ETSI TS 118 130 V4.0.1 (2023-06)

TECHNICAL SPECIFICATION

oneM2M;
Ontology based Interworking
(oneM2M TS-0030 version 4.0.1 Release 4)

Reference
RTS/oneM2M-000030v4

Keywords
M2M, ontology

ETSI

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

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

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

Contents

Intellectual Property Rights

Essential patents

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

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

Trademarks

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

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

Foreword

This Technical Specification (TS) has been produced by ETSI Partnership Project oneM2M (oneM2M).

1 Scope

The present document specifies Generic Interworking of the oneM2M System with external systems (e.g. Area Networks containing non-oneM2M devices) that can be described with ontologies that are compliant with oneM2M's Base Ontology, specified in ETSI TS 118 112 [3].

In oneM2M Release 2 the specification for Ontology based Interworking had been contained in clauses 8 and 9 of ETSI TS 118 112 [3].

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 TS 118 111: "oneM2M; Common Terminology (oneM2M TS-0011)".

[2] ETSI TS 118 101: "oneM2M; Functional Architecture (oneM2M TS-0001)".

[3] ETSI TS 118 112: "oneM2M; Base Ontology (oneM2M TS-0012)".

[4] Void.

[5] ETSI TS 118 123: "oneM2M; Home Appliances Information Model and Mapping (oneM2M TS-0023)".

[6] ETSI TS 118 114: "oneM2M; LWM2M Interworking (oneM2M TS-0014)".

[7] ETSI TS 118 124: "onem2M; OIC Interworking (oneM2M TS-0024)".

[8] ETSI TS 118 104: "oneM2M; Service Layer Core Protocol Specification (oneM2M TS-0004)".

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] oneM2M Drafting Rules.

[i.2] Smart Appliances REFerence (SAREF) ontology.

3 Definitions of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TS 118 111 [1], ETSI TS 118 112 [3] and the following apply:

abstract device: virtual Device (i.e. a set of oneM2M resources together with an IPE) that allows a communicating entity to communicate with an Interworked Device, using an Abstract Information Model, without the need to know the technology specific Device Information Model of that Interworked Device

abstract information model: Information Model of common functionalities abstracted from a set of Device Information Models (see ETSI TS 118 111 [1])

abstraction: process of mapping between a set of Device Information Models and an Abstract Information Model according to a specified set of rules (see ETSI TS 118 111 [1])

abstraction application entity: specialized AE that communicates with an IPE and facilitates Abstraction by providing Services that translate between the Abstract Information Model and the Device Information Model of the IPE

communicating entity: oneM2M entity (usually an AE) that communicates with the IPE for the purpose of sending/receiving data from the Interworked Device

device information model: Information Model of the native protocol for the physical device (see ETSI TS 118 111 [1])

interworked device: Non-oneM2M Device Nodes (NoDN) for which communication with oneM2M entities can be achieved via an Interworking Proxy Application Entity (IPE) (see ETSI TS 118 112 [3])

interworking proxy application entity: specialized AE that facilitates interworking between Non-oneM2M Device Nodes (NoDN) and the oneM2M System

NOTE: An IPE maps data of the NoDN into oneM2M resources (interworked devices). It invokes operations in the NoDN when the related oneM2M resources are modified and modifies oneM2M resources based on the output of NoDN operations (see ETSI TS 118 111 [1])

ontology based interworking: ontology based Interworking allows interworking with many types of non-oneM2M Area Networks and Devices that are described in the form of a oneM2M compliant ontology which is derived from the oneM2M Base Ontology (see ETSI TS 118 112 [3])

NOTE: Ontology based Interworking supports the interworking variant "full mapping of the semantic of the non-oneM2M data model to Mca" as indicated in clause F.2 of ETSI TS 118 101 [2].

proxied device: virtual Device (i.e. a set of oneM2M resources together with an IPE) that represents the Interworked Device in the oneM2M System (see ETSI TS 118 112 [3])

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI TS 118 111 [1], ETSI TS 118 112 [3] and the following apply:

IPE Interworking Proxy Application Entity

NOTE: See ETSI TS 118 111 [1].

4 Conventions

The key words "Shall", "Shall not", "May", "Need not", "Should", "Should not" in the present document are to be interpreted as described in the oneM2M Drafting Rules [i.1].

5 Introduction to Ontology based Interworking (informative)

5.1 Basic concepts of Ontology based Interworking

5.1.1 Ontology based Interworking vs. Specific interworking

oneM2M supports interworking with several specific non-oneM2M solutions. Examples are: LWM2M Interworking (ETSI TS 118 114 [6]) or OIC Interworking (ETSI TS 118 124 [7]). While these examples refer to specific technologies oneM2M also allows to specify only data models - e.g. in ETSI TS 118 123 [5] - which does not assume that a specific technology is used. The data model in ETSI TS 118 123 could e.g. be implemented with 'native' oneM2M entities like ASNs, ADNs and MNs or it could just as well be implemented in a non-oneM2M solution that is interworked with oneM2M via an Interworking Proxy Application Entity (IPE).

Ontology based Interworking is taking an approach similar to ETSI TS 118 123 [5], however in this case the data model is not specified but can flexibly be provided in form of an ontology. That ontology needs to be formally described (e.g. in OWL format).

Ontology based Interworking can be used in cases where oneM2M does not provide a standardized datamodel but still interworking is desired. Such a situation may arise if e.g. a company wants to publish their proprietary datamodel for interworking purposes but does not wish to reveal their proprietary technology (radio technology, communication protocol) for data transmission.

For Ontology based Interworking the ontology that describes the data model of the interworked technology needs to be provided in the oneM2M solution. This ontology enables the IPE to create specific resourcetypes (specializations of <*flexContainer*>), through dynamically created XSDs that are derived from the ontology.
From these resourcetypes oneM2M resources are created by the IPE for communication of oneM2M communicating entities with the IPE.

As with any other form of interworking, the IPE provides the translation of data in these resources into/from the external technology.

5.1.2 Use of ontologies for Ontology based Interworking with Area Networks

Interworking with Area Networks is accomplished in oneM2M through functionality provided by Interworking Proxy Entities (IPEs).

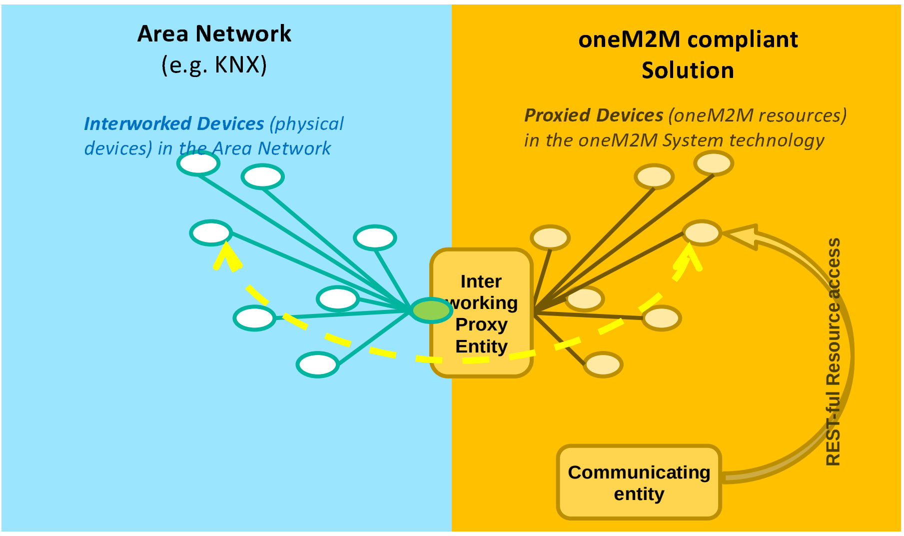

Figure 1: Interworking

The IPE creates "proxied" devices as oneM2M Resources (e.g. AEs) in the oneM2M Solution that can be accessed by communicating entities (e.g. oneM2M Applications) in the usual way.

To accomplish the creation of proxied devices the IPE uses an ontology that describes the Device Information Model of the interworked Area Network and its entities (device types, their operations, etc.).

EXAMPLE 1: In figure 1, an ontology that describes a KNX Area Network and its entities would be needed.

To achieve the flexibility for the IPE to create proxied devices for many different types of Area Networks each ontology that describes a specific Device Information Model needs to be derived from the Base Ontology that is specified in ETSI TS 118 112 [3]. This allows to specify a common scheme of mapping the classes of the ontology that describes the Device Information Model into oneM2M resources (see clause 7).

EXAMPLE 2: The OWL representation of an ontology that describes the Device Information Model of an Area Network of type "KNX" needs to:

a) contain an 'include' statement which includes Base Ontology;

b) the Class of "KNX Nodes" needs to be a subclass of the "Device" Class of oneM2M's Base Ontology;

c) the Class of "KNX Communication Objects" needs to be a subclass of the "Service" Class of the Base Ontology;

d) etc.

NOTE: For the purpose of Ontology based Interworking with Area Networks ***the Base Ontology is only used to describe type information and not for describing instances*** of these types. E.g. the Base Ontology describes the type "Device", but does not contain information about a specific device.The Base Ontology therefore only contains Classes and Properties but not instances.
That principle needs to be followed by the ontology that describes the Device Information Model.

5.2 Using Ontology based Interworking with Device Abstraction

5.2.1 General description

As explained in clause 5.1 it is the task of an IPE to interact via the Area Network with the Interworked Devices and to provide oneM2M resources (Proxied Devices) to the communicating entities for communication with the Interworked Devices. However these Proxied Devices still exhibit the native data model - the Device Information Model of the external technology of the device - and a communicating entity needs to know that native Device Information Model (e.g. ZigBee, KNX, information model).

Device abstraction relieves a communicating entity that wants to communicate with an Interworked Device (e.g. a ZigBee device) from the need to know the native Device Information Model of that Interworked Device.

Additionally to providing interworking, the IPE may translate between the - technology specific - native Device Information Model and an Abstract Information Model, that is based on of common functionalities abstracted from a set of Device Information Models. Such Abstract Information Models can be provided by industry associations of a specific industry sector. An example of an Abstract Information Model, which is specified in oneM2M is the Home Appliance Information Model (HAIM), specified in ETSI TS 118 123 [5].

As in the case of a native Device Information Model that is used by the IPE also an Abstract Information Model can be described by an ontology and that ontology needs to be derived from the Base Ontology.

5.2.2 An example, involving ZigBee, HAIM and SAREF

Figure 2 illustrates this situation for a light switch. In the example the physical implementation is a ZigBee device implementing a ZigBee Service "On/Off Cluster". An IPE for ZigBee creates the interworking towards the ZigBee network.

To enable abstraction, this device is abstracted as oneM2M device according to the Home Appliance Information Model (HAIM). In HAIM the corresponding Service is a "binary Switch".

Both types of Services expose a Function "On Off Function" which is e.g. described in the SAREF ontology.

To turn the switch on SAREF defines an "On Command".

The corresponding Service in HAIM is executed by setting an Input Datapoint called "powerState" to the binary value "TRUE".

In Zigbee an operation (ZigBee command) needs to be invoked in the On/Off Cluster with an input parameter (ZigBee Command ID) equal to 0.

A VariableConversion can been specified in the ontology of the ZigBee Device Information Model that contains the rules how to convert a value of InputDataPoint "powerState" into a value of OperationInput "ZigBee Command ID".

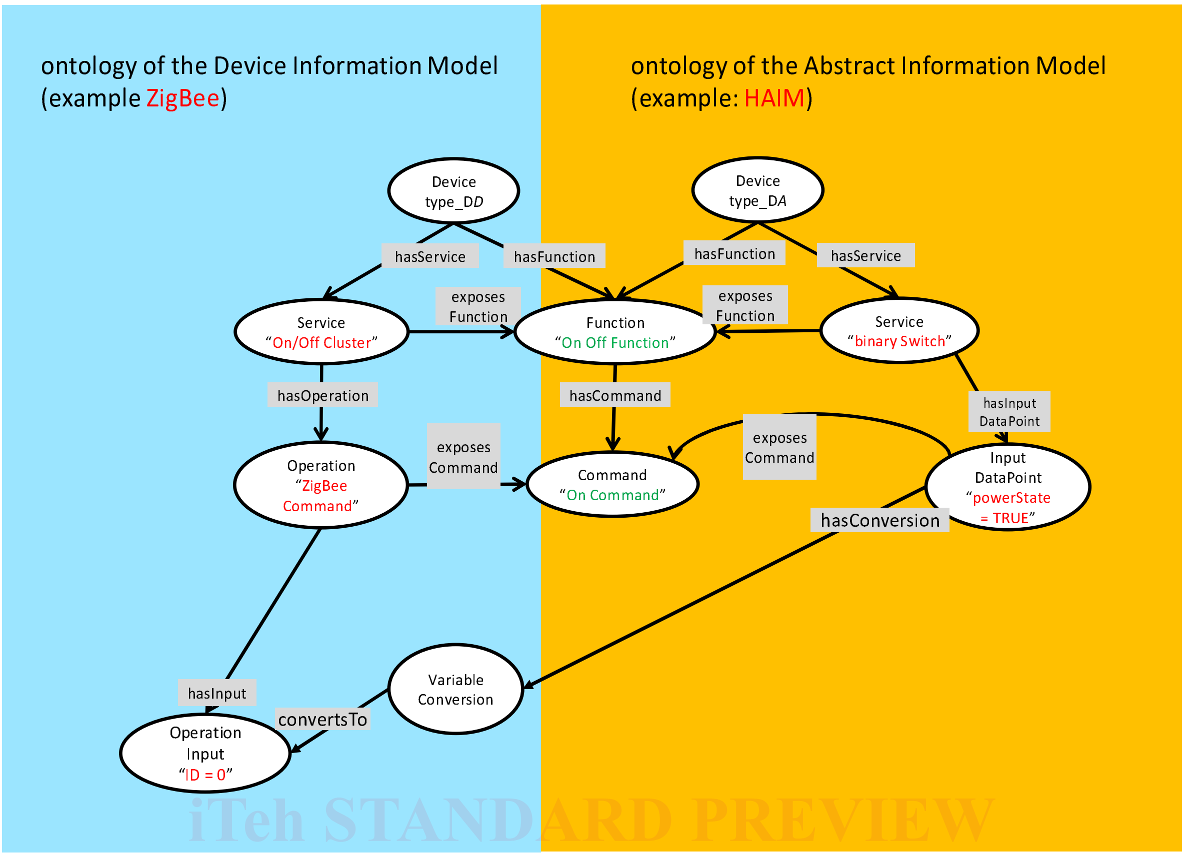

Figure 2: Ontologies relations

5.3 Priciples of data flows

5.3.1 Preconditions on the communicating entity

Data flows between a communicating entity and the IPE involve oneM2M resources - specialized <*flexContainer*>s and possibly <*AE*>s - that were created by the IPE for the purpose of that communication.

NOTE 1: In this clause and in the subsequent clauses it is assumed that the communicating entity and the IPE uses the oneM2M subscribe/notify mechanism to become informed about UPDATEd resources.
It remains, however, an implementation option if subscribe/notify is used or other mechanisms (polling or other mechanisms, e.g. in case of IPE collocated/integrated in its hosting CSE) are used.

1) Any communicating entity, that:

   a) wants to discover interworked non-oneM2M devices of the non-oneM2M system via the IPE needs to be subscribed to the <*AE*> resource of the IPE to get notified about newly created resources for Proxied Devices.
   These resources are created by the IPE to represent interworked non-oneM2M devices that were discovered by the IPE:

      ▪ A communicating entity also needs to be subscribed to the <*AE*> resource of the IPE if it wants to use network services (e.g. broadcast services, registration services, etc.) that are offered by the IPE.

NOTE 2: Since the IPE, in addition to being a oneM2M *AE*, is part of the non-oneM2M system it contains a non-oneM2M device that can offer network services.

   b) wants to communicate with a specific interworked non-oneM2M device via the IPE needs to be subscribed to the <*flexContainer*> (or <*AE*>) resource that had been created by the IPE to represent that interworked non-oneM2M device as Proxied Device.

c) wants to communicate with a sub-device of a Proxied Device, represented by a <*flexContainer*> that is a child-resource of the Device's <*flexContainer*> (or <*AE*>) resource the communicating entity needs also to be subscribed to that <*flexContainer*> child-resource.

2) The communicating entity needs also be subscribed to:

a) The <*flexContainer*> resources, representing Services, that have been created by the IPE as child resourses of the resource of the Proxied Device. The attribute *notificationContentType* of the <*subscription*> needs to be set to "modified-attributes".

b) The <*flexContainer*> resources for the Operation invocation. These resources are child-resources of their respective Service <*flexContainer*> resources.
The *eventNotificationCriteria* conditions the *notificationEventType* needs to contain:

- (B) Deletion of the subscribed-to resource
… to get notified when theoperation becomes unavailable

and, if the operation involves an answer (operation results). also

- (C) Creation of a direct child of the subscribed-to resource
… to get notified about the creation of a <*flexContainer*> for the Operation result.

c) The <*flexContainer*> for the Operation result. The attribute *notificationContentType* of the <*subscription*> needs to be set to "all-attributes".

5.3.2 Data flows for communicating with the IPE using DataPoints of a Service

Figure 3 show the data flows for communicating with the IPE using DataPoints of a Service.

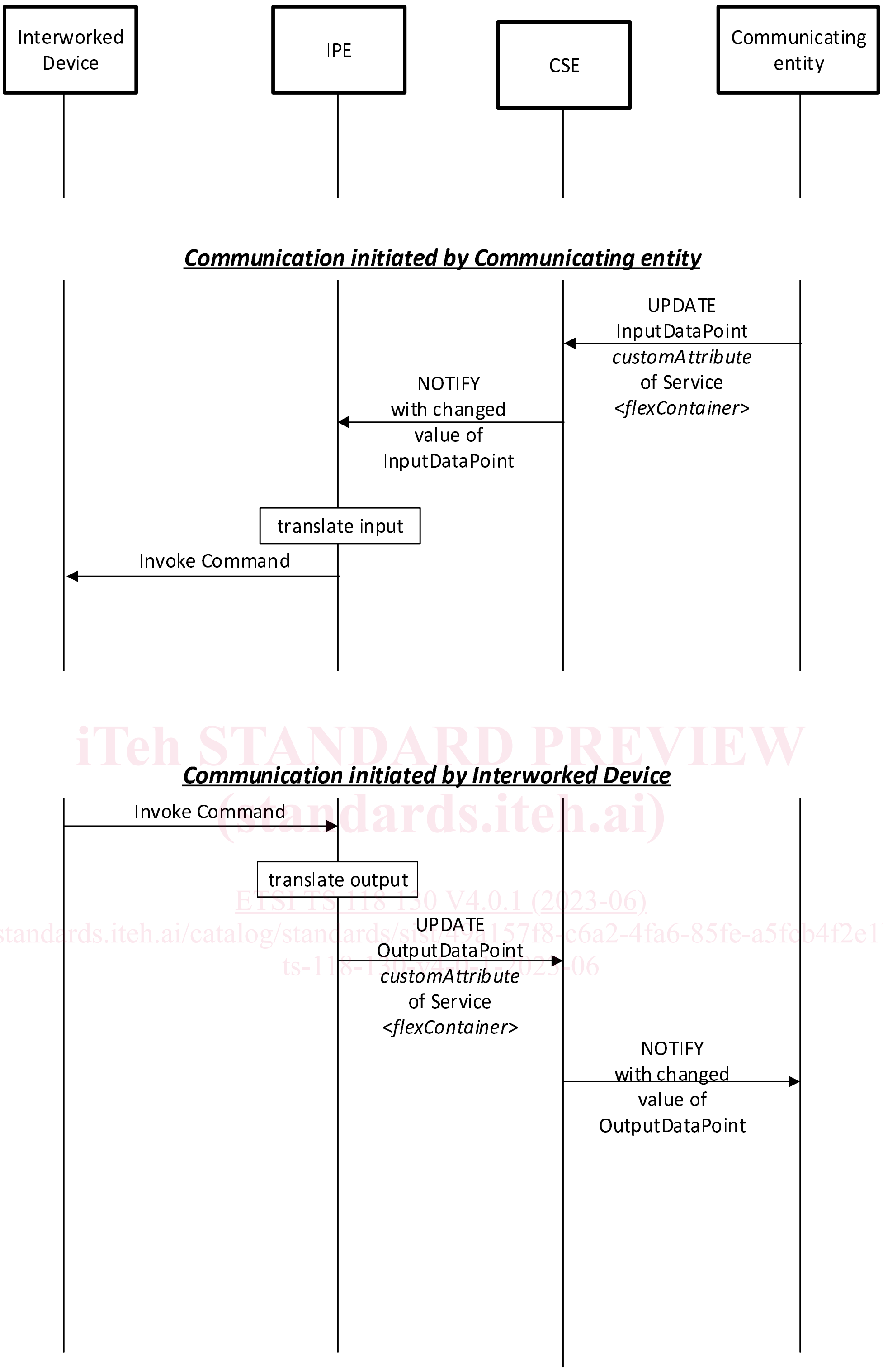

Figure 3: Data flow for an IPE involving dataPoints

When a communicating entity wants to invoke a command on the Interworked Device (e.g. an actuation command), using Datapoints of the related Service:

- The communicating entity UPDATEs the corresponding <*flexContainer*> that represents the service with the new value for *customAttribute* of the InputDatapoint.
- The CSE subsequently NOTIFY the IPE about the changed value for *customAttribute* of the InputDatapoint.
- The IPE invokes the command at the Interworked Device that sends the data of the InputDatapoint to the Interworked Device.

When the Interworked Device wants to invoke a command (e.g. a reporting) on the IPE - or a subscribed communicating entity, using Datapoints of the related Service:

- The Interworked Device invokes the command at the IPE that sends the data of the OutputDatapoint to the IPE.
- The IPE UPDATEs the corresponding <*flexContainer*> that represents the Service with the new value for *customAttribute* of that OutputDatapoint.
- The CSE subsequently NOTIFY subscribed communicating entities about the changed value for *customAttribute* of the OutputDatapoint.

5.3.3 Data flows for communicating with the IPE using Operations of a Service

Figure 4 show the data flows for communicating with the IPE using an Operation of a Service.

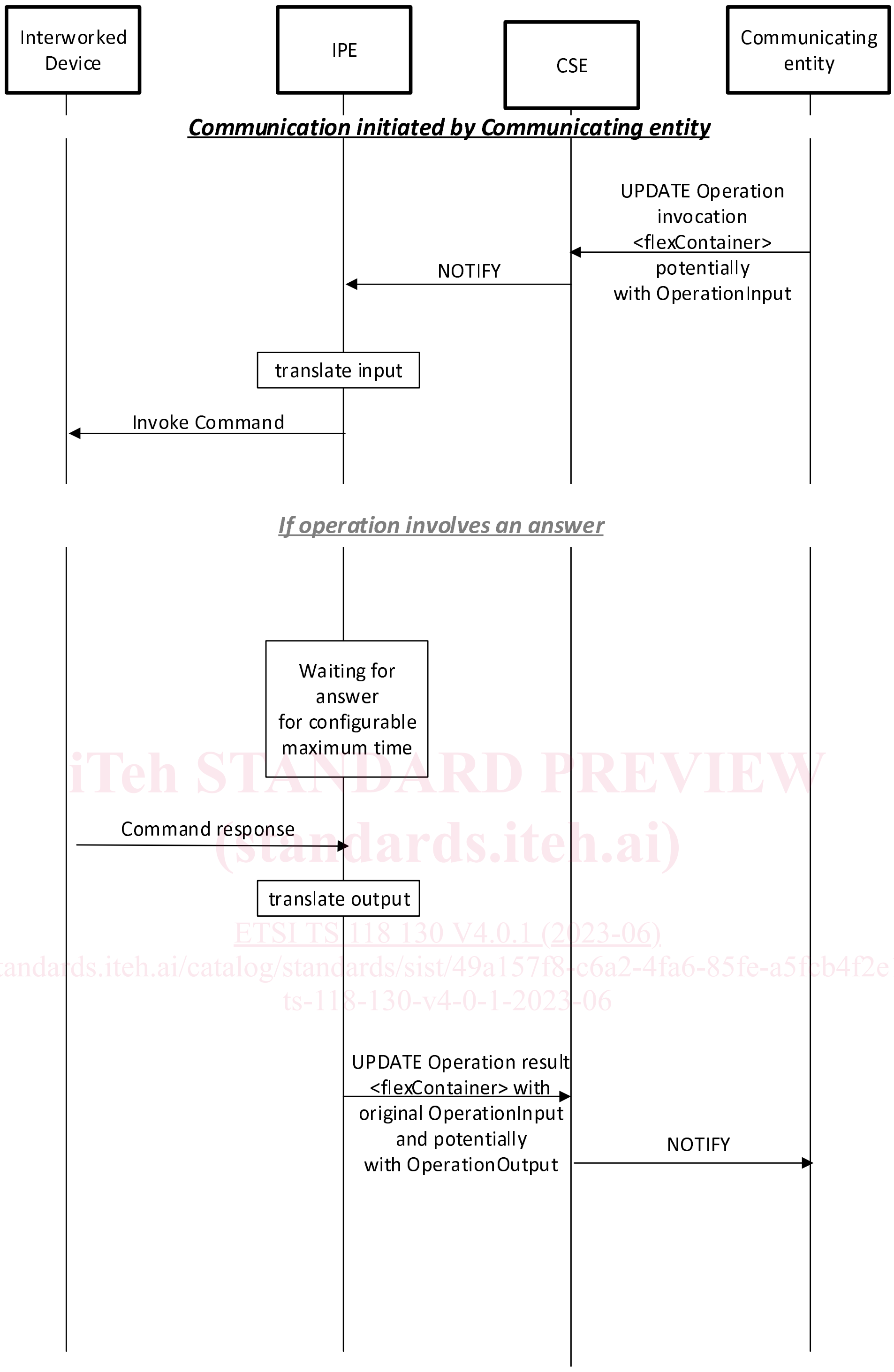

Figure 4: Data flow for a translating entity involving operations when initiated by a communicating entity

In contrast to DataPoints Operations allow grouping of input- and output parameters of a Command into a single transaction between the communicating entity and the target entity. It is permissible that no OperationInput and/or OperationOutput data exist for an Operation:

- When the communicating entity invokes an operation in the target entity it UPDATEs the <*flexContainer*> resource for the Operation invocation, potentially with OperationInput values for the related *customAttributes*. If the operation involves no OperationInput the UPDATE request contains no *customAttributes.*
- The CSE subsequently NOTIFY the IPE about the updated <*flexContainer*> resource for the Operation invocation, potetially with OperationInput values for the related *customAttributes*.