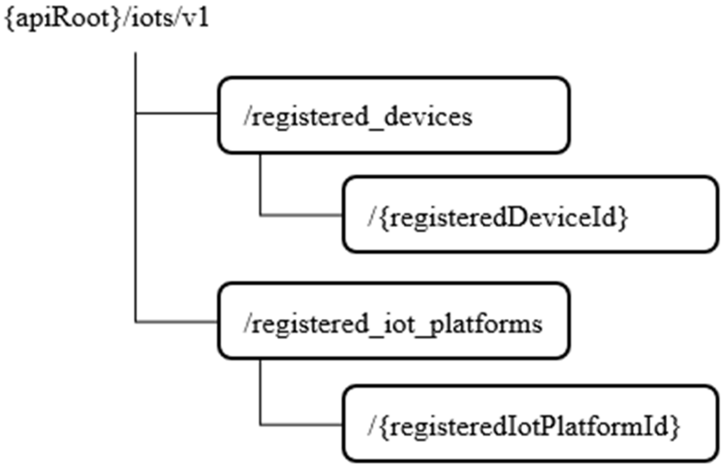

Figure 7.2-1: Resource URL structure of the IoT API

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
All registered devices	/registered_devices	GET	Retrieve information about all devices with a valid traffic rule association - see clause 5.2.2.
		POST	Register a new device - see clause 5.2.3.
A registered device	/registered_devices/{registeredDeviceId}	GET	Retrieve information about a device - see clause 5.2.4.
		PUT	Update information about a device including its association to a valid traffic rule - see clauses 5.2.5 and 5.4.3.
		DELETE	Deregister a device - see clause 5.2.6.
All registered IoT platforms	/registered_iot_platforms	GET	Retrieve information about all IoT platforms - see clause 5.3.2.
		POST	Register a IoT platform - see clause 5.3.4.
A registered IoT platform	/registered_iot_platforms/{registeredIotPlatformId}	GET	Discover native services and obtain information about user transports provided by a IoT platform - see clauses 5.3.3 and 5.4.2.
		PUT	Update information about a IoT platform - see clause 5.3.5.
		DELETE	Deregister a IoT platform - see clause 5.3.6.

7.3 Resource: all registered devices

7.3.1 Description

This resource is used to represent all registered devices.

7.3.2 Resource definition

Resource URI: **{apiRoot}/iots/v1/registered_devices**

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

Table 7.3.2-1: Resource URI variables for resource "all registered devices"

Name	Definition
apiRoot	See clause 7.2

7.3.3 Resource methods

7.3.3.1 GET

The GET method may be used by a service consumer to retrieve information about all devices with a valid traffic rule association. This method is typically used in the "registered devices query" procedure as described in clause 5.2.2.

The GET HTTP 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

Name	Data type	Cardinality	Remarks
filter	String	0..1	Attribute-based filtering parameters according to ETSI GS MEC 009 [i.1]. The API producer shall support receiving the following filtering parameters as part of the URI query string: "(eq,enabled,TRUE)".
fields	array(String)	0..N	The list may contain one or more of the following attributes from the DeviceInfo data type: • deviceMetadata; • gpsi; • msisdn; • deviceId; • requestedMecTrafficRule; • requestedIotPlatformId; • requestedUserTransportId.

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
	array(DeviceInfo)	0..N	200 OK	Upon success, a response body containing the list of registered devices is returned. The response body shall contain a list of resources that match the attribute filter.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed to the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	404 Not Found	It is used when a service consumer 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	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 supported.

7.3.3.3 PATCH

Not supported.

7.3.3.4 POST

The POST method may be used by a service consumer to register a new device. This method is typically used in the "device registration" procedure as described in clause 5.2.3.

The POST HTTP method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in the 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

Name	Data type	Cardinality	Remarks
n/a			

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

	Data type	Cardinality	Response codes	Remarks
Request body	**Data type**	**Cardinality**	**Remarks**	
	DeviceInfo	1	Payload body in the request contains the information associated to the IoT device to be registered.	
Response body	**Data type**	**Cardinality**	**Response codes**	**Remarks**
	DeviceInfo	1	201 Created	Upon success, the HTTP response shall include a "Location" HTTP header that contains the resource URI of the registered device.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed to the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	404 Not Found	It is used when a service consumer 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	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.

7.3.3.5 DELETE

Not supported.

7.4 Resource: a registered device

7.4.1 Description

This resource is used to represent a given registered device.

7.4.2 Resource definition

Resource URI: **{apiRoot}/iots/v1/registered_devices/{registeredDeviceId}**

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

Table 7.4.2-1: Resource URI variables for resource "a registered device"

Name	Definition
apiRoot	See clause 7.2
registeredDeviceId	Representation of a given registered device

7.4.3 Resource methods

7.4.3.1 GET

The GET method may be used by a service consumer to retrieve information about a device. This method is typically used in the "device registration query" procedure as described in clause 5.2.4.

The GET HTTP 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

Name	Data type	Cardinality	Remarks
n/a			

Table 7.4.3.1-2: 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
	DeviceInfo	1	200 OK	Upon success, a response body containing information about the registered device is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed to the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	404 Not Found	It is used when a service consumer 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	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.

7.4.3.2 PUT

The PUT method may be used by a service consumer to update a device registration including its association to a valid traffic rule. This method is typically used in the "device registration update" procedure as described in clause 5.2.5 and in the "user transport assignment" procedure as described in clause 5.4.3.

The PUT HTTP method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in the tables 7.4.3.2-1 and 7.4.3.2-2. Since this method is used for updating a resource, as per clause 6.8.1 of ETSI GS MEC 009 [i.1] the service consumer is required to obtain a representation of the resource by reading it first using the GET method described in clause 7.4.3.1.

Table 7.4.3.2-1: URI query parameters supported by the PUT method on this resource

Name	Data type	Cardinality	Remarks
n/a			

Table 7.4.3.2-2: Data structures supported by the PUT request/response on this resource

Request body	Data type	Cardinality	Remarks	
	DeviceInfo	1	One or more updated attributes that are allowed to be changed are included in the DeviceInfo data structure in the payload body of the request.	
Response body	Data type	Cardinality	Response codes	Remarks
	DeviceInfo	1	200 OK	Upon success, a response body containing data type describing the updated DeviceInfo is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed to the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	404 Not Found	It is used when a service consumer 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	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	412 Precondition Failed	It is used when a condition has failed during conditional requests, e.g. when using Etags to avoid write conflicts. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.

7.4.3.3 PATCH

Not supported.

7.4.3.4 POST

Not supported.

7.4.3.5 DELETE

The DELETE method may be used by a service consumer to deregister a device. This method is typically used in the "device deregistration" procedure as described in clause 5.2.6.

The DELETE HTTP 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

Name	Data type	Cardinality	Remarks
n/a			

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

<table>
<tr><th rowspan="2">Request body</th><th>Data type</th><th>Cardinality</th><th colspan="2">Remarks</th></tr>
<tr><td>n/a</td><td></td><td colspan="2"></td></tr>
<tr><th rowspan="4">Response body</th><th>Data type</th><th>Cardinality</th><th>Response codes</th><th>Remarks</th></tr>
<tr><td>n/a</td><td></td><td>204 No Content</td><td>The operation has been successful.
The response body shall be empty.</td></tr>
<tr><td>ProblemDetails</td><td>0..1</td><td>404 Not Found</td><td>It is used when a service consumer 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.</td></tr>
<tr><td>ProblemDetails</td><td>1</td><td>403 Forbidden</td><td>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.</td></tr>
</table>

7.5 Resource: all registered IoT platforms

7.5.1 Description

This resource is used to represent all registered IoT platforms.

7.5.2 Resource definition

Resource URI: **{apiRoot}/iots/v1/registered_iot_platforms**

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

Table 7.5.2-1: Resource URI variables for resource "all registered IoT platforms"

Name	Definition
apiRoot	See clause 7.2

7.5.3 Resource methods

7.5.3.1 GET

The GET method may be used by an authorized service consumer to retrieve the information of all currently registered IoT platforms. This method is typically used in the "registered IoT platforms query" procedure as described in clause 5.3.2.

The GET HTTP 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

Name	Data type	Cardinality	Remarks
fields	array(String)	0..N	The list shall contain the following attributes from the IotPlatformInfo data type: • iotPlatformId • enabled

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**
	array(IotPlatformInfo)	0..N	200 OK	Upon success, a response body containing the list of registered IoT platforms is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed to the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	404 Not Found	It is used when a service consumer 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	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.

7.5.3.2 PUT

Not supported.

7.5.3.3 PATCH

Not supported.

7.5.3.4 POST

The POST method may be used by a service consumer to register a new IoT platform. This method is typically used in the "IoT platform registration" procedure as described in clause 5.3.4.

The POST HTTP method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in the tables 7.5.3.4-1 and 7.5.3.4-2.

Table 7.5.3.4-1: URI query parameters supported by the POST method on this resource

Name	Data type	Cardinality	Remarks
n/a			

Table 7.5.3.4-2: Data structures supported by the POST request/response on this resource

<table>
<tr><th rowspan="2">Request body</th><th>Data type</th><th>Cardinality</th><th colspan="2">Remarks</th></tr>
<tr><td>IoTPlatformInfo</td><td>1</td><td colspan="2">Payload body in the request contains the information associated to the IoT platform to be registered.</td></tr>
<tr><th rowspan="5">Response body</th><th>Data type</th><th>Cardinality</th><th>Response codes</th><th>Remarks</th></tr>
<tr><td>IoTPlatformInfo</td><td>1</td><td>201 Created</td><td>Upon success, the HTTP response shall include a "Location" HTTP header that contains the resource URI of the registered IoT platform.</td></tr>
<tr><td>ProblemDetails</td><td>0..1</td><td>400 Bad Request</td><td>It is used to indicate that incorrect parameters were passed to the request.
In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.</td></tr>
<tr><td>ProblemDetails</td><td>0..1</td><td>404 Not Found</td><td>It is used when a service consumer 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.</td></tr>
<tr><td>ProblemDetails</td><td>1</td><td>403 Forbidden</td><td>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.</td></tr>
</table>

7.5.3.5 DELETE

Not supported.

7.6 Resource: a registered IoT platform

7.6.1 Description

This resource is used to represent a given registered IoT platform.

7.6.2 Resource definition

Resource URI: **{apiRoot}/iots/v1/registered_iot_platforms/{registeredIotPlatformId}**

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

Table 7.6.2-1: Resource URI variables for resource "a registered IoT platform"

Name	Definition
apiRoot	See clause 7.2
registeredIotPlatformId	Representation of a given registered IoT platform.

7.6.3 Resource methods

7.6.3.1 GET

The GET method may be used by a service consumer to obtain information about a IoT platform. This method is typically used in the "IoT platform information request" procedure as described in clause 5.3.3 and in the "user transport query" procedure as described in clause 5.4.2.

The GET HTTP 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

Name	Data type	Cardinality	Remarks
fields	array(String)	0..N	The list may contain one or more of the following attributes from the IotPlatformInfo data type: • userTransportInfo; • customServiceTransportInfo.

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**
	IotPlatformInfo	1	200 OK	Upon success, a response body containing information about the registered IoT platform is returned.
	ProblemDetails	0..1	400 Bad Request	It is used to indicate that incorrect parameters were passed to the request. In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.
	ProblemDetails	0..1	404 Not Found	It is used when a service consumer 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	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.

7.6.3.2 PUT

The PUT method may be used by a service consumer to update a IoT platform registration. This method is typically used in the "IoT platform update" procedure as described in clause 5.3.5.

The PUT HTTP method shall comply with the URI query parameters, request and response data structures, and response codes, as specified in the tables 7.6.3.2-1 and 7.6.3.2-2. Since this method is used for updating a resource, as per clause 6.8.1 of ETSI GS MEC 009 [i.1] the service consumer is required to obtain a representation of the resource by reading it first using the GET method described in clause 7.6.3.1.

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

Name	Data type	Cardinality	Remarks
n/a			

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

<table>
<tr><th rowspan="2">Request body</th><th>Data type</th><th>Cardinality</th><th colspan="2">Remarks</th></tr>
<tr><td>IotPlatformInfo</td><td>1</td><td colspan="2">One or more updated attributes that are allowed to be changed are included in the IotPlatformInfo data structure in the payload body of the request.</td></tr>
<tr><th rowspan="6">Response body</th><th>Data type</th><th>Cardinality</th><th>Response codes</th><th>Remarks</th></tr>
<tr><td>IotPlatformInfo</td><td>1</td><td>200 OK</td><td>Upon success, a response body containing data type describing the updated IotPlatformInfo is returned.</td></tr>
<tr><td>ProblemDetails</td><td>0..1</td><td>400 Bad Request</td><td>It is used to indicate that incorrect parameters were passed to the request.
In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.</td></tr>
<tr><td>ProblemDetails</td><td>0..1</td><td>404 Not Found</td><td>It is used when a service consumer 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.</td></tr>
<tr><td>ProblemDetails</td><td>1</td><td>403 Forbidden</td><td>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.</td></tr>
<tr><td>ProblemDetails</td><td>0..1</td><td>412 Precondition Failed</td><td>It is used when a condition has failed during conditional requests, e.g. when using Etags to avoid write conflicts.
In the returned ProblemDetails structure, the "detail" attribute should convey more information about the error.</td></tr>
</table>

7.6.3.3 PATCH

Not supported.

7.6.3.4 POST

Not supported.

7.6.3.5 DELETE

The DELETE method may be used by a service consumer to deregister a IoT platform. This method is typically used in the "IoT platform deregistration" procedure as described in clause 5.3.6.

The DELETE HTTP 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

Name	Data type	Cardinality	Remarks
n/a			

Table 7.6.3.5-2: Data structures supported by the DELETE request 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	404 Not Found	It is used when a service consumer 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	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.

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 IoT API a supplementary description file compliant to the OpenAPI Specification [i.3].

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/gs033-iot-api.

History

Document history		
V3.1.1	December 2022	Publication