ISO/IEC 30181

Edition 1.0 2024-11

INTERNATIONAL STANDARD

colour inside

Internet of Things (IoT) – Functional architecture for resource identifier interoperability

ISO/IEC 30181:2024-11(en)

IEC Secretariat
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigendum or an amendment might have been published.

IEC publications search - webstore.iec.ch/advsearchform
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee, …). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and once a month by email.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

IEC Products & Services Portal - products.iec.ch
Discover our powerful search engine and read freely all the publications previews, graphical symbols and the glossary. With a subscription you will always have access to up to date content tailored to your needs.

Electropedia - www.electropedia.org
The world's leading online dictionary on electrotechnology, containing more than 22 500 terminological entries in English and French, with equivalent terms in 25 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

ISO/IEC 30181

Edition 1.0 2024-11

INTERNATIONAL STANDARD

colour inside

Internet of Things (IoT) – Functional architecture for resource identifier interoperability

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 35.020

ISBN 978-2-8327-0065-5

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNET OF THINGS (IoT) – FUNCTIONAL ARCHITECTURE FOR RESOURCE IDENTIFIER INTEROPERABILITY

FOREWORD

1) ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

2) The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC and ISO National bodies.

3) IEC and ISO documents have the form of recommendations for international use and are accepted by IEC and ISO National bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC and ISO documents is accurate, IEC and ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC and ISO National bodies undertake to apply IEC and ISO documents transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC and ISO document and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC and ISO do not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC and ISO marks of conformity. IEC and ISO are not responsible for any services carried out by independent certification bodies.

6) All users should ensure that they have the latest edition of this document.

7) No liability shall attach to IEC and ISO or their directors, employees, servants or agents including individual experts and members of its technical committees and IEC and ISO National bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this ISO/IEC document or any other IEC and ISO documents.

8) Attention is drawn to the Normative references cited in this document. Use of the referenced publications is indispensable for the correct application of this document.

9) IEC and ISO draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). IEC and ISO take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, IEC and ISO had received notice of (a) patent(s), which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at https://patents.iec.ch and www.iso.org/patents. IEC and ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 30181 has been prepared by subcommittee 41: Internet of Things and Digital Twin, of ISO/IEC joint technical committee 1: Information technology. It is an International Standard.

The text of this International Standard is based on the following documents:

Draft	Report on voting
JTC1-SC41/458/FDIS	JTC1-SC41/471/RVD

Full information on the voting for its approval can be found in the report on voting indicated in the above table.

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1, and the ISO/IEC Directives, JTC 1 Supplement available at www.iec.ch/members_experts/refdocs and www.iso.org/directives.

IMPORTANT – The "colour inside" logo on the cover page of this document indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

Internet of Things (IoT) is defined as an infrastructure of interconnected entities, people, systems and information resources together with services which processes and reacts to information from the physical world and virtual world. IoT has attracted significant social attention globally and is expanding in various fields such as smart homes, healthcare, smart cities, logistics, smart cars, etc. In particular, IoT platforms are essential because they connect various devices (e.g. sensors, access points, and data networks) and provide services to the user. Heterogeneous IoT platforms refer to IoT platforms developed based on different standards such as various data models, policies, vendors, interfaces, and specifications. Therefore, interoperability, such as requesting services and sharing resources among heterogeneous IoT platforms, is important, and it is essential for a real IoT system.

IoT platform has many challenges to interoperability, such as support for diverse protocols, discovery service, well-defined semantic management, and processing of data formats in heterogeneous IoT platforms. However, current diverse IoT platforms and related standards make it difficult to achieve interoperability and collaboration between heterogeneous IoT platforms. Especially regarding resource interoperability issues, each IoT platform has been developed using a specific and unique resource identifier, including a different type of resource-request format, so it is difficult to identify resources among heterogeneous IoT platforms. Furthermore, the existing approaches mainly focus on integrating and managing each IoT platform's ontology and a method of duplicating resources for the target IoT platforms. It makes it a burden for the developer to construct specific ontologies for the diverse IoT platforms.

This document provides a functional architecture for resource identifier (ID) interoperability, which converts the format of a resource identifier among heterogeneous IoT platforms. This document concentrates on converting resource paths (e.g. uniform resource identifier (URI)) used in a specific IoT platform to the target IoT platform. In addition, this document provides an IoT resource name system (RNS) architecture based on the comparative analysis of heterogeneous IoT platforms and a smart city scenario, including resource registration, resource deletion, sharing mapping tables, and resource path conversion. To ensure the user can use heterogeneous IoT resources, IoT RNS analyses and converts identifier into desired resource-request formats, including reconfiguring resource requests between heterogeneous IoT platforms as appropriate for the user-requested resources.

This document has the ISO/IEC 30141 [1][1] IoT reference architecture as a reference to consider interoperability among heterogeneous components and systems. In addition, this document has IEC 61406-2 [2] as a reference to specify minimum requirements for a globally unique identification of resources which constitutes a link to its related digital information. Furthermore, the IoT RNS in this document can be modularized in middleware as edge computing in the IoT system. Therefore, this document has as a reference ISO/IEC TR 30164 [3], which describes the general concepts, terms, characteristics, use cases, and techniques (e.g. data management, coordination, processing, network functionality, heterogeneous computing, security, hardware and software optimization) of edge computing for IoT system applications.

[1] Numbers in square brackets refer to the Bibliography.

INTERNET OF THINGS (IoT) – FUNCTIONAL ARCHITECTURE FOR RESOURCE IDENTIFIER INTEROPERABILITY

1 Scope

This document specifies functional requirements and architecture about the following items for resource interoperability among heterogeneous IoT platforms through the conversion of resource identifiers (IDs) and paths (e.g. uniform resource identifier (URI)):

– requirements for interoperability of resource IDs in the heterogeneous IoT platforms;

– functional architecture for converting IDs and paths of resources on heterogeneous IoT platforms; and,

– functional architecture for mapping and managing resource IDs among heterogeneous IoT platforms.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- IEC Electropedia: available at https://www.electropedia.org/
- ISO Online browsing platform: available at https://www.iso.org/obp

3.1
identifier
information that unambiguously distinguishes one entity from other entities in a given identity context

Note 1 to entry: It refers to a name used to identify and distinguish an object.

Note 2 to entry: In the IoT system, it is used to identify resources such as devices and services and related policies can be different for each platform.

[SOURCE: ISO/IEC 20924:2024, 3.1.19 [4], modified – The Notes to entry have been added.]

3.2
identity context
environment where an entity can be sufficiently identified by a certain set of its attributes and values

[SOURCE: ISO/IEC 20924:2024, 3.1.20]

3.3
Internet of Things
IoT
infrastructure of interconnected entities, people, systems and information resources together with services which processes and reacts to information from the physical world and virtual world

Note 1 to entry: In this document, IoT is described as a hyper-connection among smart things, services, and humans that provides useful and seamless services with minimum human involvement.

[SOURCE: ISO/IEC 20924:2024, 3.2.8 modified – Note 1 to entry has been added.]

3.4
IoT platform
software that connects various devices, including sensors, access points, and data networks

Note 1 to entry: An IoT platform is the role of middleware that can connect to networked devices and provide a hosted infrastructure to cost-effectively and securely manage and path data.

Note 2 to entry: It is important that scalable and provide interoperability that handles connectivity to large numbers of devices and easily interacts with them.

Note 3 to entry: It provides user security such as authentication and access control, and a service that collects, visualizes, and analyses data from sensors.

3.5
IoT resource name system
IoT RNS
module that converts resource requests into the respective format of each IoT platform

Note 1 to entry: The IoT RNS can be implemented as local IoT RNS or root IoT RNS.

Note 2 to entry: The local IoT RNS is modularized on each IoT platform to store metadata of registered devices and services. It also converts the resource paths of heterogeneous IoT platforms.

Note 3 to entry: All resource metadata are stored in the root resource table of the root IoT RNS and sent to each local IoT RNS if the root resource table is updated.

3.6
near field communication
NFC
wireless technology that enables communication between devices over a short distance

[SOURCE: ISO 20252:2019, 3.55]

3.7
next generation service interface for linked data
NGSI-LD
information model and API for publishing, querying and subscribing to context-aware information management in IoT systems

3.8
object identifier
OID
ordered list of primary integer values from the root of the international object identifier tree to a node, which unambiguously identifies that node

[SOURCE: ISO/IEC 9834-1:2012, 3.5.11]

3.9
resource
application service used between devices based on various IoT platforms

3.10
radio frequency identification
RFID
wireless use of electromagnetic fields to transfer data, for the purposes of automatically identifying and tracking tags attached to objects

[SOURCE: ISO/IEC 18038:2020, 3.25]

3.11
discovery service
service to find resources, entities, or services based on a specification of the desired target

[SOURCE: ISO/IEC 20924:2024, 3.1.14, modified – Note 1 to entry has been deleted.]

3.12
uniform resource identifier
URI
compact sequence of characters that identifies an abstract or physical resource

Note 1 to entry: It is a conceptual term that refers to a unified identifier for Internet application information resources.

Note 2 to entry: It includes a URN indicating an identifier using the name and a URL indicating a resource path.

[SOURCE: ISO/IEC 12785-1:2009, 3.23, modified – Note 1 to entry has been replaced and Note 2 to entry has been added.]

4 Abbreviated terms

ADN	application dedicated node
ASN	application service node
CSE	common service entity
ID	identifier
IN	infrastructure node
IoT	Internet of Things
MN	middle node
OID	object identifier
RNS	resource name system
URI	uniform resource identifier
URL	uniform resource locator
URN	uniform resource name

5 IoT resource name system

5.1 Requirements for the interoperability of the resource ID in an IoT platform

5.1.1 General

IoT systems consist of devices or smart objects that can interrelate and interconnect among themselves and the environment to provide services to end-users. Figure 1 shows an illustrative example of the IoT metamodel provided in [5]. It shows the relationships of the main elements in the IoT system and is not prescriptive. IoT metamodels are used to understand the complexity of IoT systems and simplify the design, development, and management process. It supports optimizing system architecture and minimizing the problems. In addition, it facilitates communication between various stakeholders in an IoT system. IoT systems distinguish three types of node: physical, intermediate, and application. Every node can uniquely identify each node and component in an IoT system with a URI. A physical node represents the things within an IoT system, such as sensors and actuators. Application service used between devices based on various IoT platforms is referred to as the resource. Application nodes consume resources from physical nodes and provide them to users. Intermediate nodes connect physical and application nodes when they belong to different networks. It can be a bridge or gateway and connect networks using different protocols.

Interoperability in IoT systems is an important requirement so that nodes can exchange resources with each other, regardless of the specific technology or protocol used by service providers and device manufacturers. IoT systems are usually composed of various devices, services, and applications from different vendors or service providers that use different communication technologies and data exchange formats. This document describes the requirements for the interoperability of the resource ID in an IoT system.

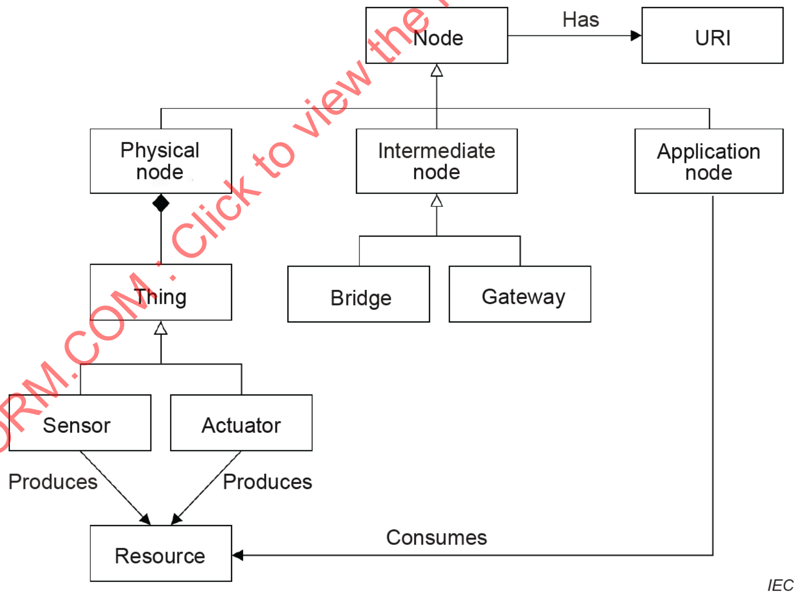


Figure 1 – The IoT metamodel

5.1.2 Uniqueness

Unique ID shall identify entities within the specific application, and IDs used across the IoT platform shall also be unique. Thus, ID allocation shall be organized such that individual IoT platforms can allocate their own set of IDs without conflicting with other IoT platforms.

5.1.3 Equality

Various established and emerging IDs used by the many different IoT platforms shall be considered and supported. Even if IoT platforms use IDs in different formats, equality can be supported by converting the IDs of each platform into a common identifier format. In addition, methods that support ID mapping between different IoT platforms shall be supported to satisfy equality among heterogeneous ID systems.

5.1.4 Persistency

Some users ask that an ID persists throughout the entity's lifetime, whereas others allow IDs to change (e.g. if the entity owner changes), and hence IDs are revocable and replaceable. The persistence of identifiers requires restrictions for some identifier types, such as IP addresses, because identifiers are requested in many requests and responses over the lifetime of an entity. In addition, different persistency policies can be allowed between heterogeneous IoT platforms. Each IoT platform shall explicitly specify its policies and implement strategies to prevent accidental errors that could impact third-party resources (e.g. data corruption due to software malfunctions or service interruptions from hardware failures). Therefore, the IoT platform shall consider the ID treatment method in which the issued ID identifies the correct entity and remains during allocation, transfer, and use.

5.1.5 Scalability

Scalability refers to the ability to be extensible in terms of the number of users and physical nodes without negatively affecting the quality of the services provided by the IoT system. Implementing scalability between nodes in heterogeneous IoT platforms requires an efficient way to manage an IoT system's internal and external nodes. This node management includes node registration and identification and storing and processing the massive volume of data generated by physical nodes. In addition, metadata values such as IDs of nodes used in various IoT platforms require efficient management.

5.1.6 Security

Authenticating the ID shall avoid duplication of ID used for other entities, and verification of ID authenticity shall be possible both online and offline as proof that the ID is assigned to the correct entity. Security requirements (e.g. ID anonymization, using IDs that contain no personal data, disabling ID tracking, and access control to ID information) that the system shall support for ID interoperability between heterogeneous IoT platforms are provided in [6].

5.2 IoT RNS architecture

5.2.1 Assumption

To describe the architecture for the IoT RNS, the following assumptions are made in this document.

– In numerous IoT platforms, various devices are abstracted to different levels. For example, some devices may be core devices performing data collection, calculation, and processing. Some may be sensors that only measure specific data or devices that perform simple services. These low-performance sensors have limitations for requesting and processing resources. Since there are many different types of device in the IoT system, it is difficult to generalize. Therefore, in this document it is assumed that resource requests using the converted resource table are not sent directly to the end device but are sent through the specific platform.

– Currently, some IoT platforms only provide discovery services to users. Discovery service is the service to find available resources based on a specification of the desired target. In the scenario of this document, the root IoT RNS must manage the information of resources registered to each IoT platform. Therefore, this document assumes that each platform can provide a discovery service.

- As mentioned previously, this document defines a resource as the application service used between devices based on various IoT platforms. The interoperability framework concentrates on converting resource paths (e.g. URI used in heterogeneous IoT platforms) in a specific IoT platform to the target IoT platform. The assumed request is limited to one direction, and additional data exchange issues according to the request are not considered.
- The structure, function, and sequence of the figures that describe the framework are illustrative examples and are not prescriptive.

5.2.2 Architecture

Figure 2 represents an illustrative example of system structure and components. It is divided into three factors: server, middleware, and end device.

The server includes the root IoT RNS that manages the registered resources of various IoT platforms. The root IoT RNS has a database that manages the total resource list, mapping table, and resource metadata. Total resource list means the discovered resources (i.e. service name) from IoT platforms. The mapping table includes mapped ID formats among heterogeneous IoT platforms. In addition, resource metadata includes resource name, platform type, device type, device ID, IP address, and resource path.

The middleware includes the IoT platform. The local IoT RNS is modularized in the IoT platform and contains the resource list, resource metadata, local mapping table, local resource table, and the path conversion module.

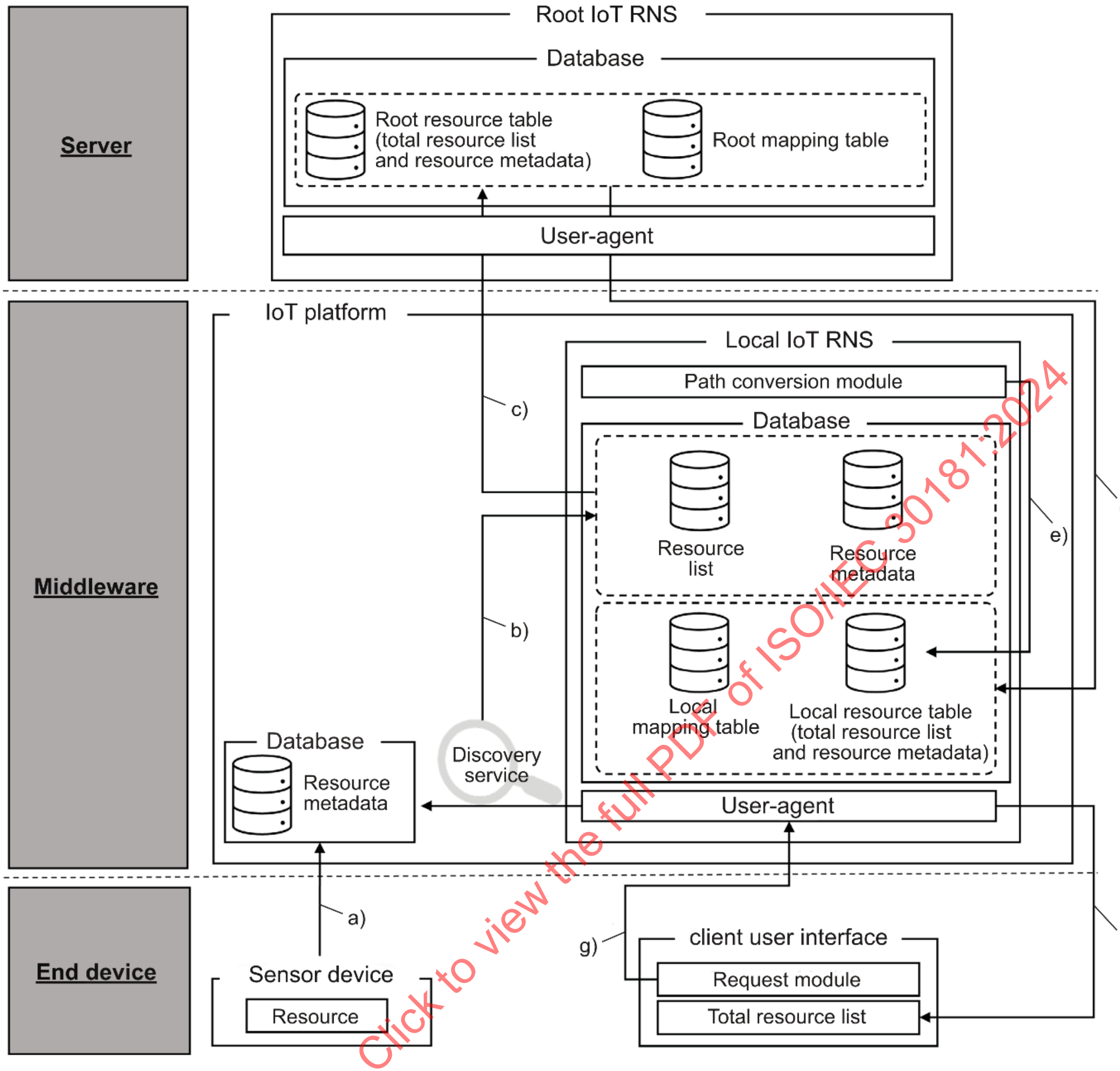


Figure 2 – Overview of system structure and components

The end device includes the sensor devices that perform the service and the client user interface with the total resource list and the request module. The overall flow is as follows.

a) The resource metadata is stored in the IoT platform when a new sensor device is registered.

b) Periodically, metadata through discovery service of the IoT platform are stored in local IoT RNS.

c) In addition, the local IoT RNS periodically shares resource metadata with the root IoT RNS.

d) Resource metadata received from all local IoT RNS are stored in the resource table of the root IoT RNS. This resource table and root mapping table are sent to each local IoT RNS.

e) Each local IoT RNS converts the resource path of total resource list and resource metadata.

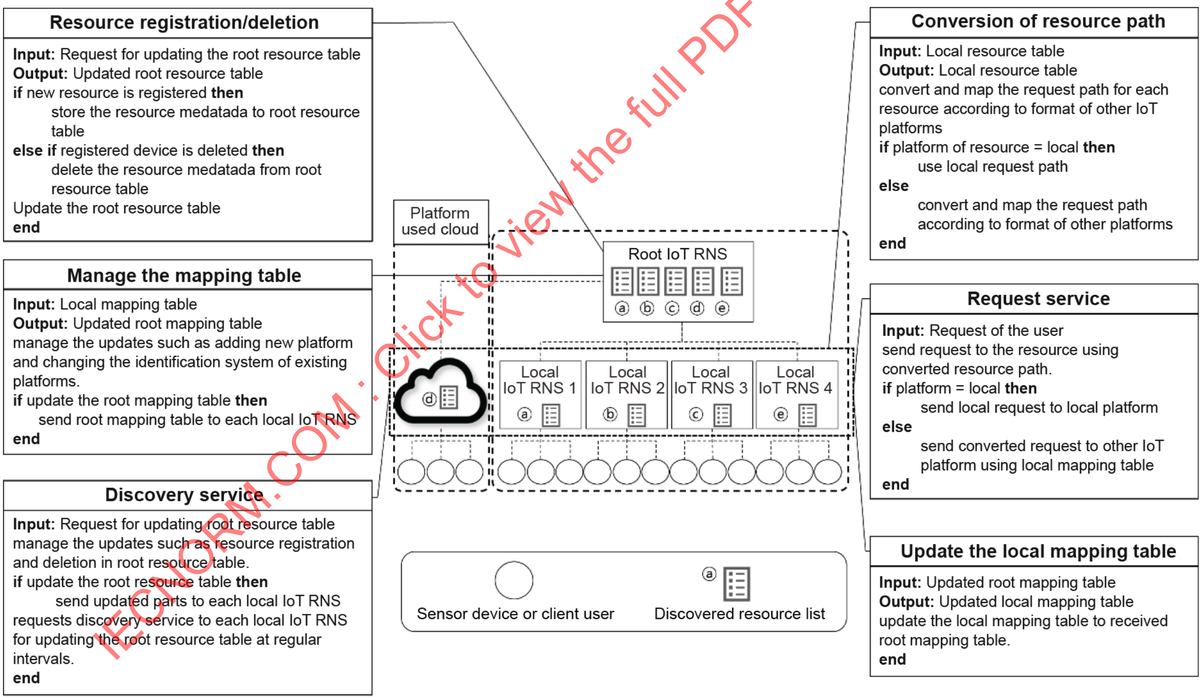
f) The local IoT RNS sends the total resource list to the client user interface.

g) A user requests a resource through the client user interface, and the local IoT RNS uses the converted path to use the resource of another platform.

An IoT RNS is divided into a local IoT RNS that exists as a module on each platform and a root IoT RNS that manages the entire resource table and the mapping table. Figure 3 represents an illustrative example of the IoT RNS architecture. The root IoT RNS manages the metadata for registered and deleted resources. Each local IoT RNS sends related metadata (i.e. device type and device ID) to the root IoT RNS when new devices and services are registered and informs the root IoT RNS when devices and services are deleted.

The root IoT RNS updates the root resource table with resource metadata received from local IoT RNSs and then transmits the root resource table to local IoT RNSs. Furthermore, in addition to when new resource metadata are registered, the root IoT RNS requests discovery service to update the root resource table at regular intervals. Each local IoT RNS converts and stores the resource path in the local resource table received from the root IoT RNS, depending on the format requested by each IoT platform using the local mapping table.

The mapping table includes the root and local mapping table. The root mapping table manages the local mapping tables in various IoT platforms. If a new IoT platform is added or the existing ID format changes, the root mapping table is updated and sent to the local IoT RNS. Then each local IoT RNS updates the local mapping table received from the root mapping table. The mapping table contains resource metadata of diverse IoT platforms, including attributes such as IoT platform type, device type, device ID, IP address, and resource path. The device ID and request format for each IoT platform differ for each IoT platform and are used by the local IoT RNS to convert into the appropriate request format. The examples of resource ID formats for several IoT platforms are detailed in Annex A.



Resource registration/deletion

```
Input: Request for updating the root resource table
Output: Updated root resource table
if new resource is registered then
        store the resource medatada to root resource
        table
else if registered device is deleted then
        delete the resource medatada from root
        resource table
Update the root resource table
end
```

Manage the mapping table

```
Input: Local mapping table
Output: Updated root mapping table
manage the updates such as adding new platform
and changing the identification system of existing
platforms.
if update the root mapping table then
     send root mapping table to each local IoT RNS
end
```

Discovery service

```
Input: Request for updating root resource table
manage the updates such as resource registration
and deletion in root resource table.
if update the root resource table then
        send updated parts to each local IoT RNS
requests discovery service to each local IoT RNS
for updating the root resource table at regular
intervals.
end
```

Conversion of resource path

```
Input: Local resource table
Output: Local resource table
convert and map the request path for each
resource according to format of other IoT
platforms
if platform of resource = local then
        use local request path
else
        convert and map the request path
        according to format of other platforms
end
```

Request service

```
Input: Request of the user
send request to the resource using
converted resource path.
if platform = local then
        send local request to local platform
else
        send converted request to other IoT
        platform using local mapping table
end
```

Update the local mapping table

```
Input: Updated root mapping table
Output: Updated local mapping table
update the local mapping table to received
root mapping table.
end
```

IEC

Figure 3 – The IoT RNS architecture

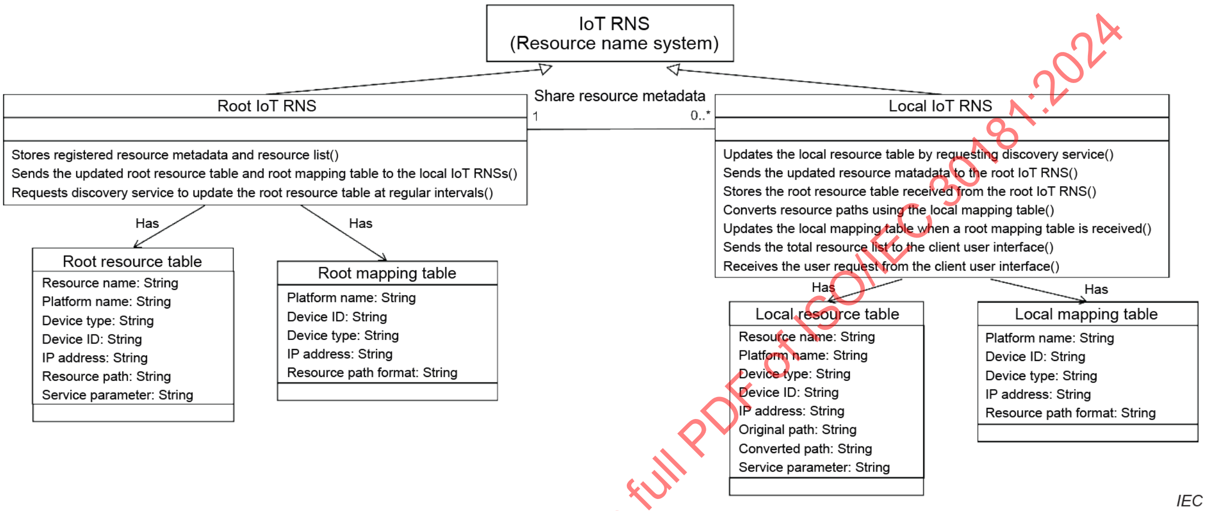
5.2.3 Metamodel

Metamodelling the IoT RNS architecture using the class diagram is shown in Figure 4. As mentioned previously, IoT RNS is divided into root IoT RNS and local IoT RNS. The root IoT RNS performs three functions as follows:

- Stores registered resource metadata and resource list.
- Sends the updated root resource table and root mapping table to the local IoT RNSs.
- Requests discovery service to update the root resource table at regular intervals.

And the local IoT RNS performs seven functions as follows:

- Updates the local resource table by requesting discovery service.
- Sends the updated resource metadata to the root IoT RNS.
- Stores the root resource table received from the root IoT RNS.
- Converts resource paths using the local mapping table.
- Updates the local mapping table when a root mapping table is received.
- Sends the total resource list to the client user interface.
- Receives the user request from the client user interface.


IoT RNS
(Resource name system)
Share resource metadata
1
0..*
Root IoT RNS
Stores registered resource metadata and resource list()
Sends the updated root resource table and root mapping table to the local IoT RNSs()
Requests discovery service to update the root resource table at regular intervals()
Has
Has
Root resource table
Resource name: String
Platform name: String
Device type: String
Device ID: String
IP address: String
Resource path: String
Service parameter: String
Root mapping table
Platform name: String
Device ID: String
Device type: String
IP address: String
Resource path format: String
Local IoT RNS
Updates the local resource table by requesting discovery service()
Sends the updated resource matadata to the root IoT RNS()
Stores the root resource table received from the root IoT RNS()
Converts resource paths using the local mapping table()
Updates the local mapping table when a root mapping table is received()
Sends the total resource list to the client user interface()
Receives the user request from the client user interface()
Has
Has
Local resource table
Resource name: String
Platform name: String
Device type: String
Device ID: String
IP address: String
Original path: String
Converted path: String
Service parameter: String
Local mapping table
Platform name: String
Device ID: String
Device type: String
IP address: String
Resource path format: String

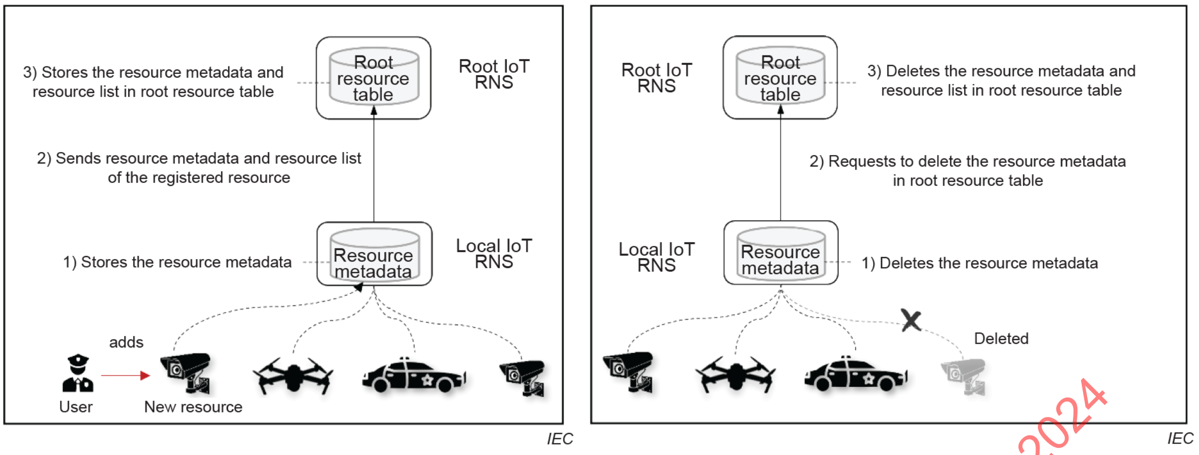
IEC

Figure 4 – The metamodel of IoT RNS

When the resource is updated from the local IoT RNSs, the root IoT RNS sends only the updated resources to the local IoT RNSs. Thus, it can reduce traffic issues between the root and local IoT RNSs. In addition, since the entire resource table is sent at regular intervals, it is possible to check all available resources. The root resource table includes resource name, platform name, device ID, device type, IP address, and resource path. The local resource table includes resource name, platform name, device ID, device type, IP address, original path, and converted path. Root and local mapping tables include platform name, device ID, device type, IP address, and resource path format.

5.2.4 Sequence and algorithms

Each IoT platform contains registered resource metadata. Figure 5 a) represents how the local IoT RNS sends the resource metadata (i.e. platform name, device type, device ID, IP address, and original path) to the root IoT RNS when new resources are registered. The root IoT RNS stores and manages the registered resource metadata of all connected IoT platforms and employs default or arbitrary values for any missing parameter when mapping the metadata by data type.


3) Stores the resource metadata and resource list in root resource table
Root resource table
Root IoT RNS
2) Sends resource metadata and resource list of the registered resource
1) Stores the resource metadata
Resource metadata
Local IoT RNS
adds
User
New resource
IEC
Root IoT RNS
Root resource table
3) Deletes the resource metadata and resource list in root resource table
2) Requests to delete the resource metadata in root resource table
Local IoT RNS
Resource metadata
1) Deletes the resource metadata
Deleted
IEC

a) **Registration of resource**

b) **Deletion of resource**

Figure 5 – Resource registration and deletion of IoT RNS

Figure 5 b) represents how the local IoT RNS manages the deleted resource. IoT RNS checks the availability of the resources regularly by confirming whether the connection is temporarily disconnected or not. The local IoT RNS requests related metadata to be deleted from the root IoT RNS resource table when existing resources disconnect.

Figure 6 represents an illustrative example of converting the resource path in the local IoT RNS. The root IoT RNS manages and updates newly registered and deleted resource metadata. Then, it sends the updated resource table to all connected local IoT RNSs. The local IoT RNS converts and maps the resource path from the root IoT RNS into the resource-request format of its corresponding platform.

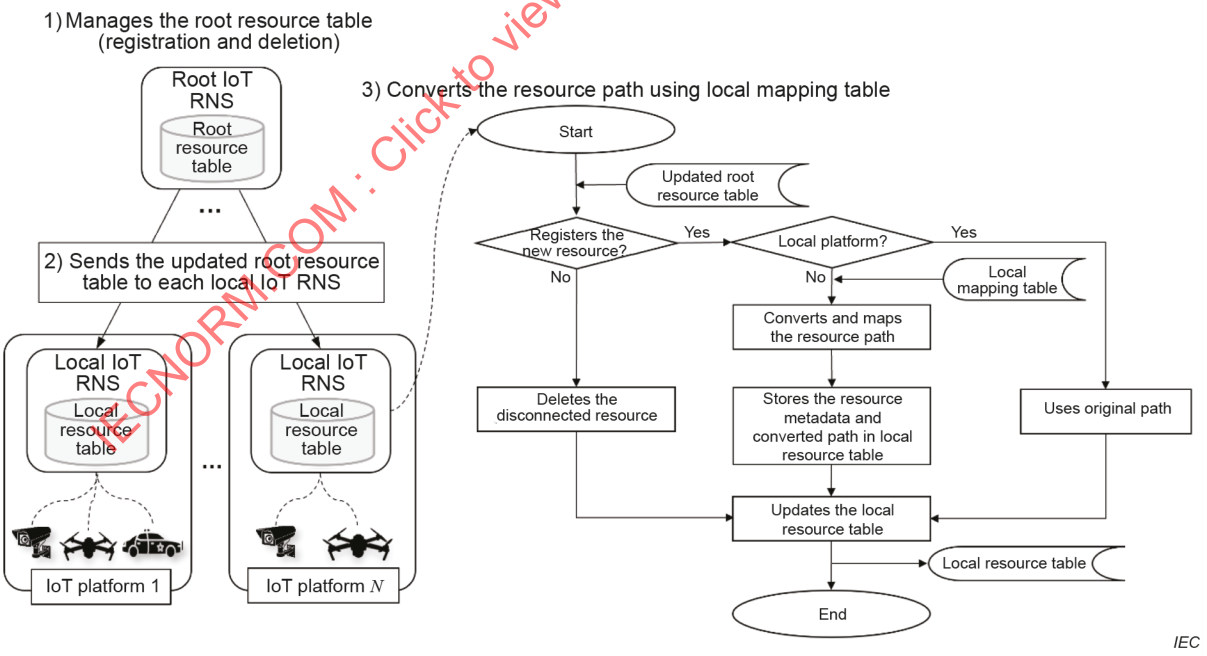

1) Manages the root resource table (registration and deletion)
Root IoT RNS
Root resource table
...
2) Sends the updated root resource table to each local IoT RNS
Local IoT RNS
Local resource table
IoT platform 1
...
Local IoT RNS
Local resource table
IoT platform N
3) Converts the resource path using local mapping table
Start
Updated root resource table
Registers the new resource?
Yes
No
Local platform?
Yes
No
Local mapping table
Converts and maps the resource path
Deletes the disconnected resource
Stores the resource metadata and converted path in local resource table
Uses original path
Updates the local resource table
Local resource table
End
IEC

Figure 6 – Discovery service and path conversion in the local IoT RNS

The conversion algorithm is as follows:

a) Inputs the updated resource table from the root IoT RNS.

b) Checks if the resource is newly updated.

1) If the resource has disconnected: deletes the related metadata.

2) If the resource is newly registered: checks if it is a local platform.

   i) If it is a local platform, uses the original resource path.

   ii) If it is another IoT platform, converts the resource path using the resource-request format in the local mapping table and stores it in the resource table.

c) Updates the resource table.

Annex A
(informative)

Resource identifier format of various IoT platforms

A.1 Overview

This document compares and analyses the resource ID formats of the five IoT platforms in Annex A. Table A.1 represents each IoT platform's device ID and resource-request format. The resource ID format of each platform is analysed below.

Table A.1 – Comparison of five IoT platforms' resource ID formats

Platform[1]	Type	Device ID format
		Resource-Request format
oneM2M	OID based	[OID(Higher Arc)].[ManufacturerID].[DeviceTypeID].[DeviceSerialNo]
		(HTTP) [Server_IP_address]/[CSEBase_name]/[cse_name]{n}/[ae_name]/[cnt_name]
GS1 Oliot	OID based	[GS1 OID({2.51})].[Identification Keys(1)].[ID Key Type], [GS1prefix].[CompanyNo].[ReferenceNo].[Serial/ExtensionNo]
		(HTTP) urn:epc:id:[ID Key Type]:[GS1 ID Key]
IBM Watson IoT	Client ID	d:[orgID]:[deviceType]:[deviceID]
		(HTTP) GET /device/types/[typeId]/devices/[deviceId]/state/[logicalInterfaceId]
OCF IoTivity	Resource Type, Device ID	[di], rt:oic.wk.d, oic.d.[*]
		(coap)://[IP_address]/[URL path]
FIWARE	Entity Type, Entity ID	No specific restriction except some characters (e.g. <, >, etc.)
		(HTTP) [ip address]: [port]/v2/entities/[id] or [type]

[1] Any proprietary technology listed in this table is for the convenience of the user of this document and does not constitute an endorsement by IEC and ISO.

A further resource ID format, which is not linked or limited to a specific IoT platform, is an Identification Link [7].

A.2 oneM2M

The oneM2M standard uses object identifiers (OIDs) to identify devices. The OIDs are object identifiers organized into tree structures specified by the abstract syntax notation (ANS.1) and jointly developed by IEC, ISO and ITU-T, as shown in Figure A.1.

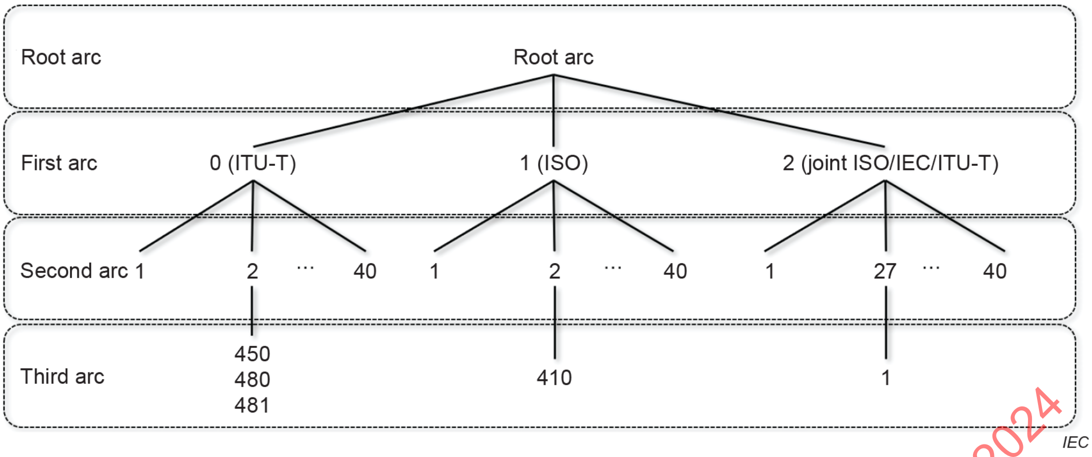


Figure A.1 – International OID tree

The international OID tree is referred to as the higher arc and is used as a prefix for the device ID of oneM2M. As depicted in Figure A.2, the higher arc is followed by "x", "y", "z", where "x" is the device manufacturer, "y" is the device type, and "z" is the device serial number.

The resource structure of oneM2M is provided in [8] and is shown in Figure A.3. This structure consists of an infrastructure node (IN), middle node (MN), application service node (ASN), and an application dedicated node (ADN). Except for the ADN, every other node has a common service entity (CSE). The CSE has a resource structure with a tree format. In the oneM2M infrastructure, each resource has a resource ID and resource name. The latter two are used to request resources.

EXAMPLE In order to request the ASN-AE with a specific resource ID 006 and resource name dev02, the path format is "server01/gateway01/asn01/dev02".

Currently, the oneM2M standard does not use the device ID when it requests resources.

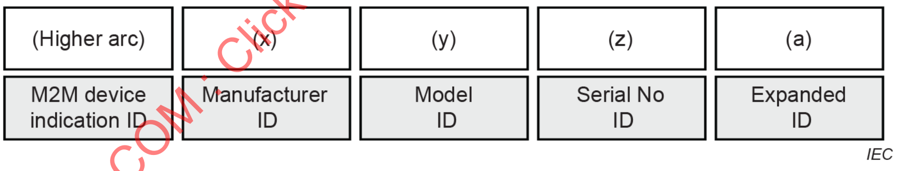


Figure A.2 – oneM2M standard object identifiers

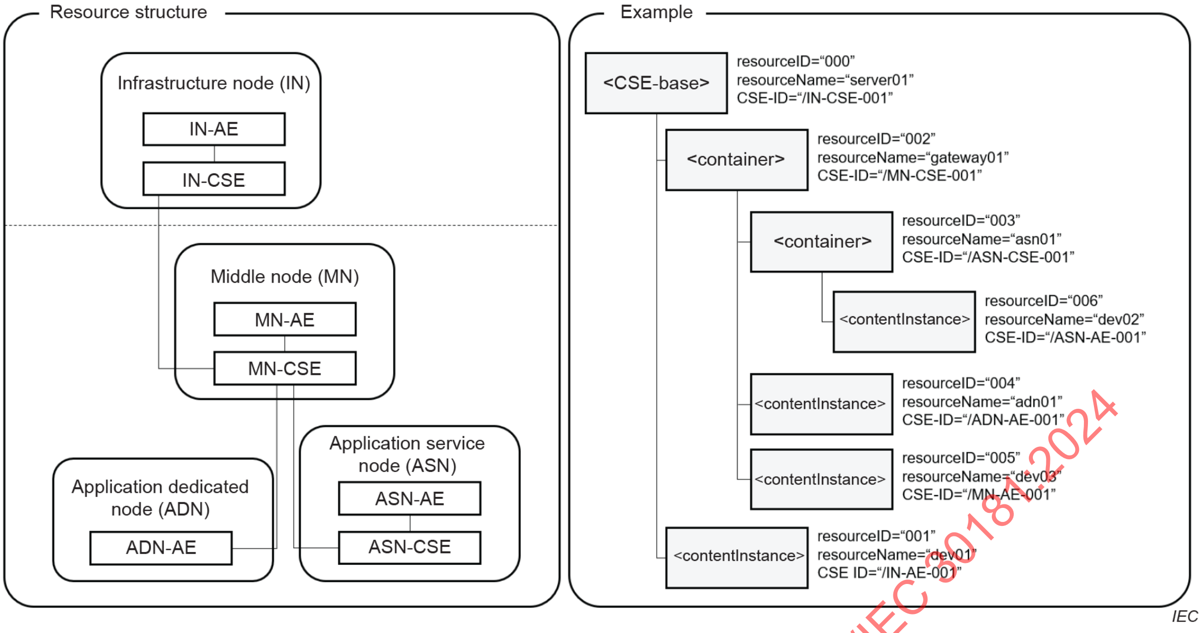


Figure A.3 – oneM2M resource structure

A.3 GS1 Oliot

GS1 uses the resource ID structure provided in [9], and the Oliot project is being developed based on this structure. The overall specifications of the Oliot project are provided in [10]. The GS1 Oliot uses an OID-based ID key (namely the GS1) to identify devices and events. The OID assigned to GS1 is {2.51}. The first arc (2) contains the organization or institution code that represents a joint project between ITU-T and ISO. The second arc (51) represents the GS1. {2.51} is followed by GS1 ID keys (1), GS1 supplementary data (2), GS1 business data (3), and GS1 technical data (4) as child nodes. GS1 ID keys (1) are used as device IDs in the GS1, where the OID is {2.51.1}. The child nodes of {2.51.1} can have 10 key types, which are shown in Table A.2. GS1 ID keys have various types depending on their uses, such as the global trade item number (GTIN) and serial shipping container code (SSCC). The GS1 ID key types can be identified by the field or application of the device. Individual devices can be identified through additional values. These GS1 ID key values include the company prefix, serial number, etc., as shown in Figure A.4.

Table A.2 – GS1 identification key type

OID	ID Key Type	Name
2.51.1.1	GTIN	global trade item number
2.51.1.2	SSCC	serial shipping container code
2.51.1.3	GLN	global location number
2.51.1.4	GRAI	global returnable asset identifier
2.51.1.5	GIAI	global individual asset identifier
2.51.1.6	GDTI	global document type identifier
2.51.1.7	GSRN	global service relation number
2.51.1.8	GSIN	global shipment identification number
2.51.1.9	GINC	global identification number for consignment
2.51.1.10	GCN	global coupon number
2.51.1.11	CPID	component/part identifier
2.51.1.12	GMN	global model number

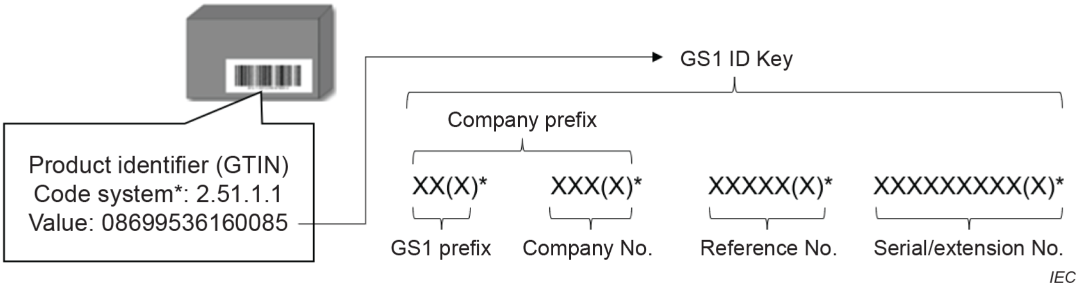


Figure A.4 – GS1 ID key value

GS1 Oliot uses the electronic product code information services (EPCIS) to store and manage devices and resources in an event format. In order to identify and request these devices and resources, the following ID format is used: "urn:epc:id:sgtin:[GS1 ID key]". This format contains the type and value of the GS1 ID key.

A.4 IBM Watson IoT

The ID of the IBM Watson IoT identifies individual devices with unique client IDs. The types and structures of client IDs used by Watson IoT are provided in [11]. The client IDs are formatted by the client type. The client type is divided into application, expandable application, device, and gateway. The format of each client ID is shown in Table A.3. The client ID used to identify each device has the following format: "d:[orgID]:[deviceType]:[deviceID]". In this format, "orgID" is the user's own organization ID. In order to register the device on the IBM Watson IoT platform, the user needs his or her own organization. IBM provides the "orgID" when the user registers his or her own account. Generally, IBM assigns a random six digit-long string to every user. The description of "deviceType" and "deviceID" is shown in Table A.4. The "deviceType" is the type or model of the device. The "deviceID" is the serial number of the device.

Table A.3 – Type of Watson IoT client ID

Client type	ID format
Application	a:[orgID]:[appID]
Expanded application	A:[orgID]:[appID]
Device	d:[orgID]:[deviceType]:[deviceID]
Gateway	g:[orgID]:[typeID]:[deviceID]

Table A.4 – Request identifier parameter

Parameter	Description
typeID	The device type identifier.
deviceID	The device identifier.
logicalInterfacedID or alias	The identifier created for the logical interface or the user-specified alias name.

A.5 OCF IoTivity

The resource structure and format of OCF IoTivity are provided in reference [12]. The OCF IoTivity identifies all resources, including the device, resource, etc., with the value of the resource type (rt). Among them, the device is identified by using the "rt" and an additional value referred to as the device identifier (di). The "rt" value can have the format "oic.wk.d" or "oic.d.[*]". The "oic.d.[*]" represents the specified device attribute. Currently, the "rt" value is assigned the device type of the field for smart home applications and health care applications. The resource request format used by the OCF IoTivity is as follows: "coap://[IP_address]/[URL path]". In this format, the "di" value is used for the URL path. The "di" value uses the universally unique identifier (UUID) and has 36 characters.

A.6 FIWARE

The resource structure and format of FIWARE are provided in [13]. The device ID of FIWARE is based on NGSI standard. FIWARE uses "entity id" and "entity type" to identify a specific entity. These entities also include each device connected to FIWARE, and the device ID and device type that identify each device can be set to "entity id" and "entity type". "Entity id" and "entity type" can be set freely by the user at the time of creation. There is no specific restriction on the format, but it is important that "entity id" is unique within the API and some elements (i.e. <, >, ", ', =,;, (,)) are forbidden. The resource request format of FIWARE is different from each IoT agent. The IoT agent is a component that mediates between a set of devices using their own native protocols and an NGSI-compliant data and service source. In FIWARE, the intelligence data advanced solution (IDAS) generic enabler (GE) offers a wide range of IoT agents making it easier to interface with devices using the most widely used IoT protocols (e.g., lightweight M2M (LWM2M) over constrained application protocol (CoAP), JavaScript object notation (JSON), UltraLight over HTTP/MQTT, or OPC unified architecture (OPC-UA)) as shown in Figure A.5.

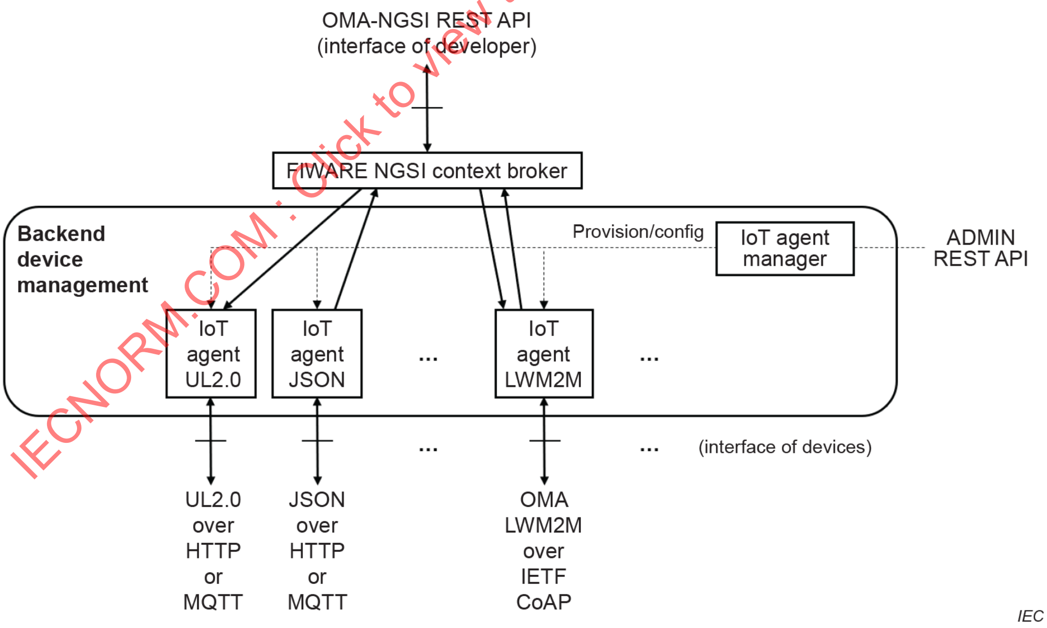


Figure A.5 – FIWARE IoT device management architecture based on IoT agents

This document analyses the NGSI-based HTTP resource request format. The HTTP request format of FIWARE is [ip address]: [port] / v2 / entities / [ID] or [type]. "Ip address" is the IP of the device receiving the request, and FIWARE Orion uses port number 1026. "ID" is the identifier of the entity, and the attribution value is added to the format when users get or update the specific state information of the entity. The format for getting or updating specific status information is [ip address]: [port] / v2 / entities / [ID] or [type] / attrs / [attrsName].

A.7 Identification Link

An Identification Link is a globally unique identification of physical objects which also constitutes a link to its related digital information. The Identification Link is machine-readable and is attached to the physical object in a 2D symbol or NFC tag. The pure data part is the Identification Link string. When an Identification Link string is encoded in a 2D symbol, this symbol is marked with an Identification Link frame. Similarly, when an Identification Link is encoded in an NFC tag, the emblem of this NFC tag is marked with an Identification Link frame. The Identification Link frame is a reserved graphical symbol in IEC 60617 [14]. Figure A.6 shows an example of an Identification Link with a QR-Code in an Identification Link frame, marked with an Identification Link frame when encoded in a data carrier. The Identification Link string has the data format of a link (URL).

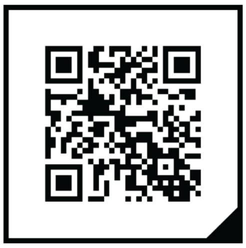


Figure A.6 – Example of Identification Link with QR-Code in Identification Link frame

Figure A.7 shows an RFID emblem with an Identification Link frame. The characters and combinations of allowed characters are restricted to exclude unintended altering of the string during URL processing. Identification Links do not require any specific logic or resolvers. However, they can be used as unique identification and link to related digital data, such as documents or digital twin, without any specific resolvers or logic. Combining these two functionalities in one code and symbol makes it possible to reduce ambiguities and space requirements, especially on type plates and in the various processes requiring identification or linking. Identification Links are a suited ID for assets in IoT systems, e.g. in the context of digital nameplates as a submodel in IEC 61360-4 (Common Data Dictionary and Asset Administration Shells). Adding the prefix "ilstring:" to an Identification Link string makes it possible to generate a related Secure Device Identifier that conforms to IEEE 802.1AR [15].

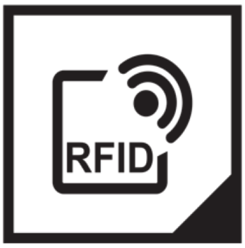


Figure A.7 – Example of RFID emblem with Identification Link frame

Annex B
(informative)

Resource interoperability scenario and implementation examples between heterogeneous IoT platforms in a smart city

B.1 Overview

This document describes an example of adopting the IoT RNS for resource identifier interoperability among heterogeneous IoT platforms in a smart city. Figure B.1 represents the assumed smart city scenario. A smart city is an urban area that provides the information to efficiently manage assets and resources using various electronic data collection sensors.

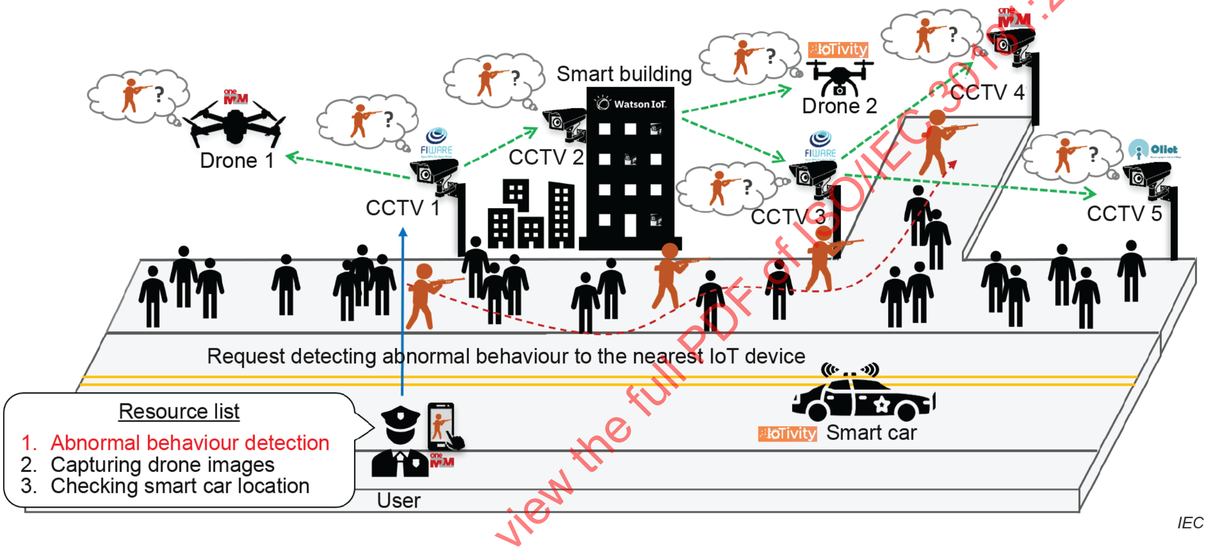
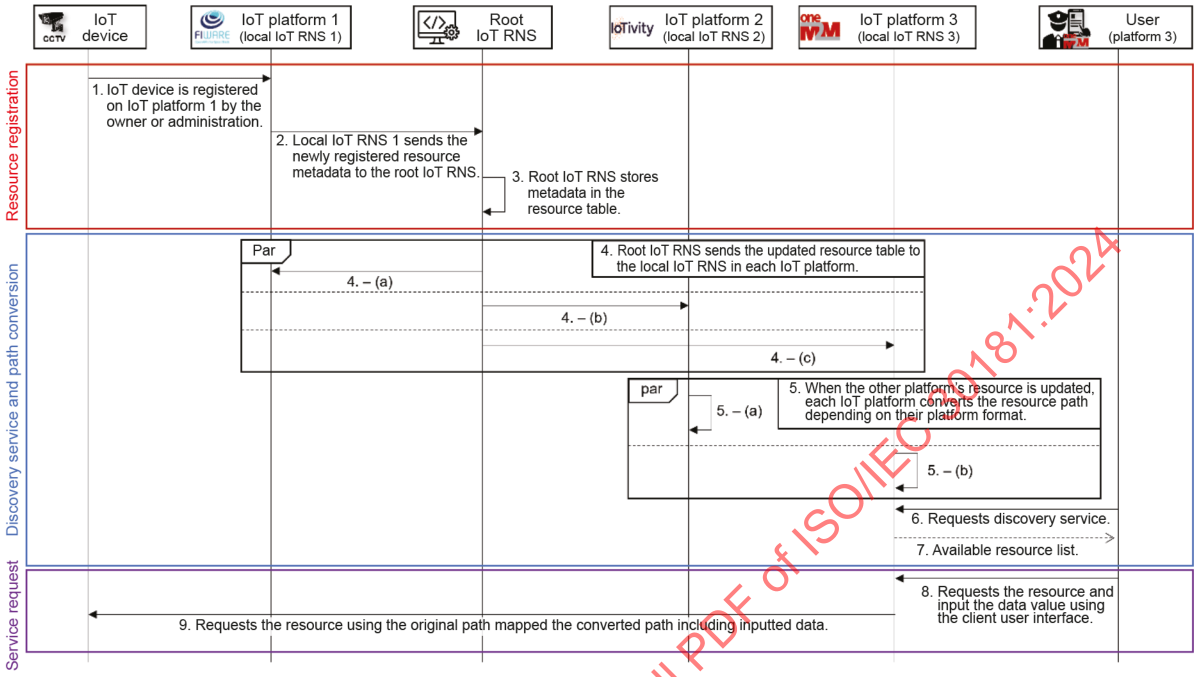


Figure B.1 – IoT RNS interoperability scenario in a smart city

Smart cities connect various resources within the city through a network and optimize city operations by introducing IoT, artificial intelligence, and big data technologies. This scenario is for requesting resources among devices on heterogeneous platforms in a smart city. This scenario is a limited smart city environment, which considers only five IoT platforms (i.e. oneM2M, GS1 Oliot, IBM Watson IoT, OCF IoTivity, and FIWARE) and five hardware device types (i.e. drones, CCTVs, smart buildings, cars, and phones). In this scenario, the root IoT RNS is assumed to be managed by a trusted organization such as a trusted third party (TTP), including the government. The local IoT RNS is modularized in each platform or gateway to communicate with the root IoT RNS. In a smart city, the number of resources in a specific range can be huge due to the nature of resource mobility, and the root IoT RNS can be saturated accordingly. Therefore, this document assumed that the root IoT RNS, which manages all resource lists and metadata, has high computing power. In addition, all IoT platforms are connected to the root IoT RNS and share the list of available resources in advance. A detailed description of the scenario is as follows.

- A user checks the available resources using the oneM2M's smartphone application. At this time, interworking between the user IoT platform (i.e. oneM2M) and the nearest IoT device platform (i.e. FIWARE) is enabled.
- After the user selects the resource (i.e. Abnormal behaviour detection), the resource request is sent to the nearest IoT device (i.e. CCTV 1).
- The resource request is shared with other IoT devices (e.g. CCTVs and drones operated on other IoT platforms) without requiring human intervention.
- Every IoT device that detects abnormal activity sends the results to the user.
- Results can also be sent to other smart objects (e.g. smart cars) if necessary.

This document presents a scenario-based sequence diagram of an example that converts the resource path among heterogeneous IoT platforms, as shown in Figure B.2. It represents the sequence in which a new device is registered in the FIWARE platform, and the resource path for this device is converted and used by other IoT platforms.


IoT device
IoT platform 1 (local IoT RNS 1)
Root IoT RNS
IoT platform 2 (local IoT RNS 2)
IoT platform 3 (local IoT RNS 3)
User (platform 3)
Resource registration
Discovery service and path conversion
Service request
1. IoT device is registered on IoT platform 1 by the owner or administration.
2. Local IoT RNS 1 sends the newly registered resource metadata to the root IoT RNS.
3. Root IoT RNS stores metadata in the resource table.
Par
4. Root IoT RNS sends the updated resource table to the local IoT RNS in each IoT platform.
4. – (a)
4. – (b)
4. – (c)
par
5. When the other platform's resource is updated, each IoT platform converts the resource path depending on their platform format.
5. – (a)
5. – (b)
6. Requests discovery service.
7. Available resource list.
8. Requests the resource and input the data value using the client user interface.
9. Requests the resource using the original path mapped the converted path including inputted data.

IEC

Figure B.2 – Scenario-based sequence diagram that converts the resource path among heterogeneous IoT platforms

Firstly, a new IoT device (i.e. CCTV) is registered on the FIWARE platform by the device owner or administrator. The local IoT RNS modularized in the FIWARE server sends the metadata of newly registered CCTV (e.g. device ID, IP, platform, etc.) to the root IoT RNS. Then, the root IoT RNS stores the metadata of the newly registered device in the resource table and sends the newly added metadata to the local IoT RNS modularized to each IoT platform. Since different platforms' resources have been registered, local IoT RNSs modularized in IoTivity, and oneM2M server converts and stores the resource path according to the ID format used by its own platform. When a user with a oneM2M device checks the available list using discovery service and requests another platform resource, the user's device requests a resource using the converted path. Then, the local IoT RNS modularized in the oneM2M server can request the resource using the original path mapped to the request. This sequence diagram is divided into three stages (i.e. resource registration, discovery service and path conversion, and resource request).

B.2 Resource registration and deletion

The FIWARE server sends the newly registered resource metadata to the root IoT RNS, including resource name, platform type, device type, device ID, IP address, and original resource path, which is stored in the root IoT RNS's resource table as shown in Figure B.3.