TTC技術レポート
Technical Report

TR-M2M-0045v2.0.0

Implementation semantics

2018 年 5 月 11 日制定

一般社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

TR-M2M-0045v2.0.0

Implementation semantics

＜参考＞ [Remarks]

１．国際勧告等の関連 [Relationship with international recommendations and standards]

本技術レポートは、oneM2M で作成された Technical Report 0045 (Version 2.0.0) に準拠している。

[This Technical Report is transposed based on the Technical Report 0045 (Version 2.0.0) developed by oneM2M.]

２．作成専門委員会 [Working Group]

oneM2M 専門委員会 [oneM2M Working Group]

oneM2M
Technical Report

Document Number	TR-0045- V2.0.0
Document Name:	Developer Guide: Implementing Semantics
Date:	2018-03-12
Abstract:	This developer guide is to describe how developer can quickly implement semantic functionnality of the release 2. The intended work is about a basic scenario describing the semantic annotation mechanisms of oneM2M using the semantic descriptor resources and semantic discovery.

Template Version: January 2017 (Do not modify)

About oneM2M

The purpose and goal of oneM2M is to develop technical specifications which address the need for a common M2M Service Layer that can be readily embedded within various hardware and software, and relied upon to connect the myriad of devices in the field with M2M application servers worldwide.

More information about oneM2M may be found at: http//www.oneM2M.org

Contents

1 Scope

The present document provides a simple use case for guiding application developers to develop applications using functionalities provided by a oneM2M service platform with the scope of as follows:

- Objective of the use case,
- The architecture of the use case mapped into a oneM2M service platform including semantic resources,
- The semantic modelling of the devices and the information according to a suitable ontology
- The execution procedures for implementation of the use case with a focus on the semantic aspects, and
- Implementation details of the use case using RDF (Resource Description Framework) for representing semantic descriptions and SPARQL queries on the RDF descriptions to identify fitting resources in the semantic resource discovery.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

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.

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.

NOTE: Available at: http://www.onem2m.org/images/files/oneM2M-Drafting-Rules.pdf.

[i.2] oneM2M TS-0011: "Common Terminology".

[i.3] oneM2M TS-0012: "Base Ontology".

[i.4] oneM2M TS-0030: "Generic Interworking".

[i.5] oneM2M TR-0025: "Application Developer Guide".

[i.6] oneM2M TS-0001: "Functional Architecture".

[i.7] oneM2M TS-0004: "Service Layer Core Protocol".

[i.8] oneM2M TS-0009: "HTTP Protocol Binding".

3 Definitions

For the purposes of the present document, the terms and definitions given in oneM2M TS-0011 [i.2] apply.

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 Motivation

The assumption of many existing oneM2M applications is that they interact with other oneM2M applications through known resource structures. They either create the resources themselves or are configured to use specific resources. Information is typically stored in containers, often as base64-encoded content instances, with the implicit assumption that applications have a-priori agreed on the syntax and semantics of this information.

Such an approach works well for small-scale and relatively static settings. When changes happen, the configuration will be updated manually. However, in more dynamic settings where the relevant resources frequently change, this becomes impractical. To satisfy those settings, relevant resources need to be discovered. Since Release 1, discovery of resources based on specific attributes and the use of labels has been made possible. The agreement of a fixed set of labels - which can only be combined using a logical OR operation in a discovery request - could be a viable solution.

For more heterogeneous, dynamic and larger scale scenarios, a more expressive approach for describing and discovering resources is needed. There are heterogeneous underlying technologies that can provide their information according to a different syntax, according to different units, e.g. Celsius, Fahrenheit and Kelvin. Those technologies may measure different aspects, e.g. indoor temperature, outdoor temperature, fridge temperature, etc., and the quality of the measurement may differ.

Another motivation of semantic annotations is to support the re-use of the same information by multiple applications. For example, in a smart city, applications may need to dynamically discover relevant resources according to multiple criteria at the same time - as sketched in the previous paragraph.

With semantic annotations, all the different aspects of IoT data can be described using RDF, which is a standardized semantic format. The vocabulary used for this description can be defined according to an ontology. With semantic discovery, applications can describe precisely what information they need or can deal with. This is powered by specifying a semantic filter using the SPARQL query language. The SPARQL filter is matched against the respective semantic annotation of each resource within the discovery scope, and the resource is included in the result of the discovery request only if the filter fits.

6 Use Case

This clause briefly describes a use case from the perspective of services provided by the oneM2M platform. The physical device components will be introduced in this clause.

In the use case, different sensor measure temperature information, but the temperature relates to different aspects, i.e. indoor, outdoor and fridge temperature and the temperature is measured in different scales. Simply discovering temperature does not solve the problem as the temperatures relate to different aspects. Through semantic annotation and semantic discovery of resources the resources that are relevant in a particular use case can be found.

An overview of the use case is shown in figure 6-1. The main components include:

- The temperature sensors are connected to a server.
- The server provides a set of services to enable the applications on smartphone #1 to discover and retrieve the relevant temperature information.
- The server provides a set of services to enable the applications on smartphone #2 to manage and semantically annotate the services providing temperature information.

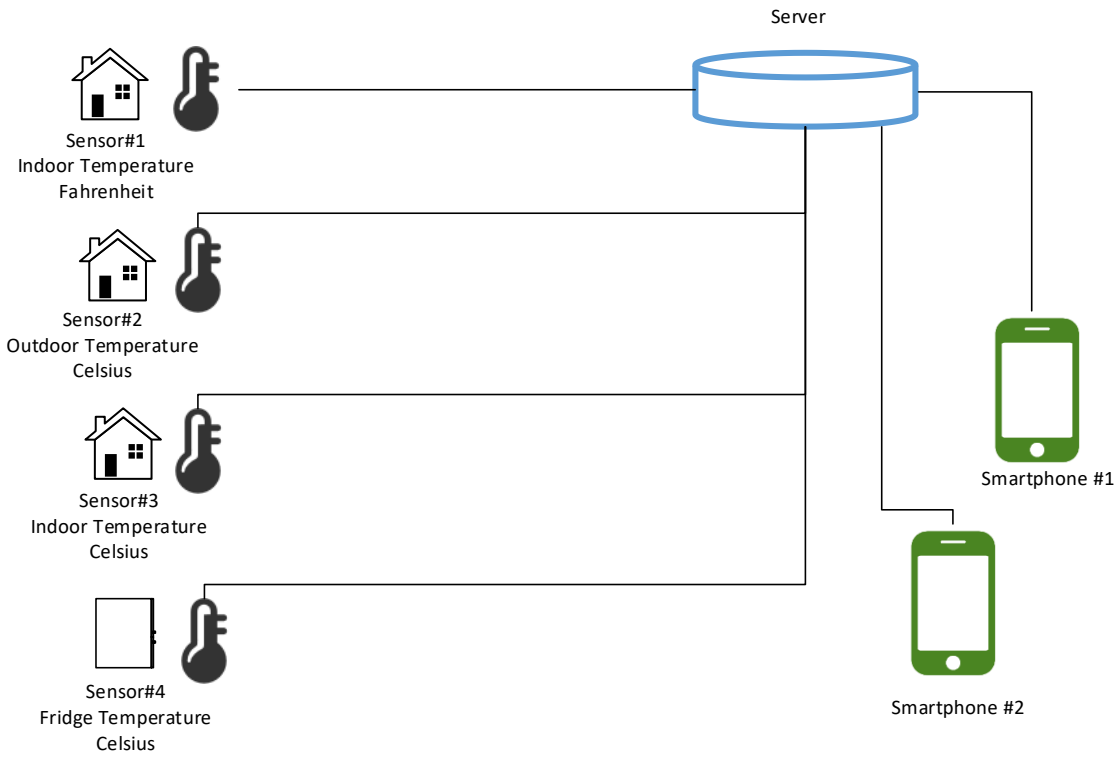

Figure 6-1: Overview of semantic annotation and discovery use case

7 Architecture Configuration

This clause describes the architecture of this use case with components represented by the oneM2M entity roles.

In the oneM2M, two basic types of entities are defined. One is an Application Entity (AE) and the other is a Common Services Entity (CSE). As shown in figure 7-1, the previous use case is modelled using oneM2M entities:

- Each sensor and the smartphone hosts an AE. The AE resides in the Application Dedicated Node is called ADN-AE.
- The server hosts an Infrastructure Node CSE (IN-CSE).

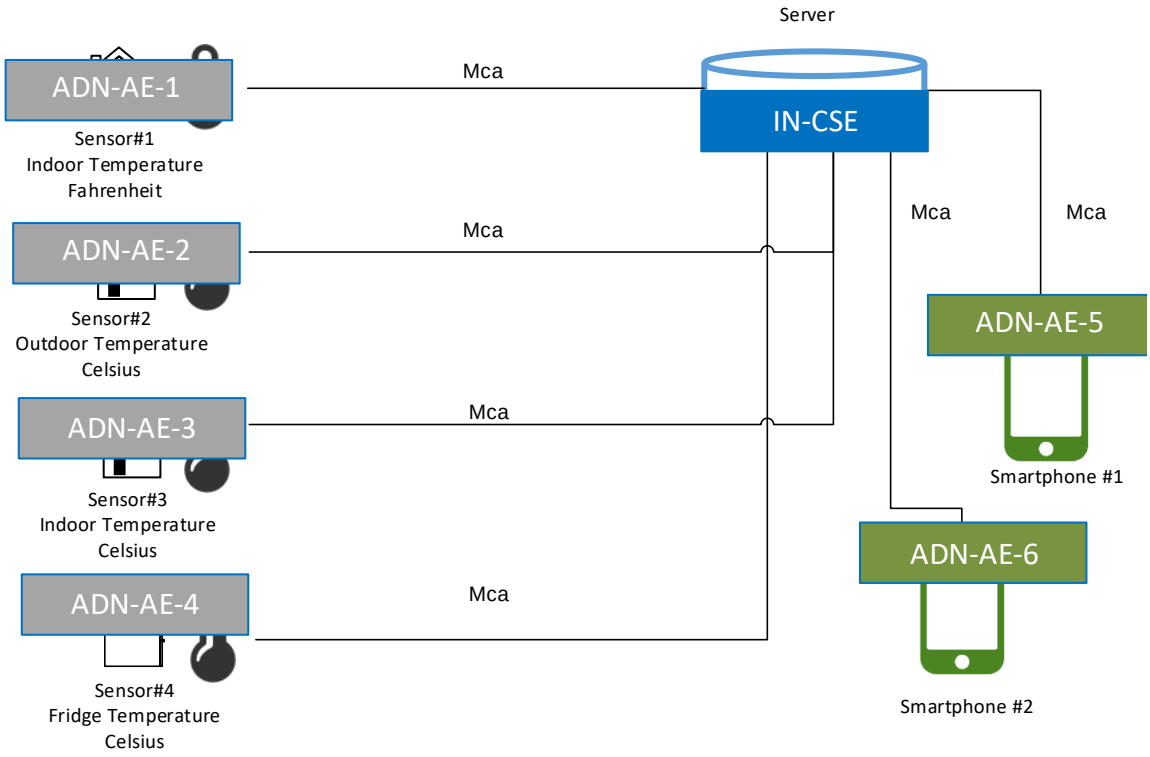

Figure 7-1: oneM2M functional architecture of semantic annotation and discovery use case

The oneM2M defined Mca reference point is used to interface an AE and CSE. Therefore, in this use case:

- The reference point used between temperature sensor AEs and the IN-CSE or Smartphone AEs and IN-CSE is Mca.

In summary, applications used in the current use case are classified as follows:

- ADN-AE-1: an application embedded in *Sensor#1* with capabilities to monitor *Sensor#1* and interact with the IN-CSE through the *Mca* reference point.
- ADN-AE-2: an application embedded in *Sensor#2* with capabilities to monitor *Sensor#2* and interact with the IN-CSE through *Mca* reference point.
- ADN-AE-3: an application embedded in *Sensor#3* with capabilities to monitor *Sensor#3* and interact with the IN-CSE through *Mca* reference point.
- ADN-AE-4: an application embedded in *Sensor#4* with capabilities to monitor *Sensor#4* and interact with the IN-CSE through *Mca* reference point.
- ADN-AE-5: a smartphone application embedded in the smartphone device with capabilities to interact directly with the oneM2M service platform IN-CSE through *Mca* reference point to discover and retrieve information related to *Sensor#1*, *Sensor#2*, *Sensor#3* and *Sensor#4*.
- ADN-AE-6: a smartphone application embedded in the smartphone device with capabilities to interact directly with the oneM2M service platform IN-CSE through Mca reference point to manage and semantically annotate resources related to *Sensor#1*, *Sensor#2*, *Sensor#3* and *Sensor#4*.

8 Semantic Modelling

The semantic modelling is typically based on an ontology that specifies classes and properties, i.e. the ontology defines the vocabulary to be used for the semantic description. In Figure 8-1, a simple ontology for the semantic annotation and discovery use case is visualized. Table 8-1 shows the actual OWL (Web Ontology Language) ontology in an RDF representation.

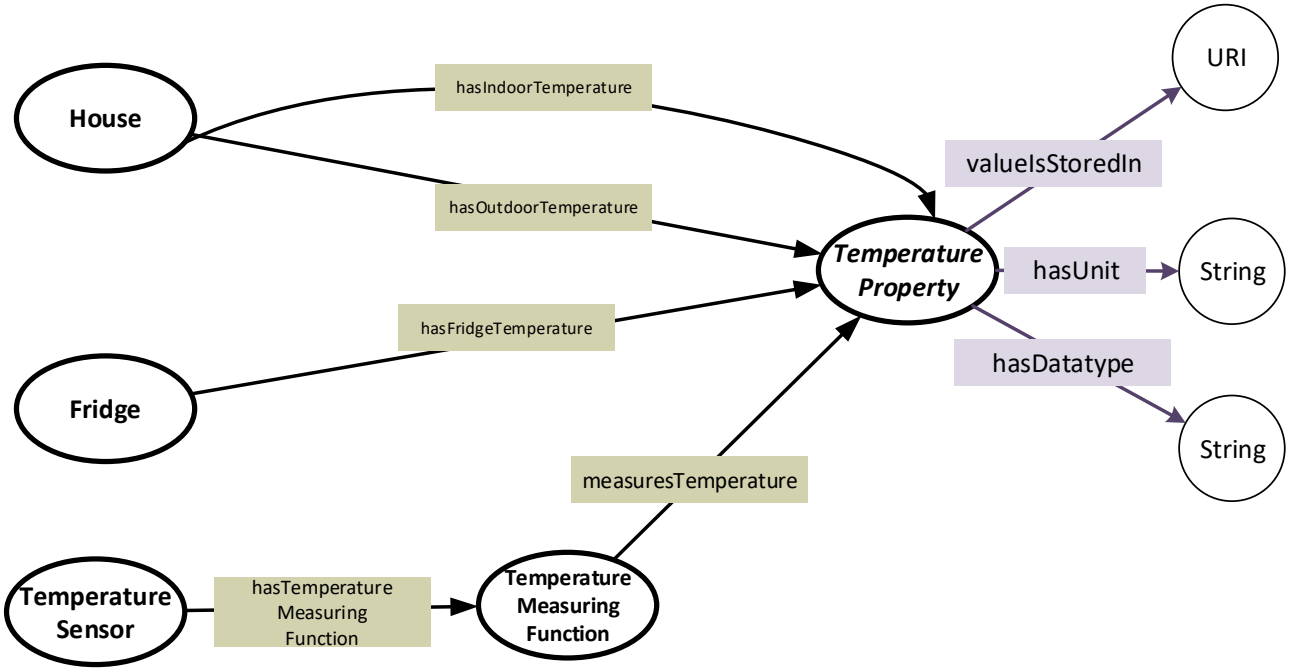

Figure 8-1: Simple ontology for the semantic annotation and discovery use case

Table 8-1: OWL/RDF representation of the simple ontology for the semantic annotation and discovery use case

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.onem2m.org/ontology/temperature_example#"
     xml:base="http://www.onem2m.org/ontology/temperature_example"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Ontology rdf:about="http://www.onem2m.org/ontology/temperature_example"/>

    <!-- Classes -->
    <owl:Class rdf:about="http://www.onem2m.org/ontology/temperature_example#Fridge"/>
    <owl:Class rdf:about="http://www.onem2m.org/ontology/temperature_example#House"/>
    <owl:Class
rdf:about="http://www.onem2m.org/ontology/temperature_example#TemperatureMeasuringFunction"/>
    <owl:Class
rdf:about="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
    <owl:Class rdf:about="http://www.onem2m.org/ontology/temperature_example#TemperatureSensor"/>

    <!-- Object Properties -->
    <owl:ObjectProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#hasFridgeTemperature">
        <rdfs:domain rdf:resource="http://www.onem2m.org/ontology/temperature_example#Fridge"/>
        <rdfs:range
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#hasIndoorTemperature">
        <rdfs:domain rdf:resource="http://www.onem2m.org/ontology/temperature_example#House"/>
        <rdfs:range
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#hasOutdoorTemperature">
        <rdfs:domain rdf:resource="http://www.onem2m.org/ontology/temperature_example#House"/>
        <rdfs:range
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#hasTemperatureMeasuringFunction">
        <rdfs:domain
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureSensor"/>
```

```
        <rdfs:range
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureMeasuringFunction"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#measuresTemperature">
        <rdfs:domain
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureMeasuringFunction"/>
        <rdfs:range
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
    </owl:ObjectProperty>

    <!-- Data properties -->
    <owl:DatatypeProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#hasDatatype">
        <rdfs:domain
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:about="http://www.onem2m.org/ontology/temperature_example#hasUnit">
        <rdfs:domain
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty
rdf:about="http://www.onem2m.org/ontology/temperature_example#valueIsStoredIn">
        <rdfs:domain
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureProperty"/>
        <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#anyURI"/>
    </owl:DatatypeProperty>

</rdf:RDF>
```

Annex A shows how the simple ontology for the semantic annotation and discovery use case can be mapped to the oneM2M Base Ontology [i.3]. As a result, it is semantically integrated with the core concepts that have been identified for oneM2M itself. Such a mapping may be helpful to semantically relate to other parts of the oneM2M system, but is not required, i.e. depending on what is to be achieved, the above ontology can be used "stand-alone" as the basis for supporting semantic annotation and discovery in a particular use case as shown in this developer guide.

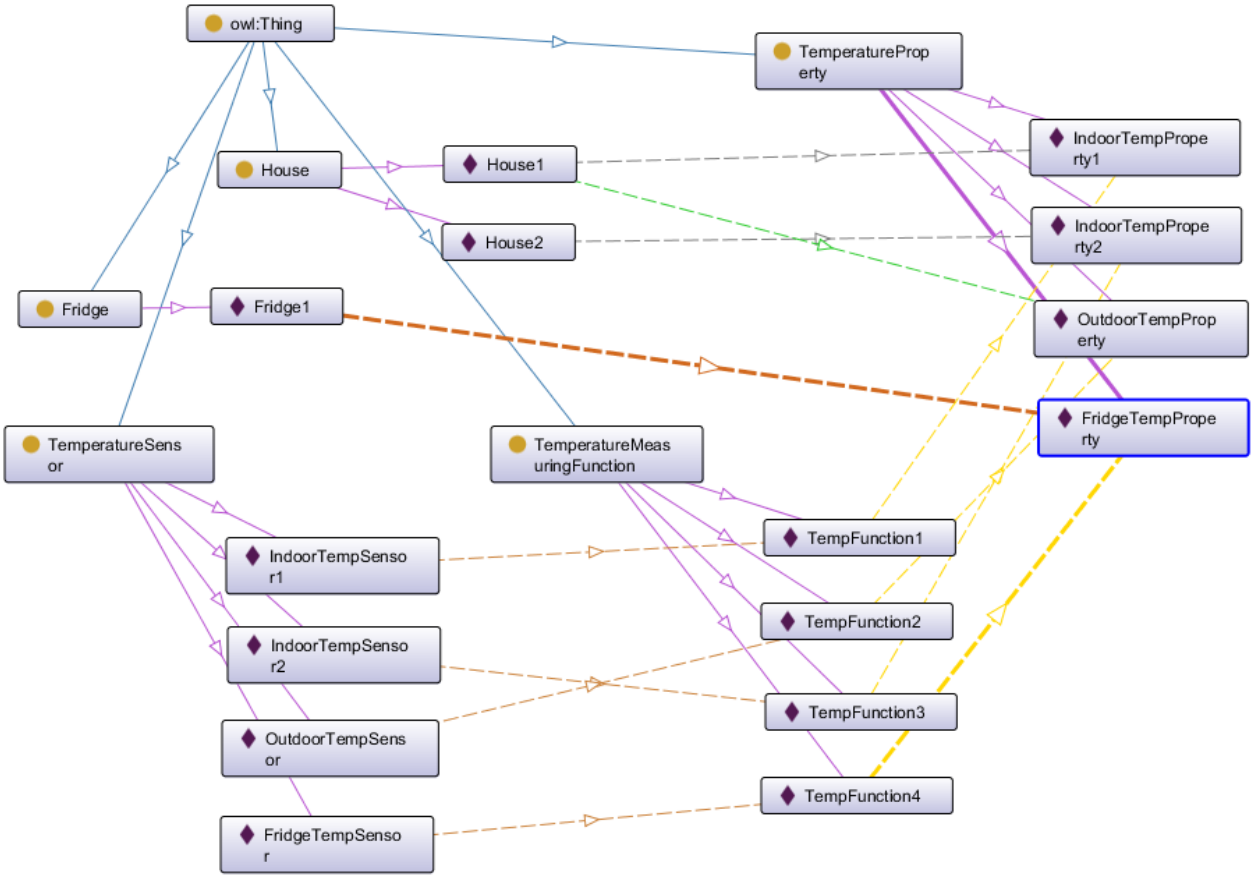

Figure 8-2: Semantic description of the scenario according to the ontology

Figure 8-2 shows all the instances needed for modelling the semantic annotation and discovery use case, their respective types and their relations (in different colours and styles (e.g. dashed), but without annotations indicating the name of the respective relation). Each sensor will be represented by an AE and all the semantic instances related to the given sensor will be contained in the semantic descriptor. For example, IndoorTempSensor1 models the sensor, which has a function called TempFunction1, which measures IndoorProperty1, which in turn describes the indoor temperature of House 1. In addition there are some datatype properties like the unit of measurement, which are not shown in this figure.

9 Procedures

9.1 Introduction

In order to implement the presented use case based on oneM2M, required procedures are classified and shown below. This clause focuses on the core semantic aspects. Typical common aspects such as the creation of appropriate access control policies are not included. The interested reader is referred to other developer guides, e.g. oneM2M TR-0025 [i.5].

1) **Application Registration:** This procedure contains sensor application registration and smartphone application registration.

2) **Initial resource creation:** This procedure contains container resources creation and contentInstance resources creation.

3) **Semantic descriptor resource creation:** This procedure shows the creation of semantic descriptor resources by a smartphone application that is annotating the resources created earlier by the sensor applications.

4) **Semantic discovery of relevant sensor resources:** This procedure shows how all sensor applications that fit the semantic filter criteria specified in SPARQL are discovered by a smartphone application.

5) **Retrieval of semantic descriptor and sensor information:** This procedure shows how the smartphone application first retrieves the semantic descriptor resources of the discovered resource and then, based on the information contained in the semantic descriptor, retrieves the contentInstance resource containing the latest sensor reading.

9.2 Call Flows

9.2.1 Application registration

Figure 9.2.1-1 depicts how the applications register with oneM2M and can be described as follows:

1) Sensor applications (ADN-AE1, ADN-AE2, ADN-AE3 and ADN-AE4) register with the oneM2M service platform (IN-CSE).

2) Smartphone applications (ADN-AE5 and ADN-AE6) register with the oneM2M service platform (IN-CSE).

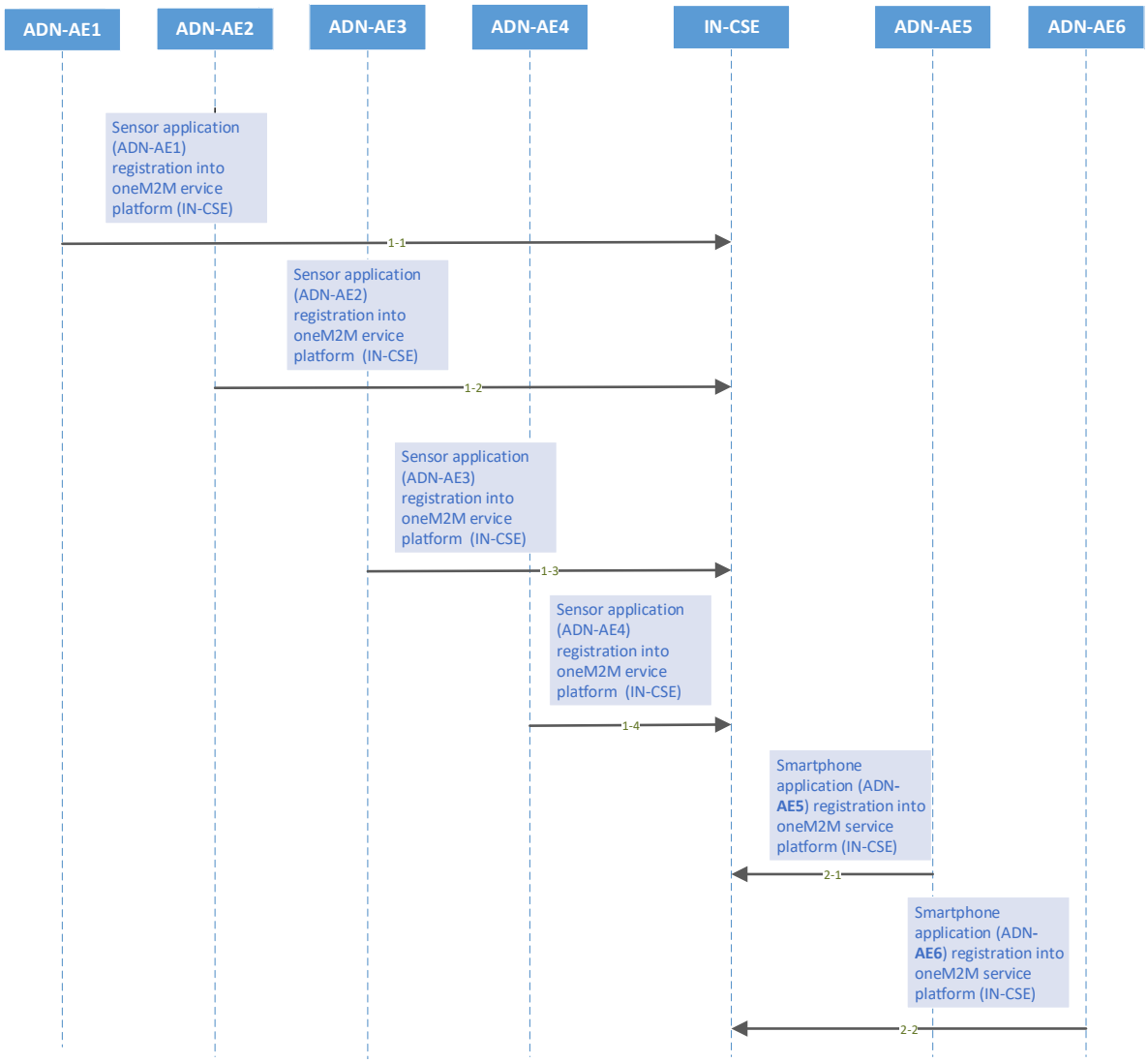

Figure 9.2.1-1: Registration phase call flows

9.2.2 Initial resource creation

Call flows regarding the initial resource creation phase depicted in figure 9.2.2-1 can be described as follows:

1) Four container resources are created in the oneM2M service platform (IN-CSE) to store the temperature values measured by Sensor#1, Sensor#2, Sensor#3 and Sensor#4 under the registered application ADN-AE1, ADN-AE2, ADN-AE3 and ADN-AE4, respectively (in figure 9.2.2-1 this is only shown for ADN-AE1 and ADN-AE2, ADN-AE3 and ADN-AE4 are handled in exactly the same way).

2) Content Instance resources are created by the applications (ADN-AE1, ADN-AE2, ADN-AE3 and ADN-AE4) under each created container and represent the measured temperatures (in figure 9.2.2-1 this is only shown for the respective first contentInstances of ADN-AE1 and ADN-AE2, ADN-A3 and ADN-AE4 are handled in exactly the same way and contentInstances are created continuously).

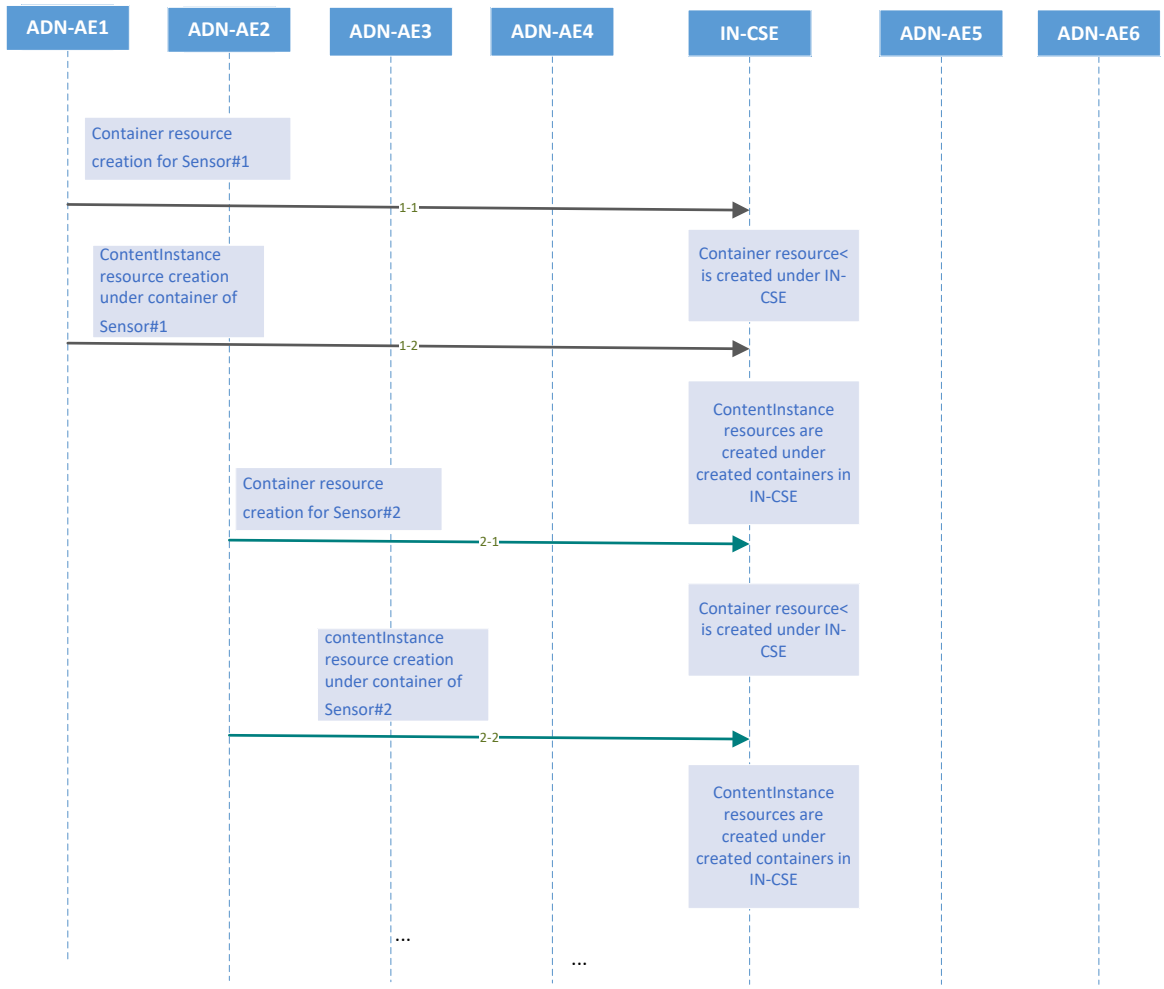

Figure 9.2.2-1: Initial resource creation phase call flows

9.2.3 Semantic descriptor resource creation

Call flows regarding the semantic descriptor resource creation phase are depicted in figure 9.2.3-1 and can be described as follows:

1) ADN-AE5 creates a semantic descriptor resource under each of the ADN resources representing a sensor application instance (i.e. ADN-AE1, ADN-AE2, ADN-AE3 and ADN-AE4).The semantic information has been provided by a human user using the smartphone application. It provides information about the sensor, the sensor measurements (e.g. unit of measurement) and what is being measured. In the given example these are the information measured by the sensor (temperature), the unit of measurement (Celsius or Fahrenheit), and the relation to the Thing for which the value is being measured (house or fridge).

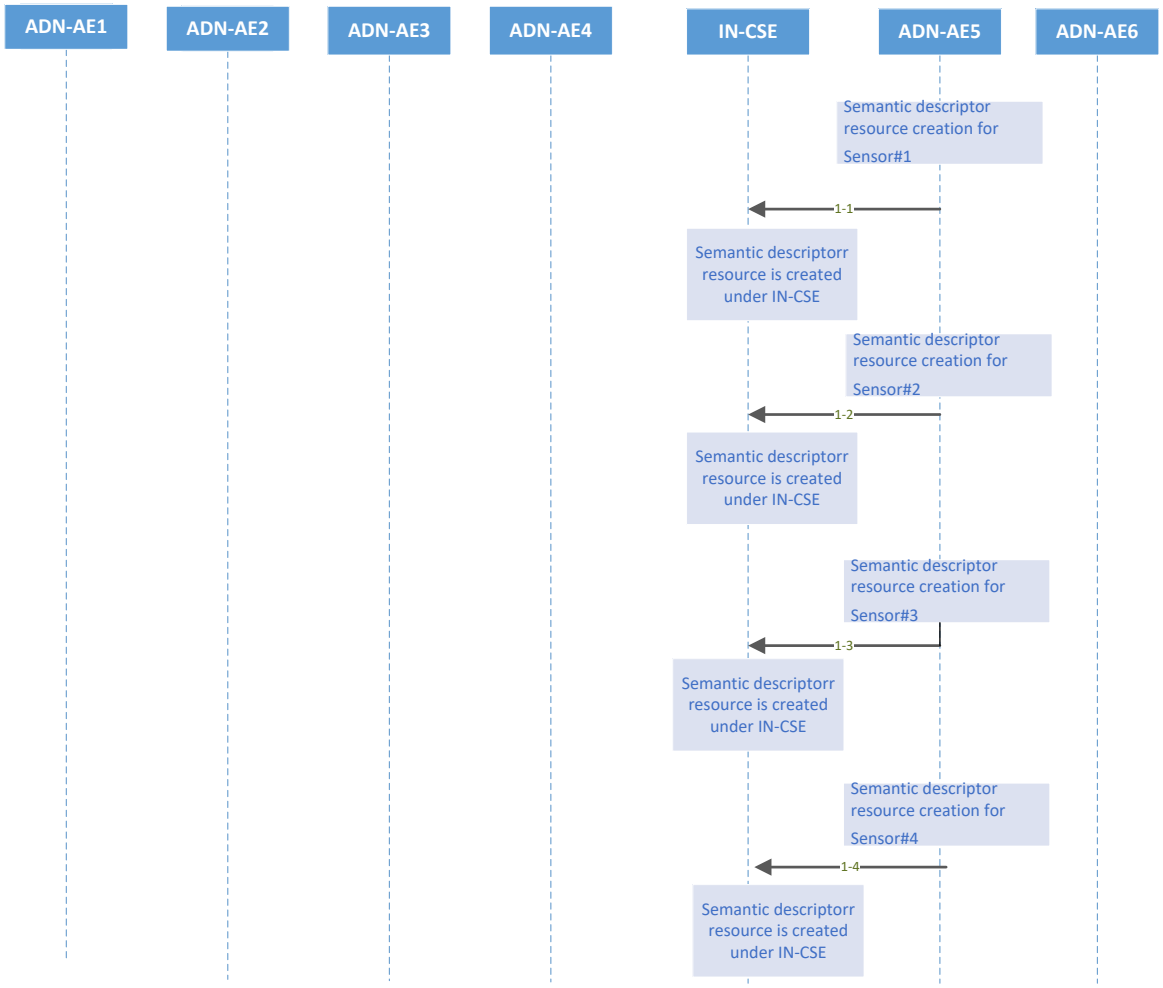

Figure 9.2.3-1: Semantic descriptor resource creation phase call flows

9.2.4 Semantic discovery of relevant sensor resources

Call flows regarding the discovery are depicted in figure 9.2.4-1 and can be described as follows:

1) Smartphone application (ADN-AE6) sends a RETRIEVE request including the parameter *filterUsage* and specific semantic filter criteria condition(s) provided in SPARQL for discovery of sensor application resources stored in the oneM2M service platform (IN-CSE).

2) The oneM2M service platform (IN-CSE) responds with URIs of the discovered sensor resources, if any, to the smartphone application (ADN-AE6) according to the filter criteria.

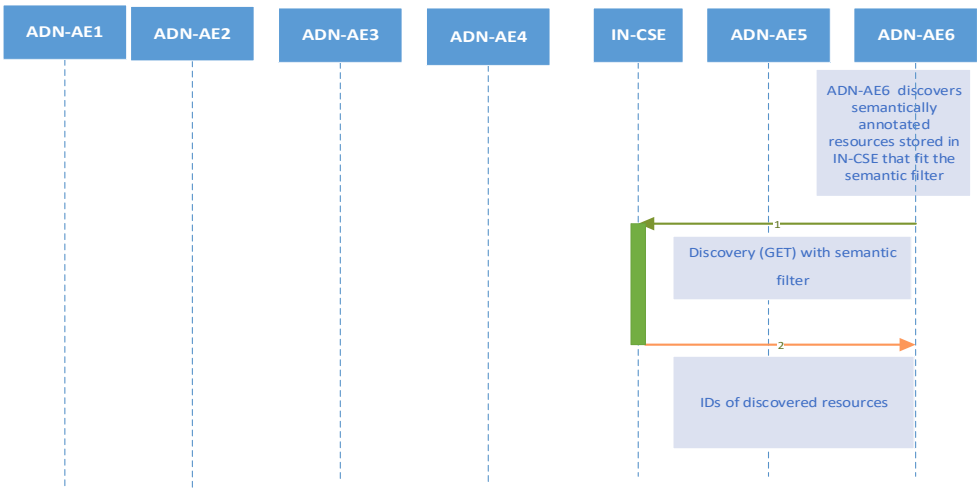

Figure 9.2.4-1: Semantic discovery phase call flows

9.2.5 Retrieval of semantic descriptor and sensor information

Call flows regarding the retrieval of the semantic descriptor and the contentInstance resources are depicted in figure 9.2.5-1 and can be described as follows:

1) The smartphone application (ADN-AE6) sends GET requests for retrieval of the semantic descriptor child resources of the discovered sensor resources.

2) The oneM2M service platform (IN-CSE) returns the requested semantic descriptor child resources.

3) Based on the respective information contained in the retrieved semantic descriptor, the smartphone application (ADN-AE6) requests the latest content instance provided by the sensor application containing the sensor measurement.

4) The oneM2M service platform (IN-CSE) returns the requested content instance containing the sensor information.

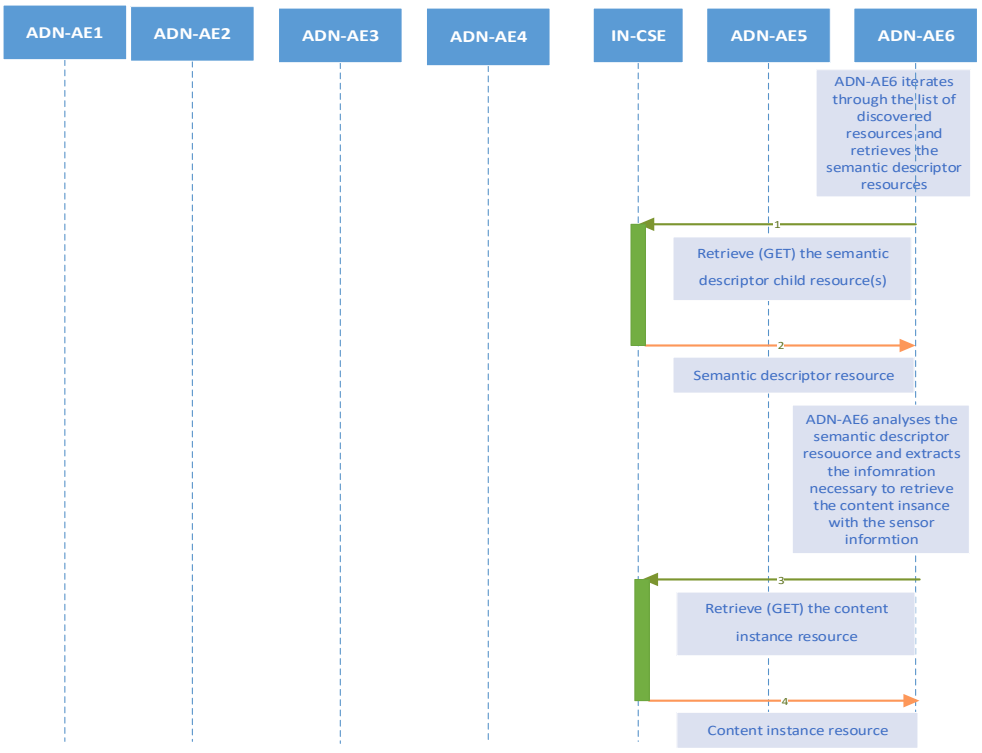

Figure 9.2.5-1: Retrieval of semantic descriptor and sensor information phase call flows

10 Implementation

10.1 Introduction

Clause 10 presents necessary procedures required for the implementation of the semantic annotation and discovery use case, including conditions to be met for the correct implementation of the current use case, and resource tree etc.

10.2 Assumptions

Assumptions are presented as below in order to ensure the semantic annotation and discovery use case can be correctly implemented:

- Security is not considered in the current use case.
- HTTP binding of oneM2M primitives is used in the current use case.
- XML serializations of oneM2M primitives are used in the current use case.
- All mandatory HTTP headers are presented in the HTTP requests while optional headers are selectively used in the current use case.
- All mandatory resource attributes for resources presented in the current use case are presented in the HTTP requests while optional resource attributes are selectively used in the current use case.
- All AEs in the current use case are initially registered with the IN-CSEs and the identifier of the AEs are assigned by it acting as the Registrar CSE of the AE.
- All resources created in the current use case are addressable with the oneM2M Resource Identifier form of *Hierarchical address*.

- Short names for the representation of the resources and attributes are used in the current use case.
- Default access control policy has already been created under IN-CSE.
- All request originators send *Blocking Requests* for accessing resources located in CSEs.

10.3 Resource Structure

10.3.1 Introduction

The development of an oneM2M application includes the design of the resource trees of service capability layers, which is the IN-CSE in the current use case. The resource tree is constructed with child resources created according to the high level procedures presented in clause 9. All the child resources shown in the resource trees are mandatorily required in order to correctly implement the semantic annotation and discovery use case.

10.3.2 Resource Structure of IN-CSE

The resource tree of IN-CSE starts with a CSEBase named ***server*** depicted in figure 10.3.2-1.

The root CSEBase has four direct child resources directly relevant to the semantic annotation and discovery use case, which are the tempsensorae1, tempsensorae2, tempsensorae3 and tempsensorae4 resources, representing ADN-AE1, ADN-AE2, ADN-AE3, and ADN-AE4, respectively. Each AE will have a container named temperature where the content instances (CI1 to CIN) are stored, and a semantic descriptor named SD1. As the substructure of each AE resource is exactly the same, only the substructure of tempsensorae1 is shown.

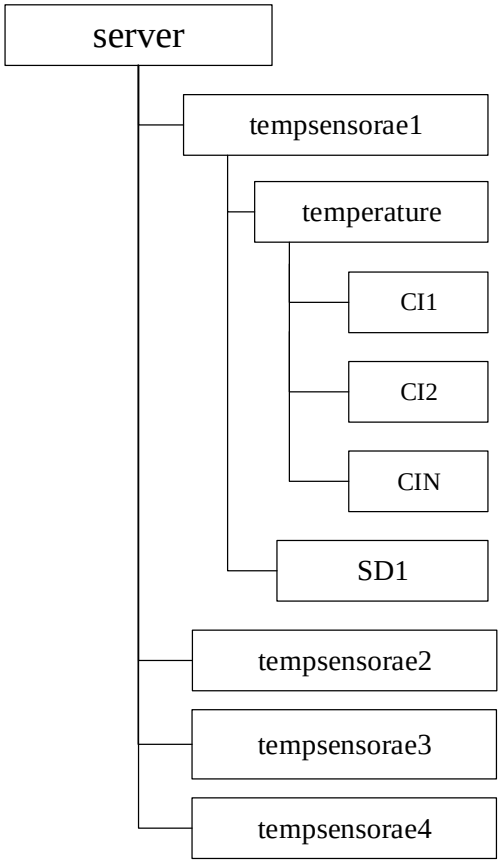

Figure 10.3.2-1: IN-CSE resource structure

10.4 Role of Entities

10.4.1 oneM2M service platform (IN-CSE)

The oneM2M service platform is modelled as an IN-CSE and is responsible for

- handling the registration requests from all ADN-AEs in the use case;
- creating and managing the resource structures for storing sensor information and semantic annotations;
- executing semantic resource discovery.

10.4.2 Temperature sensor applications (ADN-AE1, ADN-AE2, ADN-AE3 and ADN-AE4)

Each of the temperature sensor applications are modelled as an ADN-AE and are responsible for:

- measuring the sensor information;
- registering the temperature sensors with the IN-CSE;
- creating container resources named "temperature";
- creating content instance resources under the container "temperature" that contain the temperature measurements.

10.4.3 Semantic Annotation Application (ADN-AE5)

The semantic annotation application, which is expected to run on a smartphone or other device with user interface is responsible for the following:

- enable the user to create semantic annotations for resources representing sensors;
- create semantic descriptor resources as child resources of sensor AE resources that contain the semantic annotations.

10.4.4 Semantic Discovery Application (ADN-AE6)

The semantic discovery application, which is expected to run on a user device, is responsible for the following:

- enable the user to formulate the semantic filter for the semantic discovery (in SPARQL);
- retrieve the semantic annotation for discovered resources;
- retrieve sensor measurements based on the semantic annotations.

10.5 Implementation Procedures

10.5.1 Introduction

The implementation procedures in the current use case are mapped into HTTP bindings with XML serializations of oneM2M primitives according to the standard APIs describing the reference points Mca and Mcc, as defined in oneM2M TS-0001 [i.6], oneM2M TS-0004 [i.7], and the HTTP binding oneM2M TS-0009 [i.8].

In addition, *short names* for the representation of the resources and attributes are used in the implementation procedures.

10.5.2 Application registration

The following example shows how the tempsensorae1 (ADN-AE) is registered with the IN-CSE. The same is done for tempsensorae2, tempsensorae3 and tempsensorae4, which is not explicitly shown here.

```
HTTP Request:

POST /server?rcn=0 HTTP/1.1
Host: in.provider.com:7579
X-M2M-RI: 12345
X-M2M-Origin: SOrigin
Content-Type: application/xml;ty=2

<?xml version="1.0" encoding="UTF-8"?>
<m2m:ae mlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="tempsensorae1">
    <api>0.2.481.2.0001.001.000111</api>
    <lbl>semanticallyAnnotated tempSensor</lbl>
    <rr>true</rr>
</m2m:ae>

HTTP Response:

201 Created

X-M2M-RI: 12345
X-M2M-RSC: 2001
Content-Location: /server/ae-349923453
```

10.5.3 Initial resource creation

The following example shows how the temperature container child resource of tempsensorae1 is created, which will hold the content instances with the measured sensor values. The same is done for the temperature container child resources of tempsensorae2, tempsensorae3 and tempsensorae4, which is not explicitly shown here.

```
HTTP Request:

POST /server/tempsensorae1?rcn=0 HTTP/1.1
Host: in.provider.com:7579
X-M2M-RI:12346
X-M2M-Origin:SOrigin
Content-Type:application/xml;ty=3

<?xml version="1.0" encoding="UTF-8"?>
<m2m:cnt xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="temperature">
    <lbl>container</lbl>
</m2m:cnt>

HTTP Response:

201 Created
X-M2M-RI: 12346
X-M2M-RSC: 2001
Content-Location: /server/cnt-282750912
```

The example below shows how a content instance holding a temperature value is created in the temperature container.

```
HTTP Request:

POST /server/tempsensorae1/temperature?rcn=0 HTTP/1.1
Host: in.provider.com:7579
Accept:application/xml
X-M2M-RI:12347
X-M2M-Origin:SOrigin
Content-Type:application/xml; ty=4

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<m2m:cin xmlns:m2m="http://www.onem2m.org/xml/protocols"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <con>75</con>
</m2m:cin>

HTTP Response:

201 Created
X-M2M-RI: 12347
X-M2M-RSC: 2001
Content-Location: /server/cin-234523664
```

10.5.4 Semantic descriptor resource creation

In the semantic annotation and discovery use case, a semantic annotation user application, e.g. a smartphone app, is used for semantically annotating a given resource. For this purpose, the semantic annotation application (ADN-AE5) creates a semantic descriptor child resource of the resource to be semantically annotated. The actual semantic description is represented in base64-encoded RDF/XML, as indicated by the descriptorRepresentation attribute (dcrp), and can be found in the descriptor attribute of the semantic descriptor resource.

According to the scenario, there is one AE representing each temperature sensor. In the semantic description, all relevant information regarding the sensor and what it is measuring is represented, e.g. that the unit of measurement is Celsius, the temperature value is represented as an integer, the value represents the indoor temperature of a particular house and how the latest value can be retrieved. Figure 8-2 shows the complete semantic information of the scenario combined, the semantic annotations listed below each contain the relevant information for one temperature sensor. In base64-encoded form they are stored in the descriptor attributes of the respective semantic descriptor resources.

Semantic descriptor of indoor temperature sensor 1:

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.onem2m.org/ontology/houses_temperature_example#"
     xml:base="http://www.onem2m.org/ontology/houses_temperature_example"

xmlns:temperature_example="http://www.onem2m.org/ontology/temperature_example#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#House1">
        <rdf:type
rdf:resource="http://www.onem2m.org/ontology/temperature_example#House"/>
        <temperature_example:hasIndoorTemperature
rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#IndoorT
empProperty1"/>
    </owl:NamedIndividual>

    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#IndoorTemp
Property1">
        <rdf:type
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperaturePro
perty"/>
      <temperature_example:hasDatatype>xsd:int</temperature_example:hasDatatype>
      <temperature_example:hasUnit
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Fahrenheit</temperature_
example:hasUnit>
      <temperature_example:valueIsStoredIn>http://
in.provider.com:7579/server/tempsensorae1/temperature/latest</temperature_examp
le:valueIsStoredIn>
    </owl:NamedIndividual>

    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#IndoorTemp
Sensor1">
        <rdf:type
```

```
        rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureSen
        sor"/>
                <temperature_example:hasTemperatureMeasuringFunction
        rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#TempFun
        ction1"/>
            </owl:NamedIndividual>

            <owl:NamedIndividual
        rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#TempFuncti
        on1">
                <rdf:type
        rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureMea
        suringFunction"/>
                <temperature_example:measuresTemperature
        rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#IndoorT
        empProperty1"/>
            </owl:NamedIndividual>
        </rdf:RDF>
```

Semantic descriptor of outdoor temperature sensor:

```
        <?xml version="1.0"?>
        <rdf:RDF xmlns="http://www.onem2m.org/ontology/houses_temperature_example#"
             xml:base="http://www.onem2m.org/ontology/houses_temperature_example"

        xmlns:temperature_example="http://www.onem2m.org/ontology/temperature_example#"
             xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:owl="http://www.w3.org/2002/07/owl#"
             xmlns:xml="http://www.w3.org/XML/1998/namespace"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
             xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

            <owl:NamedIndividual
        rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#House1">
                <rdf:type
        rdf:resource="http://www.onem2m.org/ontology/temperature_example#House"/>
                <temperature_example:hasOutdoorTemperature
        rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#Outdoor
        TempProperty"/>
            </owl:NamedIndividual>

            <owl:NamedIndividual
        rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#OutdoorTem
        pProperty">
                <rdf:type
        rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperaturePro
        perty"/>
              <temperature_example:hasDatatype>xsd:int</temperature_example:hasDatatype>
               <temperature_example:hasUnit
    rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Celsius</temperature_example:ha
    sUnit>

    <temperature_example:valueIsStoredIn>http://in.provider.com:7578/server/tempsensorae2/
    temperature/latest</temperature_example:valueIsStoredIn>
            </owl:NamedIndividual>

            <owl:NamedIndividual
        rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#OutdoorTem
        pSensor">
                <rdf:type
        rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureSen
        sor"/>
                <temperature_example:hasTemperatureMeasuringFunction
        rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#TempFun
        ction2"/>
            </owl:NamedIndividual>

            <owl:NamedIndividual
        rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#TempFuncti
        on2">
```

```
            <rdf:type
        rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureMea
        suringFunction"/>
            <temperature_example:measuresTemperature
        rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#Outdoor
        TempProperty"/>
        </owl:NamedIndividual>
    </rdf:RDF>
```

Semantic descriptor of indoor temperature sensor 2:

```
    <?xml version="1.0"?>
    <rdf:RDF xmlns="http://www.onem2m.org/ontology/houses_temperature_example#"
         xml:base="http://www.onem2m.org/ontology/houses_temperature_example"

    xmlns:temperature_example="http://www.onem2m.org/ontology/temperature_example#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:owl="http://www.w3.org/2002/07/owl#"
         xmlns:xml="http://www.w3.org/XML/1998/namespace"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

        <owl:NamedIndividual
    rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#House2">
            <rdf:type
    rdf:resource="http://www.onem2m.org/ontology/temperature_example#House"/>
            <temperature_example:hasIndoorTemperature
    rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#IndoorT
    empProperty2"/>
        </owl:NamedIndividual>

        <owl:NamedIndividual
    rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#IndoorTemp
    Property2">
            <rdf:type
    rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperaturePro
    perty"/>
      <temperature_example:hasDatatype>xsd:double</temperature_example:hasDatatype>
      <temperature_example:hasUnit
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Celsius</temperature_example:hasUn
it>

<temperature_example:valueIsStoredIn>http://in.provider.com:7579/server/tempsensorae3/tem
perature/latest</temperature_example:valueIsStoredIn>
        </owl:NamedIndividual>

        <owl:NamedIndividual
    rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#IndoorTemp
    Sensor2">
            <rdf:type
    rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureSen
    sor"/>
            <temperature_example:hasTemperatureMeasuringFunction
    rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#TempFun
    ction3"/>
        </owl:NamedIndividual>

        <owl:NamedIndividual
    rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#TempFuncti
    on3">
            <rdf:type
    rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureMea
    suringFunction"/>
            <temperature_example:measuresTemperature
    rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#IndoorT
    empProperty2"/>
        </owl:NamedIndividual>
    </rdf:RDF>
```

Semantic descriptor of fridge temperature sensor 1:

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.onem2m.org/ontology/houses_temperature_example#"
     xml:base="http://www.onem2m.org/ontology/houses_temperature_example"

xmlns:temperature_example="http://www.onem2m.org/ontology/temperature_example#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#Fridge1">
        <rdf:type
rdf:resource="http://www.onem2m.org/ontology/temperature_example#Fridge"/>
        <temperature_example:hasFridgeTemperature
rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#FridgeT
empProperty"/>
    </owl:NamedIndividual>

    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#FridgeTemp
Property">
        <rdf:type
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperaturePro
perty"/>
  <temperature_example:hasDatatype>xsd:double</temperature_example:hasDatatype>
  <temperature_example:hasUnit
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Celsius</temperature_exa
mple:hasUnit>

<temperature_example:valueIsStoredIn>http://in.provider.com:7579/server/tempsen
sorae4/temperature/latest</temperature_example:valueIsStoredIn>
    </owl:NamedIndividual>

    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#FridgeTemp
Sensor">
        <rdf:type
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureSen
sor"/>
        <temperature_example:hasTemperatureMeasuringFunction
rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#TempFun
ction4"/>
    </owl:NamedIndividual>


    <owl:NamedIndividual
rdf:about="http://www.onem2m.org/ontology/houses_temperature_example#TempFunctio
on4">
        <rdf:type
rdf:resource="http://www.onem2m.org/ontology/temperature_example#TemperatureMea
suringFunction"/>
        <temperature_example:measuresTemperature
rdf:resource="http://www.onem2m.org/ontology/houses_temperature_example#FridgeT
empProperty"/>
    </owl:NamedIndividual>
</rdf:RDF>
```

The following example shows how the semantic descriptor child resource of the tempsensorae1 (ADN-AE1) is created. The semantic descriptors of tempsensorae2, tempsensorae3 and tempsensorae4 are created in the same way. Please note that the content of the descriptor attribute first needs to be base64-enoded.

```
HTTP Request:

POST /server/tempsensorae1?rcn=0 HTTP/1.1
Host: in.provider.com:7579
```

```
X-M2M-RI:12349
X-M2M-Origin:SOrigin
Content-Type:application/xml;ty=24

<?xml version="1.0" encoding="UTF-8"?>
<m2m:smd
    xmlns:m2m="http://www.onem2m.org/xml/protocols"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    rn="SD1">
    <dcrp>application/rdf+xml:1</dcrp>
    <dsp>
PD94bWwgdmVyc2lvbj0iMS4wIj8+DQo8cmRmOlJERiB4bWxucz0iaHR0cDovL3d3dy5vbmVtMm0ub3J
nL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxlIyINCiAgICAgeG1sOmJhc2U9Imh0dH
A6Ly93d3cub25lbTJtLm9yZy9vbnRvbG9neS9ob3VzZXNfdGVtcGVyYXR1cmVfZXhhbXBsZSINCiAgI
CAgeG1sbnM6dGVtcGVyYXR1cmVfZXhhbXBsZT0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5
L3RlbXBlcmF0dXJlX2V4YW1wbGUjIg0KICAgICB4bWxuczpmZGY9Imh0dHA6Ly93d3cudzMub3JnLzE
5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiDQogICAgIHhtbG5zOm93bD0iaHR0cDovL3d3dy53My5vcm
cvMjAwMi8wNy9vd2wjIg0KICAgICB4bWxuczp4bWw9Imh0dHA6Ly93d3cudzMub3JnL1hNTC8xOTk4L
25hbWVzcGFjZSINCiAgICAgeG1sbnM6eHNkPSJodHRwOi8vd3d3LnczLm9yZy8yMDAxL1hNTFNjaGVt
YSMiDQogICAgIHhtbG5zOnJkZnM9Imh0dHA6Ly93d3cudzMub3JnLzIwMDAvMDEvcmRmLXNjaGVtYSM
iPg0KDQogICAgPG93bDpOYW1lZEluZGl2aWR1YWwgcmRmOmFib3V0PSJodHRwOi8vd3d3Lm9uZW0ybS
5vcmcvb250b2xvZ3kvaG91c2VzX3RlbXBlcmF0dXJlX2V4YW1wbGUjSG91c2UxIj4NCiAgICAgICAgP
HJkZjp0eXBlIHJkZjpyZXNvdXJjZT0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L3RlbXBl
cmF0dXJlX2V4YW1wbGUjSG91c2UiLz4NCiAgICAgICAgPHRlbXBlcmF0dXJlX2V4YW1wbGU6aGFzSW5
kb29yVGVtcGVyYXR1cmUgcmRmOnJlc291cmNlPSJodHRwOi8vd3d3Lm9uZW0ybS5vcmcvb250b2xvZ3
kvaG91c2VzX3RlbXBlcmF0dXJlX2V4YW1wbGUjSW5kb29yVGVtcFByb3BlcnR5MSIvPg0KICAgIDwvb
3dsOk5hbWVkSW5kaXZpZHVhbD4NCg0KICAgIDxvd2w6TmFtZWRJbmRpdmlkdWFsIHJkZjphYm91dD0i
aHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxlI0l
uZG9vclRlbXBQcm9wZXJ0eTEiPg0KICAgICAgICA8cmRmOnR5cGUgcmRmOnJlc291cmNlPSJodHRwOi
8vd3d3Lm9uZW0ybS5vcmcvb250b2xvZ3kvdGVtcGVyYXR1cmVfZXhhbXBsZSNUZW1wZXJhdHVyZVByb
3BlcnR5Ii8+DQogICAgICAgIDx0eXBlIHJkZjpyZXNvdXJjZT0iaHR0cDovL3d3dy53My5vcmcvMjAw
MS8wMy9zb2FwLWVuYy1saXRlcmFsIi8+DQogICAgPC93bDpOYW1lZEluZGl2aWR1YWw+DQogICAgPG
dsOmhhc1VuaXQ+RmFocmVuaGVpdDwvdGVtcGVyYXR1cmVfZXhhbXBsZTpoYXNVbml0Pg0KICAgIC
AgICA8dGVtcGVyYXR1cmVfZXhhbXBsZTp2YWx1ZUlzU3RvcmVkSW4+aHR0cDovL2luLnByb3ZpZGVyL
mNvbTo3NTc5L3NlcnZlci90ZW1wc2Vuc29yYWUxL3RlbXBlcmF0dXJlL2xhdGVzdDwvdGVtcGVyYXR1
cmVfZXhhbXBsZTp2YWx1ZUlzU3RvcmVkSW4+DQogICAgPC9vd2w6TmFtZWRJbmRpdmlkdWFsPg0KDQo
gICAgPG93bDpOYW1lZEluZGl2aWR1YWwgcmRmOmFib3V0PSJodHRwOi8vd3d3Lm9uZW0ybS5vcmcvb2
50b2xvZ3kvaG91c2VzX3RlbXBlcmF0dXJlX2V4YW1wbGUjSW5kb29yVGVtcFNlbnNvcjEiPg0KICAgI
CAgICA8cmRmOnR5cGUgcmRmOnJlc291cmNlPSJodHRwOi8vd3d3Lm9uZW0ybS5vcmcvb250b2xvZ3kv
dGVtcGVyYXR1cmVfZXhhbXBsZSNUZW1wZXJhdHVyZVNlbnNvciIvPg0KICAgICAgICA8dGVtcGVyYXR
1cmVfZXhhbXBsZTpoYXNUZW1wZXJhdHVyZU1lYXN1cmluZ0Z1bmN0aW9uIHJkZjpyZXNvdXJjZT0iaH
R0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxlI1Rlb
XBGdW5jdGlvbjEiLz4NCiAgICA8L293bDpOYW1lZEluZGl2aWR1YWw+DQoNCiAgICA8b3dsOk5hbWVk
SW5kaXZpZHVhbCByZGY6YWJvdXQ9Imh0dHA6Ly93d3cub25lbTJtLm9yZy9vbnRvbG9neS9ob3VzZXN
fdGVtcGVyYXR1cmVfZXhhbXBsZSNUZW1wRnVuY3Rpb24xIj4NCiAgICAgICAgPHJkZjp0eXBlIHJkZj
pyZXNvdXJjZT0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L3RlbXBlcmF0dXJlX2V4YW1wb
GUjVGVtcGVyYXR1cmVNZWFzdXJpbmdGdW5jdGlvbiIvPg0KICAgICAgICA8dGVtcGVyYXR1cmVfZXhh
bXBsZTptZWFzdXJlc1RlbXBlcmF0dXJlIHJkZjpyZXNvdXJjZT0iaHR0cDovL3d3dy5vbmVtMm0ub3J
nL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxlI0luZG9vclRlbXBQcm9wZXJ0eTEiLz
4NCiAgICA8L293bDpOYW1lZEluZGl2aWR1YWw+ICAgDQo8L3NlYXJjaEREY+DQo=</dspt>
    <or>http://www.onem2m.org/ontology/temperature_example</or>
</m2m:smd>
```

HTTP Response:

```
201 Created
Content-Location: /server/sdt-982750012
X-M2M-RI: 12349
X-M2M-RSC: 2001
```

10.5.5 Semantic discovery of relevant sensor resources

For the semantic discovery, a semantic filter is specified. The semantic filter is formulated as a SPARQL query. For all child resources that themselves have semantic descriptor child resources, the SPARQL query is executed on the content of the descriptor attribute of each of the semantic descriptor resources. If the SPARQL query returns a valid result on one of the semantic descriptor child resources, the parent resource will be included in the result list. There are also options for linking additional semantic content to be considered together with the semantic content of the descriptor for evaluating the SPARQL query - either through the *relatedSemantics* attribute of the semantic descriptor resource, or through the *onem2m:resourceDescriptorLink* annotation property in the RDF description in the descriptor attribute itself - but this is beyond the scope of this developer guide. Details for this can be found in TS-0001 [i.6] and [i.7].

In the following examples, a number of different semantic filters formulated in SPARQL are provided together with the results that would be returned, thus determining which (parent) resources would be included in the result of the discovery. A complete request is also shown. The SPARQL request has to be URI encoded as it is used as the value of the *smf* (semantic filter) parameter.

Query 1:

Natural language query: "Give me all the sensor related resources that measure the temperature in the unit Celsius."

```
PREFIX temp: <http://www.onem2m.org/ontology/temperature_example#>
SELECT ?sensor
 WHERE {
         ?sensor temp:hasTemperatureMeasuringFunction ?tempFunction .
                      ?tempFunction temp:measuresTemperature ?property .
         ?property temp:hasUnit
                      "Celsius"^^<http://www.w3.org/2001/XMLSchema#string>
}
```

?sensor	does not match	temp:IndoorTempSensor1	→ tempsensorae1 is not included in the result
?sensor	matches	temp:OutdoorTempSensor	→ tempsensorae2 is included in the result
?sensor	matches	temp:IndoorTempSensor2	→ tempsensorae3 is included in the result
?sensor	matches	temp:FridgeTempSensor	→ tempsensorae4 is included in the result

Query 2:

Natural language query: "Give me all the sensor related resources that measure the indoor temperature of a house."

```
PREFIX temp: <http://www.onem2m.org/ontology/temperature_example#>
SELECT ?sensor
 WHERE { ?thing temp:hasIndoorTemperature ?property .
         ?tempFunction temp:measuresTemperature ?property .
         ?sensor temp:hasTemperatureMeasuringFunction ?tempFunction
}
```

?sensor	matches	temp:IndoorTempSensor1	→ tempsensorae1 is included in the result
?sensor	does not match	temp:OutdoorTempSensor	→ tempsensorae2 is not included in the result
?sensor	matches	temp:IndoorTempSensor2	→ tempsensorae3 is included in the result
?sensor	does not match	temp:FridgeTempSensor	→ tempsensorae4 is not included in the result

Query 3:

Natural language query: "Give me all the sensor related resources that measure the indoor temperature of a house and provide the measurement in the unit Fahrenheit."

```
PREFIX temp: <http://www.onem2m.org/ontology/temperature_example#>
SELECT ?sensor
 WHERE { ?thing temp:hasIndoorTemperature ?property .
         ?property temp:hasUnit
                      "Fahrenheit"^^<http://www.w3.org/2001/XMLSchema#string> .
         ?tempFunction temp:measuresTemperature ?property .
         ?sensor temp:hasTemperatureMeasuringFunction ?tempFunction
}
```

?sensor	matches	temp:IndoorTempSensor1	→	tempsensorae1 is included in the result
?sensor	does not match	temp:OutdoorTempSensor	→	tempsensorae2 is not included in the result
?sensor	does not match	temp:IndoorTempSensor2	→	tempsensorae3 is not included in the result
?sensor	does not match	temp:FridgeTempSensor	→	tempsensorae4 is not included in the result

The following example shows how the semantic discovery application (ADN-AE6) on the user device can perform the complete HTTP request for Query 2, where the semantic filter in SPARQL is URI-encoded.

```
HTTP Request:

GET /server?fu=1&amp;smf=
PREFIX%20temp%3A%20%3Chttp%3A%2F%2Fwww.onem2m.org%2Fontology%2Ftemperature_exam
ple%23%3E%20SELECT%20%3Fsensor%20WHERE%20%7B%20%3Fthing%20temp%3AhasIndoorTempe
rature%20%3Fproperty%20.%20%3FtempFunction%20temp%3AmeasuresTemperature%20%3Fpr
operty%20.%20%3Fsensor%20temp%3AhasTemperatureMeasuringFunction%20%3FtempFuncti
on%7D HTTP/1.1
Host: in.provider.com:7579
Accept: application/xml
X-M2M-RI: 12355
X-M2M-Origin: SOrigin

HTTP Response:

200 OK
Accept: application/xml
X-M2M-RI: 12355
X-M2M-RSC: 2000

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<m2m:uril xmlns:m2m="http://www.onem2m.org/xml/protocols"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 server/tempsensorae1
 server/tempsensorae3
</m2m:uril>
```

As expected tempsensorae1 (ADN-AE1) and tempsensorae3 (ADN-AE3) are returned.

10.5.6 Retrieval of semantic descriptor and sensor information

As the final step, the semantic discovery application (ADN-AE6) on the user device can now retrieve the semantic descriptor itself and find out how to access the latest sensor measurement (under the assumption that the convention of using the name SD1 for the semantic descriptor was used, otherwise it needs to get the information about the semantic descriptor child resource(s) first).

```
HTTP Request:

GET /server/tempsensorae1/SD1 HTTP/1.1
Host: in.provider.com:7579
Accept: application/xml
X-M2M-RI: 12645
X-M2M-Origin: SOrigin

HTTP Result:
200 OK
X-M2M-RI: 12645
X-M2M-RSC: 2000


<?xml version="1.0" encoding="UTF-8"?>
<m2m:smd
    xmlns:m2m="http://www.onem2m.org/xml/protocols"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    rn="SD1">
<pi>cse00001</pi>
<ty>24</ty>
```

```
<ct>20170519T064158</ct>
<ri>sd-342342434</ri>
<lt>20170519T064158</lt>
<et>20190518T063022</et>
<aei>SOrigin</aei>
<dcrp>application/rdf+xml:1</dcrp>
<dsp>PD94bWwgdmVyc2lvbj0iMS4wIj8+DQo8cmRmOlJERiB4bWxucz0iaHR0cDovL3d3dy5vbmVtMm
0ub3JnL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxlIyINCiAgICAgICAgeG1sOmJhc2U9I
mh0dHA6Ly93d3cub25lbTJtLm9yZy9vbnRvbG9neS9ob3VzZXNfdGVtcGVyYXR1cmVfZXhhbXBsZSIN
CiAgICAgICAgeG1sbnM6dGVtcGVyYXR1cmVfZXhhbXBsZT0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9
sb2d5L3RlbXBlcmF0dXJlX2V4YW1wbGUjIg0KICAgICAgICB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3
JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiDQogICAgIHhtbG5zOm93bD0iaHR0cDovL3d3dy53M
y5vcmcvMjAwMi8wNy9vd2wjIg0KICAgICAgICB4bWxuczp4bWw9Imh0dHA6Ly93d3cudzMub3JnL1hNTC8x
OTk4L25hbWVzcGFjZSINCiAgICAgICAgeG1sbnM6eHNkPSJodHRwOi8vd3d3LnczLm9yZy8yMDAxL1hNTFN
jaGVtYSMiDQogICAgIHhtbG5zOnJkZnM9Imh0dHA6Ly93d3cudzMub3JnLzIwMDAvMDEvcmRmLXNjaG
VtYSMiPg0KDQogICAgPG93bDpPYW1lZEluZGl2aWR1YWwgcmRmOmFib3V0PSJodHRwOi8vd3d3Lm9uZW0ybS5vcmcvb250b
2xvZ3kvaG91c2VzX3RlbXBlcmF0dXJlX2V4YW1wbGUjSW5kb29yVGVtcFByb3BlcnR5MSIvPg0KICAg
IDwvb3dsOk5hbWVkSW5kaXZpZHVhbD4NCg0KICAgIDxvd2w6TmFtZWRJbmRpdmlkdWFsIHJkZjphYm9
1dD0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcG
xlI0luZG9vclRlbXBQcm9wZXJ0eTEiPg0KICAgICAgICA8cmRmOnR5cGUgcmRmOnJlc291cmNlPSJod
HRwOi8vd3d3Lm9uZW0ybS5vcmcvb250b2xvZ3kvdGVtcGVyYXR1cmVfZXhhbXBsZSNUZW1wZXJhdHVy
ZVByb3BlcnR5Ii8+DQogICAgICAgIDx0ZW1wZXJhdHVyZV9leGFtcGxlOmhhc0RhdGF0eXBlPnhzZDp
pbnQ8L3RlbXBlcmF0dXJlX2V4YW1wbGU6aGFzRGF0YXR5cGU+DQogICAgICAgIDx0ZW1wZXJhdHVyZV
9leGFtcGxlOmhhc1VuaXQ+RmFocmVuaGVpdDwvdGVtcGVyYXR1cmVfZXhhbXBsZTpoYXNVbml0Pg0KI
CAgICAgICA8dGVtcGVyYXR1cmVfZXhhbXBsZTp2YWx1ZUlzU3RvcmVkSW4+aHR0cDovL2luLnByb3Zp
ZGVyLmNvbTo3NTc5L3NlcnZlci90ZW1wc2Vuc29yYWUxL3RlbXBlcmF0dXJlL2xhdGVzdDwvdGVtcGV
yYXR1cmVfZXhhbXBsZTp2YWx1ZUlzU3RvcmVkSW4+DQogICAgPC9vd2w6TmFtZWRJbmRpdmlkdWFsPg
0KDQogICAgPG93bDpOYW1lZEluZGl2aWR1YWwgcmRmOmFib3V0PSJodHRwOi8vd3d3Lm9uZW0ybS5vc
mcvb250b2xvZ3kvaG91c2VzX3RlbXBlcmF0dXJlX2V4YW1wbGUjSW5kb29yVGVtcFNlbnNvcjEiPg0K
ICAgICAgICA8cmRmOnR5cGUgcmRmOnJlc291cmNlPSJodHRwOi8vd3d3Lm9uZW0ybS5vcmcvb250b2x
vZ3kvdGVtcGVyYXR1cmVfZXhhbXBsZSNUZW1wZXJhdHVyZVNlbnNvciIvPg0KICAgICAgICA8dGVtcG
VyYXR1cmVfZXhhbXBsZTpoYXNUZW1wZXJhdHVyZU1lYXN1cmluZ0Z1bmN0aW9uIHJkZjpyZXNvdXJjZ
T0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxl
I1RlbXBGdW5jdGlvbjEiLz4NCiAgICA8L293bDpOYW1lZEluZGl2aWR1YWw+DQoNCiAgICA8b3bdsOk5
hbWVkSW5kaXZpZHVhbCByZGY6YWJvdXQ9Imh0dHA6Ly93d3cub25lbTJtLm9yZy9vbnRvbG9neS9ob3
VzZXNfdGVtcGVyYXR1cmVfZXhhbXBsZSNUZW1wRnVuY3Rpb24xIj4NCiAgICAgICAgPHJkZjp0eXBlI
HJkZjpyZXNvdXJjZT0iaHR0cDovL3d3dy5vbmVtMm0ub3JnL29udG9sb2d5L3RlbXBlcmF0dXJlX2V4
YW1wbGUjVGVtcGVyYXR1cmVNZWFzdXJpbmdGdW5jdGlvbiIvPg0KICAgICAgICA8dGVtcGVyYXR1cmV
fZXhhbXBsZTptZWFzdXJlc1RlbXBlcmF0dXJlIHJkZjpyZXNvdXJjZT0iaHR0cDovL3d3dy5vbmVtMm
0ub3JnL29udG9sb2d5L2hvdXNlc190ZW1wZXJhdHVyZV9leGFtcGxlI0luZG9vclRlbXBQcm9wZXJ0e
TEiLz4NCiAgICA8L293bDpOYW1lZEluZGl2aWR1YWw+ICAgDQo8L3dsp>SREY+DQo=</dsp>
    <or>http://www.onem2m.org/ontology/temperature_example</or>
</m2m:smd>
```

Decoding the base64 encoded descriptor attribute and extracting the *valueIsStoredIn* data property of *TempProperty1*, yields the URI for accessing the latest content instance storing with the most recently measured temperature value: http://in.provider.com:7579/server/tempsensorae1/temperature/latest

```
HTTP Request:

GET /server/tempsensorae1/temperature/latest HTTP/1.1
Host: in.provider.com:7579
Accept: application/xml
X-M2M-RI: 16345
X-M2M-Origin: SOrigin

HTTP Result:
200 OK
X-M2M-RI: 16345
X-M2M-RSC: 2000

<?xml version="1.0" encoding="UTF-8"?>
<m2m:cin
   xmlns:m2m=http://www.onem2m.org/xml/protocols
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="cin-342342314">
```

```
  <ty>4</ty>
  <ri>ci-234523664</ri>
  <pi>cse00001</pi>
  <ct>20170519T064023</ct>
  <lt>20170711T064023</lt>
  <et>20190518T063022</et>
  <st>3</st>
  <cnf>text/plain:0</cnf>
  <cs>2</cs>
  <con>75</con>
</m2m:cin>
```

So the content 75 is retrieved by the semantic discovery application - which it knows to be the indoor temperature of House#1 in Fahrenheit according to the semantic descriptor.

Annex A:
Mapping to oneM2M Base Ontology

Figure A-1 shows the mapping of the specific concepts of the simple ontology for the semantic annotation and discovery use case to the more general concepts of the oneM2M Base Ontology [i.3]. Thus the semantic integration with the concepts identified for oneM2M itself is achieved.

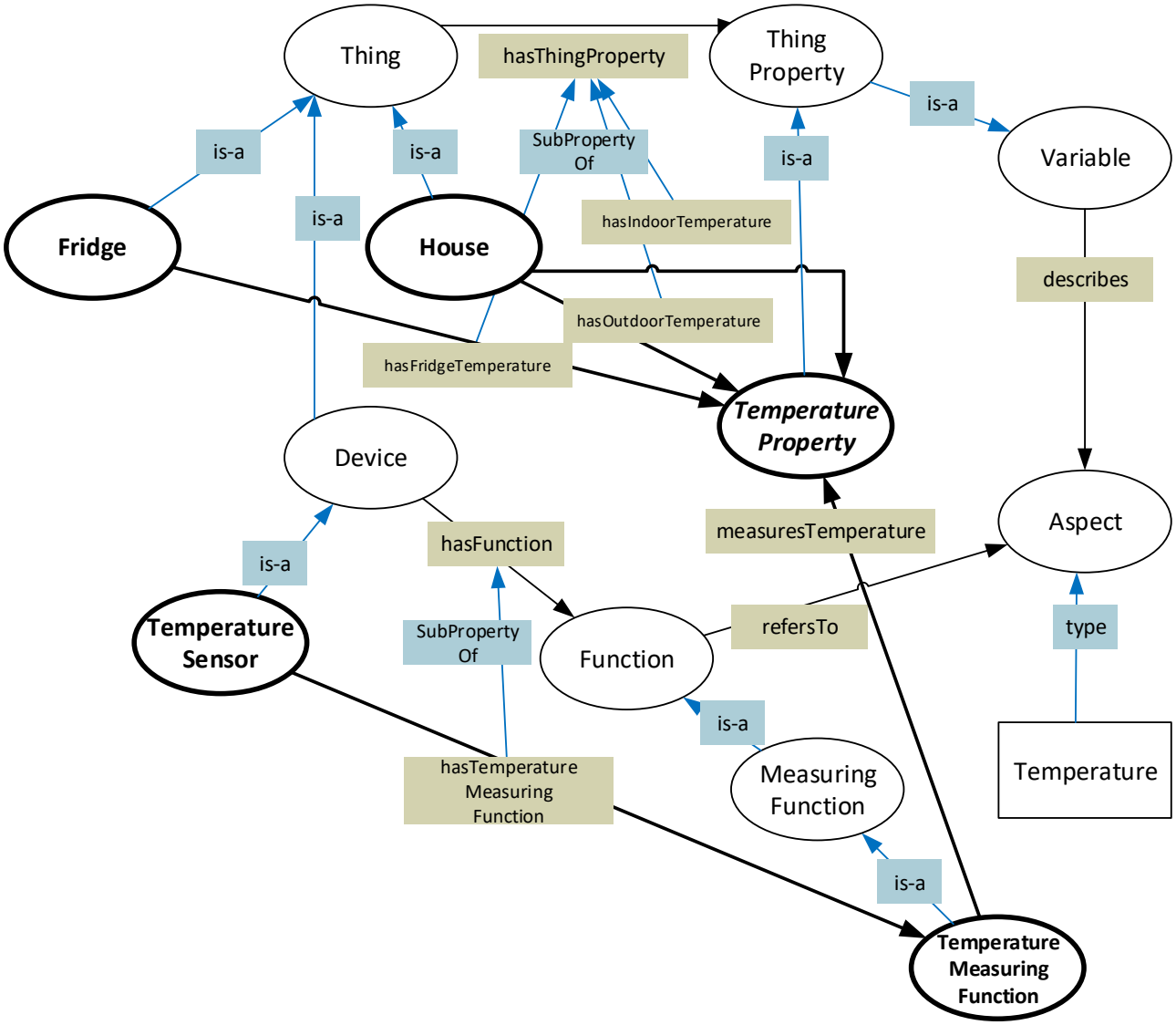

Figure A-1: Mapping of the introduced concepts to the oneM2M Base Ontology

When comparing to the concepts available in the complete oneM2M Base Ontology, it can be seen that the focus of the modelling here has been on the semantic aspects. It is assumed that the functional aspects are implemented by oneM2M AEs using the Mca interface. In the case of using non-oneM2M technologies and applications for the sensors, an interworking proxy would be needed to translate between oneM2M and the native technology. Using Generic Interworking as described in [i.4], with a complete modelling of the functionality according to the oneM2M Base Ontology (or an ontology that has been mapped to the oneM2M Base Ontology in a similar way as shown in figure A-1), would allow the automatic creation of a oneM2M resource structure for the case and enable tool-support for the partially automated creation of the required interworking proxy.

History

Publication history		
V2.0.0	12-Mar-2018	Release 2A - Publication