TTC技術レポート
Technical Report

TR-M2M-0037v2.0.0

Smart farm example using MQTT binding

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-0037v2.0.0

Smart farm example using MQTT binding

＜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 0037 (Version 2.0.0) に準拠している。

[This Technical Report is transposed based on the Technical Report 0037 (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	oneM2M-TR-0037-V-2.0.0
Document Name:	Developers Guide - TR-0037-Smart Farm Example using MQTT Binding
Date:	2018-03-12
Abstract:	The developers guide provides a use case leveraging oneM2M resource primitives and oneM2M common functionalities using MQTT protocol binding. The use case is designed to implement remote monitoring and controlling of smart devices deployed in an agriculture farm.

Template Version:23 February 2015 (Dot 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

Notice of Disclaimer & Limitation of Liability

The information provided in this document is directed solely to professionals who have the appropriate degree of experience to understand and interpret its contents in accordance with generally accepted engineering or other professional standards and applicable regulations. No recommendation as to products or vendors is made or should be implied.

NO REPRESENTATION OR WARRANTY IS MADE THAT THE INFORMATION IS TECHNICALLY ACCURATE OR SUFFICIENT OR CONFORMS TO ANY STATUTE, GOVERNMENTAL RULE OR REGULATION, AND FURTHER, NO REPRESENTATION OR WARRANTY IS MADE OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR AGAINST INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS. NO oneM2M PARTNER TYPE 1 SHALL BE LIABLE, BEYOND THE AMOUNT OF ANY SUM RECEIVED IN PAYMENT BY THAT PARTNER FOR THIS DOCUMENT, WITH RESPECT TO ANY CLAIM, AND IN NO EVENT SHALL oneM2M BE LIABLE FOR LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES. oneM2M EXPRESSLY ADVISES ANY AND ALL USE OF OR RELIANCE UPON THIS INFORMATION PROVIDED IN THIS DOCUMENT IS AT THE RISK OF THE USER.

Contents

1 Scope

The present document provides a use case leveraging oneM2M common service functionalites and oneM2M resource primitives using MQTT protocol binding. The use case is designed to implement remote monitoring and actuating of smart devces deployed in an agriculture farm.

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.

The following referenced documents are necessary for the application of the present document.

Not applicable.

2.2 Informative references

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 (http://member.onem2m.org/Static_pages/Others/Rules_Pages/oneM2M-Drafting-Rules-V1_0.doc)

[i.2] oneM2M TS-0001: "Functional Architecture".

[i.3] oneM2M TS-0004: "Service Layer Core protocol Specification".

[i.4] oneM2M TS-0010: "MQTT Protocol Binding".

[i.5] oneM2M TS-0011: "Common Terminology".

[i.6] oneM2M TS-0003: "Security Solutions".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, terms and definitions following apply:

Cloud Server: A cloud service platform that implements (full or partial) oneM2M common service functionalities and resource primtives

IoT Application: A smart application that is developed under a specific operation system e.g. Andriod, iOS etc. and hosts an application entity wich implements certain oneM2M common service functionalities and resource primitives

Farm Gateway: A gateway that hosts a middle node which implements common service functionalities

NOTE: This may contain additional information.

3.2 Abbreviations

.For the purposes of the present document, abbreviations given in TS-0001[i.2] and the following apply:

ADN	Application Dedicated Node
ADN-AE	AE which resides in the Application Dedicated Node
AE	Application Entity
CSE	Common Services Entity
CSE-ID	Common Service Entity Identifier
IN	Infrastructure Node
IN-AE	Application Entity that is registered with the CSE in the Infrastructure Node
IN-CSE	CSE which resides in the Infrastructure Node
JSON	JavaScript Object Notation
M2M	Machine to Machine
Mca	Reference Point for M2M Communication with AE
Mcc	Reference Point for M2M Communication with CSE
MN	Middle Node
MN-AE	Application Entity that is registered with the CSE in Middle Node
MN-CSE	CSE which resides in the Middle Node
SP	Service Provider
URI	Uniform Resource Identifier

4 Conventions

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

5 Design of the Use Case

5.1 Introduction

This user guide demonstrates how to implement a smart farm control use case that deploys different sensors to collect environment measurement data, and actuators to trigger some kind of IoT services by remotely human intervention using IoT applications. All devices involved in the smart farm control scenario leverage oneM2M common services functionalities.

The smart farm control use case is designed to implement functions as following:

- Remote environment measurement monitoring and remote actuators controlling using IoT applications by applying MQTT subscription and notification mechanism.
- Registration of sensors, actuators, and farm gateway application into the farm gateway as well as the farm gateway and IoT applications into a cloud server.
- Discovery of oneM2M resources representing sensors and actuators in farm gateway using IoT applications.

An overview of the smart farm control use case is depicted in **Error! Reference source not found.**, where compound sensors with sensing data of temperature, humidity and CO2, illumination sensors, soil moisture and Photosynthetic Photon Flux Density (PPFD) sensors as well as actuators used to trigger machines deployed in the farm such as fans, air conditioner, sprinkler, roof cover controller and irrigation and nutrient management system etc. Those deployed sensors and actuators are connected to a farm gateway which is capable to communicate with a oneM2M cloud server. In addition, IoT applications that interface directly with the cloud server are responsible for monitoring crops growing environment and controlling farm machines remotely. A farm gateway application is responsible for communication with the farm gateway. The subscription and notification mechnisms are applied to implement the remote farm monitoring and controlling services.

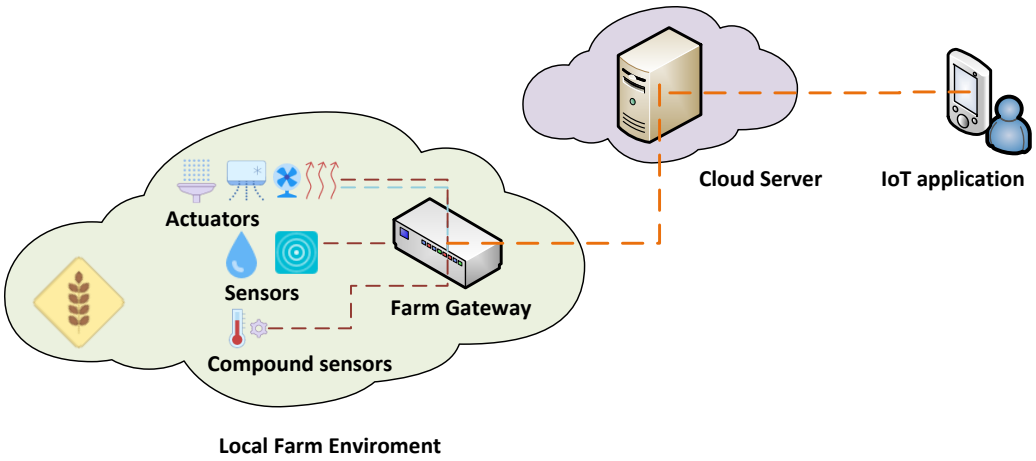

Figure 5.1-1 Overview of smart farm control use case

5.2 Modelling Components

IoT devices involved in the use case are modelled to map to entity types defined in oneM2M TS-0001. The modelling of those IoT devices are as following:

- Sensors such as compound sensors, illumination sensors, soil moisture and PPFD sensors each host an Application Dedicated Nodes with embedded application entity (ADN-AE) for collecting farm environment measurements and uploading the data into a farm gateway.
- Actuators such as fans, air conditioner, sprinkler, and roof cover controller each host an Application Dedicated Nodes with embedded application entity (ADN-AE).
- The farm gateway hosts a Middle Nodes with embedded Common Service Entity (MN-CSE) which provides both interface with sensors/actuators and interface with the cloud server.
- IoT cloud server hosts an oneM2M Infrastructure Node Common Service Entity (IN-CSE) which implements all the functionalities defined in oneM2M specifications such as registration, data management, subscription and notification etc.
- IoT applications each host an Application Dedicated Nodes with embedded application entity (ADN-AE) which interface directly with the cloud server to request interested resources exposed by the farm gateway.
- The gateway application hosts an application entity (MN-AE) that resides on a Middle Node with embedded Common Service Entity (MN-CSE), which interfaces with the farm gateway (MN-CSE).

6 Functional Architecture of the Use Case

6.1 Introduction

Functional architecture of smart farm control use case is depicted in Figure 6.1-1, where the reference points Mca and Mcc are used to transport oneM2M service primitives between ADN-AE and MN-CSE, and between MN-CSE and IN-CSE, respectively. In other words, both Mca and Mcc reference points have to be implemented as a interface using either HTTP, MQTT or CoAP protocol binding between compound sensors/actuators and farm gateway as well as

between farm gateway and the cloud server, respectively; reference point Mca also needs to be implemented between IoT applications and cloud server.

Communications between any entity i.e. ADN-AE and MN-CSE, MN-CSE and IN-CSE, and MN-CSE and IN-CSE are achieved through reference points Mca and Mcc by transport of oneM2M resource and attribute primitives such as <*AE*>, <*AccessControlPolicy*>, <*Subscription*>, <*flexContainer*> and <*ContentInstance*> etc in a protocol-specific manner where data are represented in one of serialization type JSON, XML or CBOR. The creation and management of those oneM2M resource primitives will be introduced in the following sections. A resource tree is generated as a result of successful operations to indicate parent-child relationships between any two entity.

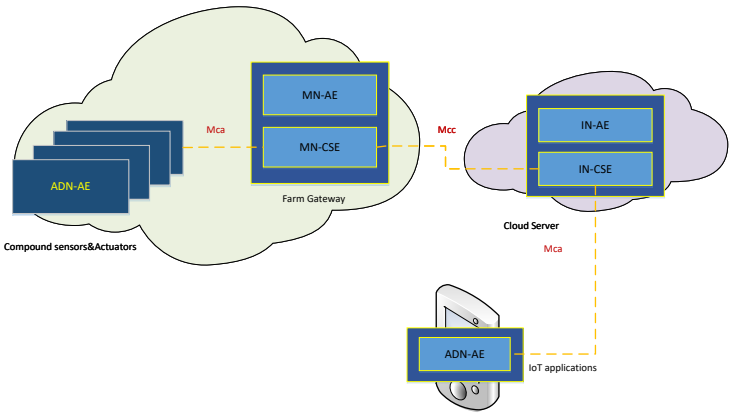

Figure 6.1-1 Functional Architecture of the smart farm control use case

7 Function Call Flows

7.0 Introduction

The use case addressing the smart farm control involves a group of components where functions are called to exchange data between any two designated components. The cloud server provides a set of services for entity registration, data management and authenticated access control etc. Function calls are normally initialized by the AE entity and reacted by CSE (including MN-CSE and IN-CSE) and result in resource created, deleted, and updated, and/or command execution etc. The following clauses introduce function calls in different procedures that are required to implement services in this use case.

7.1 Entity Registration

Different sensors and actuators are involved in the smart farm control use case and each hosts an ADN-AE to interface with the farm gateway which host a MN-CSE. In addition, an ADN-AE is also hosted on each IoT application to interface with the IN-CSE.

Sensors, actuators, IoT applications, farm gateway application and farm gateway are required to register with relevant registrar CSE in order to implement oneM2M services such as data management etc.

We assume all AE entities have never been registered with any CSE in this use case and the initial registration (that differentitates with re-registration) procedures are defined as following:

1. The farm gateway (MN-CSE) performs registration with the cloud server (IN-CSE) which results in the creation of a remoteCSE resource representing the MN-CSE under the IN-CSE and meanwhile creation of a remoteCSE representing the IN-CSE under the MN-CSE.

2. Each sensor and actuator (ADN-AE) performs initial registration with the farm gateway (MN-CSE) which results in the creation of AE resource under the MN-CSE.

3. Each IoT application (ADN-AE) performs initial registration with the cloud server (IN-CSE) which results in the creation of AE resource under the IN-CSE. The ADN-AE will be used for monitoring the resources (e.g. the working status of sensors, actuators, measurement data generated by sensors etc.) that are stored in the farm gateway.

4. The farm gateway application (MN-AE) performs initial registration with the farm gateway (MN-CSE) which results in the creation of AE resource under the MN-CSE.

Note that there is no strict rules on the sequence of those registration operations, i.e. whether the sensors/actuators or the farm gateway registration with farm gateway have to be performed before the farm gateway registration with the cloud server. It's up to developers to define the sequence for those registration procedures according to different application development.

The function call flows are depicted in Figure 7.1-1.

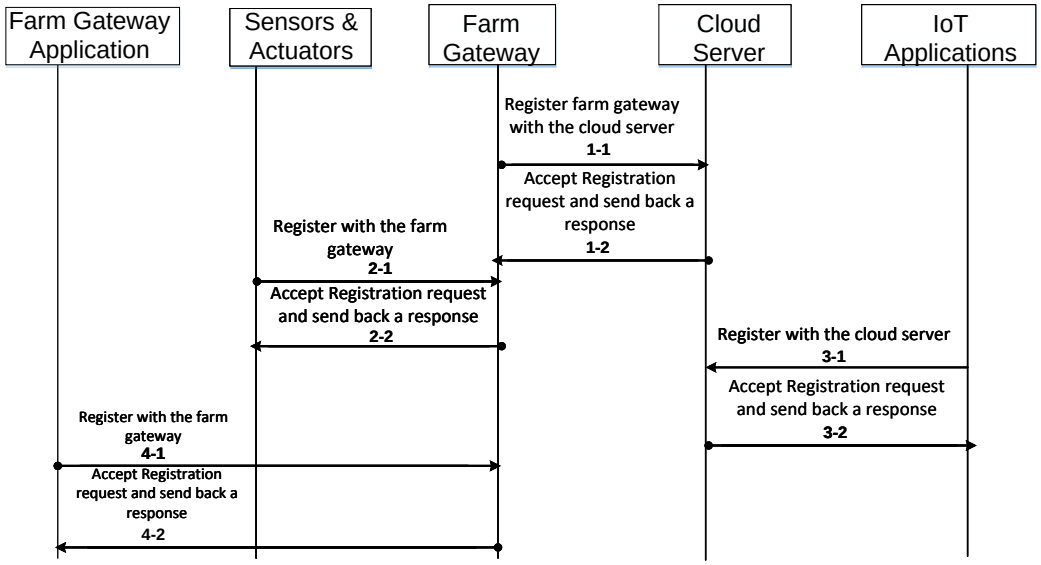

Figure 7.1-1 Entity Registration Flows

The entity registration results in the generation of hierarchical structures for IN-CSE and MN-CSE as shown in (a) and (b) in Figure 7.1-2, respectively.

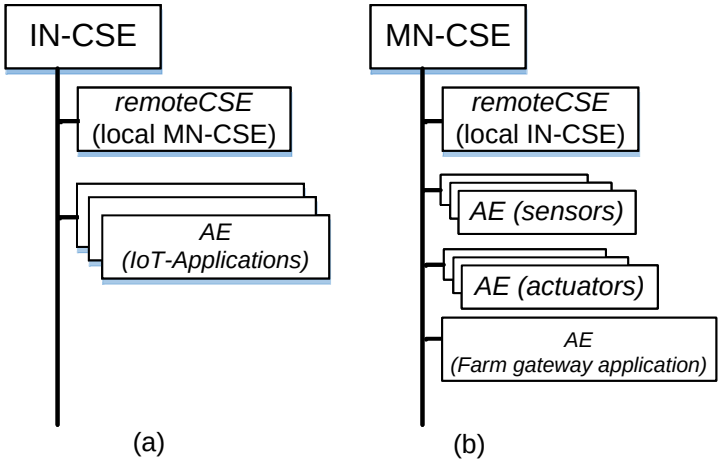

Figure 7.1-2 oneM2M Resource Tree of farm gateway and cloud server

7.2 Initial Resource Generation

A group of resources are required as preconditions to implement services such as data management, subscription and notification etc. Procedures for creating container and subscription resources are stated as following:

1. Each sensor (ADN-AE) creates a container resource under its AE resource located in the farm gateway (MN-CSE). The created container resource is used as a repository of environment measurement data and subscription resources are also created under these container resources for monitoring any update to this data repository.

2. Each actuator (ADN-AE) creates a container resource unders its AE resource located in the farm gateway (MN-CSE). The created container resource is used as a repository of execution commands for actuators in this use case. Each subscription resource (at least one) is created under each created container resource to trigger actuators using subscription&notificiation mechanism whose working principle is present at section 7.3.

3. Each IoT application (ADN-AE) subscribes to a particular container that is created as a repository of environment measurement data for a sensor to monitor the farm environment.

4. The farm gateway application (MN-AE) subscribes to a particular container resource that is created for controlling purpose in an AE hosted on actuators to receive notifications which contain triggering commands for a particular actuator.

The function call flows are depicted in Figure 7.2-1.

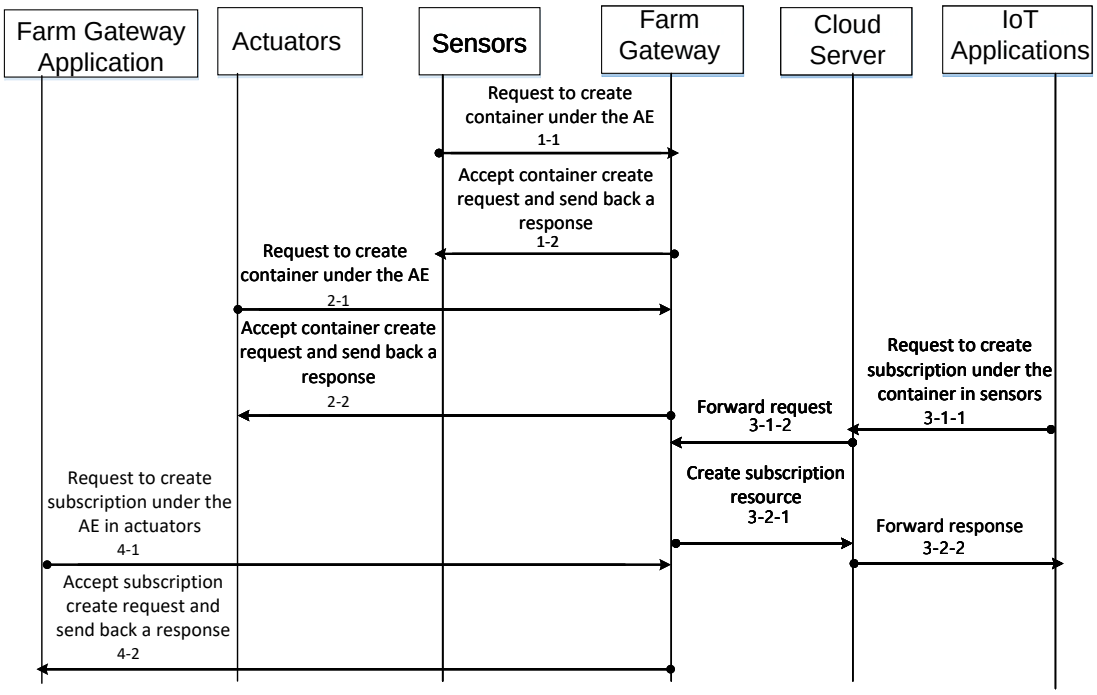

Figure 7.2-1 Initial Resource Creation Flows

The initial resource creation procedure results generation of hierarchical structure as shown in Figure 7.2-2 for an AE that is hosted on a sensor, an actuator and an farm gateway application, respectively.

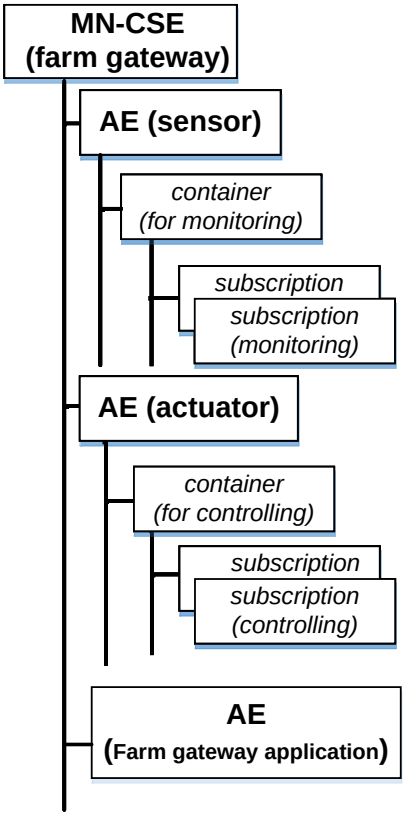

Figure 7.2-2 oneM2M Resource tree of sensor, actuator and farm gateway application registered into farm gateway

7.3 Actuation Usage using Subscription and Notification Mechanism

Actuators deployed in this use case can be actuated using differernt functionalities defined in oneM2M specification, one option is to apply subscription and notification mechanisms, and another one is using periodic polling method. In this use case, the former solution using subscription and notifications enabled by the publishing and subscribing feature of MQTT protocol is applied to implement the actuation services.

After registration with the farm gateway, an AE resource is created under the farm gateway (MN-CSE) and container resources are created under the AE resource as a repository of execution commands designed for a particular actuator. To apply subscription and notification mechanism, the farm gateway application (MN-AE) has to subscribe to the container resource (`container_control`) and configure the notification target URI to the public access address of the farm gateway application so that the farm gateway application will receive the execution commands wrapped in notification message.

In addition, the subscription resource (`sub_control`) need to be configured with notification criteria as child resource creation, and when the IoT applications (ADN-AE) acting as a controller creates a new contentInstance containing the execution command e.g. “on” or “off” in a digital format under the subscribed-to container resource, the farm gateway application will receive the notification which contains the execution commands. Then the execution commands can be abstracted by the farm gateway to trigger a particular actuator that hosts the subscription resource (`sub_control`).

Figure 7.3-1 shows the hierarchical resource structure of actuators with container and subscription created.

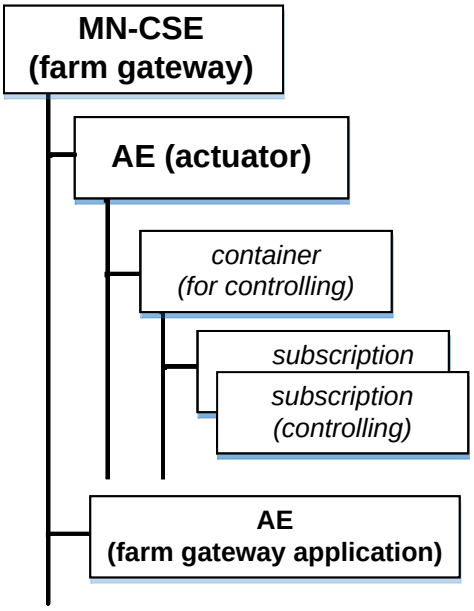

Figure 7.3-1 oneM2M Resource Tree of actuators with container and subscription resource created and the farm gateway application

7.4 Monitoring Usage using Subscription and Notification Mechanism

In the farm control use case, the measured quantity from sensors deployed in the farm will be used as a stimulus to trigger certain notification activities. The subscription and notification mechanism is applied to implement the monitoring of the environment data.

The measured quantities are normally stored in a container resource in a node which hosts an AE (ADN-AE). Subscription to container resources in AE hosted on sensors to receive notifications when certain notification events happen is commonly used to implement notifications in IoT applications e.g. whenever a new environment measurement is collected.

As depicted in Figure 7.4-1, AE hosts on any sensor creates one container for storing contentInstances containg the environment measurement data. Subscription resources are also created under these containers with different notification preference parameters such as notification event type, notification content type etc. The subscription resource creation request is initialized by the IoT application (ADN-AE) and the notification target is also pointed to the public access address of this IoT application so that the notification message can be properly send to him. Whenever there is a new contentInstance wrapped with environment measurement data created in the container created for storing the environment data, the subscriber i.e. the IoT application will receive a notification message containing the content e.g. the new environment measurement value. In practical development, developers could configure the notification message (i.e. when to receive and what content will be included in the notification message) when they initialize to create a subscription resource. Also subscription resources could be created by same IoT application or different IoT applications for different monitoring purposes.

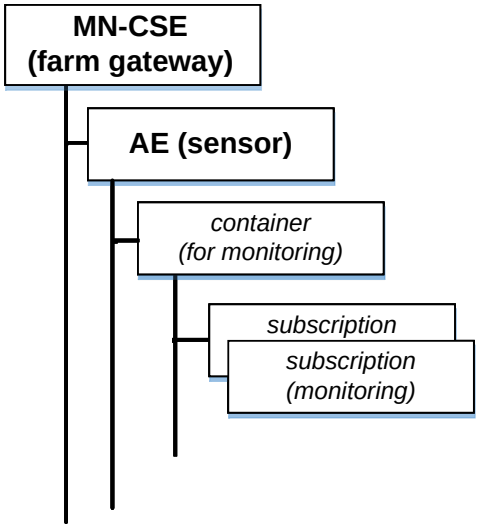

Figure 7.4-2 oneM2M Resource Tree of sensors registered into farm gateway

8 Implementation of Function Calls

8.1 Introduction

This section introduces standard APIs to implement function calls specified in section 7 using oneM2M MQTT binding and JSON serialization. The API is designed for oneM2M reference point Mca and Mcc.

8.2 Assumptions

It is assumed that the following conditions are met in the current use case:

①All devices and IoT applications deployed in the current use case are server capable, and independently addressable with host names resolved by DNS services.

②Host port number 8080 is reserved for oneM2M services.

③It is assume that Security Association Establishment is done with any two entities, but details for Security Association Establishment is out of the scope of this user guide.

④Any authentication and authorization check through access control policy will not be addressed in current document.

⑤The oneM2M request primitives are serialized into JSON serialization during communication btween any AE/CSE and a CSE via a MQTT protocol binding.

⑥The MN-CSE and IN-CSE being addressed in the current use case is deployed under a same oneM2M Service Provider domain.

⑦All application entities being modelled in this current use case will be proceeded as a fresh registration with corresponding CSE. The AE-ID-Stem then is set to character letter '`S`'.

⑧All resources being generated by oneM2M CRUD operations are addressable with their hierarchical resource identifier.

⑨All oneM2M requests initialized for accessing resources located in a CSE are proceeded as *Blocking Request* by the CSE receiver.

⑩ The MQTT server is configured to be able to form a Credential-ID, verify AE-ID or CSE-ID of the originators through successful Security Association Establishment with the orginator's MQTT client. It is assumed that Credential-ID is represented as `{CREDENTIAL_TYPE-VALUE}` in this user guide. Details of forming the real Credential-ID is out of scope of this user guide.

8.3 Entity Addressing and Resource Structuring

The entities AE and CSE can be addressed with host address that is either a IP address or a FQDN address that is resolved to IP addresses by DNS network services according to addressing rules specified in oneM2M standards.

Any oneM2M AE entities addressed in current use case is identified by a SP-Relative AE-ID, which is represented as a "`/{AE-ID-Stem}`", where the AE-ID-Stem starts with a character letter 'S'.

Any oneM2M resources resulted by a oneM2M CREATE and UPDATE operation is identified by a SP-Relative resource identifier in a non-structured manner, which is represented as "`{SP-relative-CSE-ID}/{Unstructured-CSE-relative Resource ID}`".

Note: oneM2M architecture speicifies also structured resource identifier format as well as AE-ID format in Section 7.2 of TS-0001 [i.2]. For implementation details of other formats, please refer to corresponding user guide.

Resource names listed as following are used to assign to AEs and child resources of AE in current use case. Here we provides an example for naming AE and child resources of AE, and developers can name their AE and child resources according to their naming preference and naming style.

- IN-CSE:
  - CSEBase Name:

    `server`
  - CSE-ID (SP-relative-CSE-ID):

    `/CSE1534123`
  - Resource ID:

    `incse8563723426`
  - Resource Name being created for IoT applications hosted in a smart device:

    `app_ae01`, `app_ae02`, `app_ae03`, …
- MN-CSE:
  - CSEBase Name:

    `farm_gateway`
  - CSE-ID:

    `/CSE3409165`
  - Resource Name being created for farm gateway application:

    `gateway_ae`
- ADN-AE:
  - Resource Name for <AE> resources being created for sensors:

    `sensor_ae01`, `sensor_ae02`, `sensor_ae03`, …
    - Resource Name for <container> resources being created for mornitoring purpose:

cont_monitor01, cont_monitor02, cont_monitor03, …

- Resource Name for <subscription> resources being created for subscription of environment monitoring data:

  sub_monitor01, sub_monitor02, sub_monitor03, …

- Resource Name for <AE> resources hosted actuators:

  actuator_ae01, actuator_ae02, actuator_ae03, …

  - Resource Name for <container> resources being created for controlling purpose:

    cont_ctrl01, cont_ctrl02, cont_ctrl03, …

  - Resource Name for <subscription> resources being created for subscription of device controlling signal:

    sub_ctrl01, sub_ctrl02, sub_ctrl03, …

8.4 Physical Quantity Modelling

The current use case addresses a suite of sensors and actuators. Sensors, such as compound sensors, illumination sensors, soil moisture and PPFD sensors, generate physical quantity e.g. temperature value, illumination level value, pH value etc. These physical quantity values are usually represented within a content field of a <contentIntance> resource, which is represented in a JSON serialization format.

Also the working status of the actuators, such as fans, air conditioner, sprinkler, and roof cover controller is modelled as "ON" and "OFF".

8.5 Configurations

To leverage oneM2M MQTT protocol for the communication between AE/CSE and CSE, devices (including sensors, actuators), IoT applications, farm gateway, the server platform that are deployed in the current use case are required to implement MQTT client library, while the farm gateway and the server platform also need to implement MQTT broker functionalties. Details for implementation of MQTT client library in those entities are out of scope of this use guide. Developers need to refer to other tutorials for implementation of MQTT client library by their own.

Configurations initially specified in section 5.2.3 of TS-0010 [i.4] are applied to the communication scenarios as following:

- AE to IN (section 5.2.3.1) : mapped to communications between IoT applications and the server platform;
- AE to MN (section 5.2.3.2) : mapped to communications between sensors/actuators and the farm gateway;
- MN to IN (section 5.2.3.3) : mapped to communications between the farm gateway and the server platform.

8.6 Topics Used for Initial Registration

8.6.1 Introduction

oneM2M MQTT binding protocol [i.4] specifies the format for initial registration topic as /oneM2M/reg_req/<ORIGINATOR>/<RECEIVER>/<CONTENT-TYPE> and /oneM2M/reg_resp/<ORIGINATOR>/<RECEIVER>/<CONTENT-TYPE>, where the <ORIGINATOR> is set to the credential ID for the registration originator, while the <RECEIVER> is set by the SP-relative-CSE-ID of the registrar CSE omitting the leading slash "/". In current use case where applications are allowed in situations where no Security Association has been established prior to issuing the entity registration request, the Credential-ID 'None' is used.

It also specifes topic format for sending and receiving message as `/oneM2M/req/<ORIGINATOR>/<RECEIVER>/<CONTENT-TYPE>` and `/oneM2M/resp/<ORIGINATOR>/<RECEIVER>/<CONTENT-TYPE>,` where the <ORIGINATOR> is set to the AE-ID (or CSE-ID) of the registration originator where any leading slash "/" is omitted and any slash "/" embedded in the AE-ID (or CSE-ID) is replaced with colon ":", while the <RECEIVER> is set by the SP-relative-CSE-ID of the registrar CSE omitting the leading slash "/".

8.6.2 Subscription Topic

Topic that oneM2M registration originators need to subscribe to the MQTT server accordingly are summarized as following:

- Subscription topic for farm gateway registration with the server platform

  ```
  /oneM2M/resp/CSE3409165/CSE1534123
  ```

- Subscription topic for sensors registration with the farm gateway

  ```
  /oneM2M/reg_resp/{CREDENTIAl_ID_SENSOR01}/CSE3409165
  /oneM2M/reg_resp/{CREDENTIAl_ID_SENSOR02}/CSE3409165
  …
  ```

- Subscription topic for actuators registration with the farm gateway

  ```
  /oneM2M/reg_resp/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165
  /oneM2M/reg_resp/{CREDENTIAl_ID_ACTUATOR02}/CSE3409165
  …
  ```

- Subscription topic for IoT applications registration with the server platform

  ```
  /oneM2M/reg_resp/{CREDENTIAl_ID_IOT_APP01}/CSE1534123
  /oneM2M/reg_resp/{CREDENTIAl_ID_IOT_APP02}/CSE1534123
  …
  ```

- Subscription topic for farm gateway application registration with the farm gateway

  ```
  /oneM2M/reg_resp/{CREDENTIAl_ID_GATEWAY_APP01}/CSE3409165
  ```

8.6.3 Publish Topic

Topic that oneM2M registration originators need to subscribe to the MQTT server accordingly are summarized as following:

- Publish topic for farm gateway registration with the server platform

  ```
  /oneM2M/req/CSE3409165/CSE1534123/JSON
  ```

- Publish topic for sensors registration with the farm gateway

  ```
  /oneM2M/reg_req/{CREDENTIAl_ID_SENSOR01}/CSE3409165/JSON
  /oneM2M/reg_req/{CREDENTIAl_ID_SENSOR02}/CSE3409165/JSON
  …
  ```

- Publish topic for actuators registration with the farm gateway

  ```
  /oneM2M/reg_req/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165/JSON
  /oneM2M/reg_req/{CREDENTIAl_ID_ACTUATOR02}/CSE3409165/JSON
  …
  ```

- Publish topic for IoT applications registration with the server platform

  ```
  /oneM2M/reg_req/{CREDENTIAl_ID_IOT_APP01}/CSE1534123/JSON
  /oneM2M/reg_req/{CREDENTIAl_ID_IOT_APP02}/CSE1534123/JSON
  …
  ```

- Publish topic for farm gateway application registration with the farm gateway

  ```
  /oneM2M/reg_req/{CREDENTIAl_ID_GATEWAY_APP01}/CSE3409165/JSON
  ```

Where `CSE1534123` and `CSE3409165` represents the SP-relative-CSE-ID of the IN-CSE and MN-CSE, which omits the leading slash "/" respectively. The Content-Type in the subscription topic is assigned to `JSON` to indicate the

message (payload) to be received from MQTT server is serialized into JSON format, while the Content-Type in the publish topic is assigned to JSON to indicate the payload within a PUBLISH message being sent by the originator is serialized into JSON format.

8.7 Implementation of Entity Registration

8.7.1 Introduction

The initial registrations of oneM2M entities over MQTT protocol are proceeded in ordered procedures as following:

① An originator having implemented MQTT client library sends a CONNECT packet containing a suite of paramters such as header, protocol level, user name flag, password flag, will retain flag, will qos flag, will flag, clean session flag, keep alive, as well as connect payload etc. that are defined in MQTT protocol specification. This step is used to establish the connection between MQTT client and the MQTT server through TCP.

② The originator sends a SUBSCRIBE packet containing a topic name set `/oneM2M/resp/<ORIGINATOR>/<RECEIVER>` to the MQTT server. Here, we assume that the registar CSE implemented a MQTT client library has subscribed to topic `/oneM2M/req/<ORIGINATOR>/<RECEIVER>` to ensure any registration request targeted to the registrar CSE will be delivered to the registrar CSE by the MQTT server;

③ The originator publishes an initial registration request to the topic `/oneM2M/req/<ORIGINATOR>/<RECEIVER>/<CONTENT_TYPE>` by sending a PUBLISH packet containing a serialized oneM2M entity registration request primitive in the payload, after receiving a SUBACK message indicating successful subscription to the MQTT server.

④ The originator receives a PUBLISH message that is delivered from the MQTT server after registrar CSE sending a PUBLISH message containing the registration response to the topic `/oneM2M/resp/<ORIGINATOR>/<RECEIVER>/<CONTENT-TYPE>`.

8.7.2 Farm Gateway Registration

The farm gateway subscribes to topic `/oneM2M/resp/CSE3409165/CSE1534123` for the registration with the server platform, and sends a PUBLISH packet with target topic set to `/oneM2M/req/CSE3409165/CSE1534123/JSON` to the MQTT server, where the payload of the PUBLISH packet contains the registration request primitive that is serialized into JSON format as following:

```
{
   "fr" : "/CSE3409165",
   "op" : 1,
   "pc" : {
      "m2m:csr" : {
          "csi": "/CSE3409165",
          "cb" : "gateway.provider.com/CSE3409165",
          "rr" : true,
          "poa": ["mqtt://gateway.provider.com"],
          "cst": 2,
          "rn": "farm_gateway"
      }
   },
   "rqi" : "m_createRemoteCSE674504",
   "to" : "/CSE1534123/server",
   "ty" : 16
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `oneM2M/req/CSE3409165/CSE1534123/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The MN-CSE registration request message will be delivered by a MQTT broker to the IN-CSE (the registrar CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive of the registration after the IN-CSE accepts the registration and sends a PUBLISH packet containing response primitive to topic /oneM2M/resp/CSE3409165/CSE1534123/JSON.

The response primitive contained in the PUBLISH packet being delivered to the originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createRemoteCSE674504",
   "pc"  : {
       "m2m:csr" : {
          "ty" : 16,
          "ri": "mncse8301493",
          "pi": "/CSE1534123",
          "ct": "20170524T045938",
          "lt": "20170524T045938",
          "et": "20270523T045938",
          "csi": "/CSE3409165",
          "cb" : "gateway.provider.com/CSE3409165",
          "rr" : true,
          "poa": ["mqtt://gateway.provider.com"],
          "rn": "farm_gateway"
      }
   },
   "to"   : "/CSE3409165/farm_gateway",
   "fr"   : "/CSE1534123"
}
```

8.7.3 Sensors Registration

Sensors have no AE-ID known to the MN-CSE at initial registration process. In this case, Credential-ID is used while subscribing and publishing message to MQTT server.

One sensor subscribes to topic /oneM2M/reg_resp/{CREDENTIAl_ID_SENSOR01}/CSE3409165 to receive the registration response from the farm gateway through message delivering of MQTT server.

The sensor originator sends a PUBLISH packet with target topic set to /oneM2M/reg_req/{CREDENTIAl_ID_SENSOR01}/CSE3409165/JSON to the MQTT server, where the payload of the PUBLISH packet contains the registration request primitive that is serialized into JSON format as following:

```
{
   "fr" : "S",
   "op" : 1,
   "pc" : {
      "m2m:ae" : {
          "api": "A01.com.farm.sensor01",
          "rr" : true,
          "poa": ["mqtt://sensor01.farm.com"],
          "rn" : "sensor_ae01"
      }
   },
   "rqi": "m_createAE142308",
   "to" : "/CSE3409165/farm_gateway",
   "ty" : 2
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to QoS 1 following oneM2M MQTT binding, and the Topic is set to oneM2M/reg_req/{CREDENTIAl_ID_SENSOR01}/CSE3409165/JSON.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The AE registration request message will be delivered by a MQTT broker to the MN-CSE (the registrar CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive of the registration after the MN-CSE accepts the registration and sends a PUBLISH packet containing response primitive to topic /oneM2M/reg_resp/{CREDENTIAl_ID_SENSOR01}/CSE3409165/JSON.

The response primitive contained in the PUBLISH packet being delivered to the originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createAE142308",
   "pc"  : {
       "m2m:ae" : {
          "ty" : 2,
          "ri" : "ae0349801231",
          "pi" : "mncse8301493",
          "ct" : "20170524T045938",
          "lt" : "20170524T045938",
          "et" : "20270523T045938",
          "api": "A01.com.farm.sensor01",
          "rr" : true,
          "aei": "S108653822141",
          "poa": ["mqtt://sensor01.farm.com"],
          "rn" : "sensor_ae01"
      }
   },
   "to"   : "/CSE3409165/farm_gateway/sensor_ae01",
   "fr"   : "/CSE3409165"
}
```

For any other sensor e.g. sensor_ae02, sensor_ae03 etc., the registration procedure is similar as sensor_ae01. The AE-ID S108653822141 can be used for UPDATE, RETRIEVE, and DELETE of AE resource of sensor01, as well as any CREATE, RETRIEVE, UPDATE, and DELETE operation for its child resources.

8.7.4 Actuators Registration

Actuators have no AE-ID known to the MN-CSE at initial registration process. In this case, Credential-ID is used while subscribing and publishing message to MQTT server.

One actuator subscribes to topic /oneM2M/reg_resp/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165 to receive the registration response from the farm gateway through message delivering of MQTT server.

The sensor originator sends a PUBLISH packet with target topic set to /oneM2M/reg_req/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165/JSON to the MQTT server, where the payload of the PUBLISH packet contains the registration request primitive that is serialized into JSON format as following:

```
{
   "fr" : "S",
   "op" : 1,
   "pc" : {
      "m2m:ae" : {
          "api": "B01.com.farm.actuator01",
          "rr" : true,
          "poa": ["mqtt://actuator01.farm.com"],
          "rn" : "actuator_ae01"
```

```
      }
   },
   "rqi": "m_createAE2542318",
   "to" : "/CSE3409165/farm_gateway",
   "ty" : 2
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `oneM2M/reg_req/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The AE registration request message will be delivered by a MQTT broker to the MN-CSE (the registrar CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive of the registration after the MN-CSE accepts the registration and sends a PUBLISH packet containing response primitive to topic `/oneM2M/reg_resp/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165/JSON`.

The response primitive contained in the PUBLISH packet being delivered to the originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createAE2542318",
   "pc"  : {
       "m2m:ae" : {
          "ty" : 2,
          "ri" : "ae1359351228",
          "pi" : "mncse8301493",
          "ct" : "20170524T055938",
          "lt" : "20170524T055938",
          "et" : "20270523T055838",
          "api": "B01.com.farm.actuator01",
          "rr" : true,
          "aei": "S208650938180",
          "poa": ["mqtt://actuator01.farm.com"],
          "rn" : "actuator_ae01"
       }
   },
   "to"   : "/CSE3409165/farm_gateway/actuator_ae01",
   "fr"   : "/CSE3409165"
}
```

For any other actuator e.g. `actuator_ae02`, `actuator_ae03` etc., the registration procedure is similar as `actuator_ae01`.
The AE-ID `S208650938180` can be used for UPDATE, RETRIEVE, and DELETE of AE resource of actuator01, as well as any CREATE, RETRIEVE, UPDATE, and DELETE operation for its child resources.

8.7.5 IoT Applications Registration

IoT applications have no AE-ID known to the IN-CSE at initial registration process. In this case, Credential-ID is used while subscribing and publishing message to MQTT server.

One IoT application subscribes to topic `/oneM2M/reg_resp/{CREDENTIAl_ID_IOT_APP01}/CSE1534123` to receive the registration response from the server platform through message delivering of MQTT server.

The IoT application sends a PUBLISH packet with target topic set to `/oneM2M/reg_req/{CREDENTIAl_ID_IOT_APP01}/CSE1534123/JSON` to the MQTT server, where the payload of the PUBLISH packet contains the registration request primitive that is serialized into JSON format as following:

```
{
   "fr" : "S",
```

```
   "op" : 1,
   "pc" : {
      "m2m:ae" : {
          "api": "D01.com.provider.iotapp01",
          "rr" : true,
          "poa": ["mqtt://iotapp01.provider.com"],
          "rn" : "app_ae01"
      }
   },
   "rqi": "m_createAE3567902",
   "to" : "/CSE1534123/server",
   "ty" : 2
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `/oneM2M/reg_req/{CREDENTIAl_ID_IOT_APP01}/CSE1534123/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The IoT application registration request message will be delivered by a MQTT broker to the IN-CSE (the registrar CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive of the registration after the IN-CSE accepts the registration and sends a PUBLISH packet containing response primitive to topic `/oneM2M/reg_resp/{CREDENTIAl_ID_IOT_APP01}/CSE1534123/JSON`.

The response primitive contained in the PUBLISH packet being delivered to the originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createAE3567902",
   "pc"  : {
       "m2m:ae" : {
          "ty" : 2,
          "ri" : "ae4639398345",
          "pi" : "incse8563723426",
          "ct" : "20170524T07938",
          "lt" : "20170524T075938",
          "et" : "20270523T075838",
          "api": "D01.com.provider.iotapp01",
          "rr" : true,
          "aei": "S317312761857",
          "poa": ["mqtt://iotapp01.provider.com"],
          "rn" : "app_ae01"
      }
   },
   "to"   : "/CSE1534123/server/app_ae01",
   "fr"   : "/CSE1534123"
}
```

For any other IoT applications e.g. `app_ae02`, `app_ae03` etc., the registration procedure is similar as `app_ae01`.
The AE-ID `S317312761857` can be used for UPDATE, RETRIEVE, and DELETE of AE resource of IoT application01, as well as any CREATE, RETRIEVE, UPDATE, and DELETE operation for its child resources.

8.7.6 Farm Gateway Application Registration

The farm gateway application has no AE-ID known to the MN-CSE at initial registration process. In this case, Credential-ID is used while subscribing and publishing message to MQTT server.

The farm gateway application subscribes to topic `/oneM2M/reg_req/{CREDENTIAl_ID_GATEWAY_APP01}/CSE3409165` to receive the registration response from the farm gateway through message delivering of MQTT server.

The farm gateway application sends a PUBLISH packet with target topic set to /oneM2M/reg_req/{CREDENTIAl_ID_ACTUATOR01}/CSE3409165/JSON to the MQTT server, where the payload of the PUBLISH packet contains the registration request primitive that is serialized into JSON format as following:

```
{
   "fr" : "S",
   "op" : 1,
   "pc" : {
      "m2m:ae" : {
          "api": "C01.com.farm.app01",
          "rr" : true,
          "poa": ["mqtt://gatewayapp.farm.com"],
          "rn" : "gateway_ae"
      }
   },
   "rqi": "m_createAE3942104",
   "to" : "/CSE3409165/farm_gateway",
   "ty" : 2
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to QoS 1 following oneM2M MQTT binding, and the Topic is set to oneM2M/reg_req/{CREDENTIAl_ID_GATEWAY_APP01}/CSE3409165/JSON.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The MN-AE registration request message will be delivered by a MQTT broker to the MN-CSE (the registrar CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive of the registration after the MN-CSE accepts the registration and sends a PUBLISH packet containing response primitive to topic /oneM2M/reg_resp/{CREDENTIAl_ID_GATEWAY_APP01}/CSE3409165/JSON.

The response primitive contained in the PUBLISH packet being delivered to the originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createAE3942104",
   "pc"  : {
       "m2m:ae" : {
          "ty" : 2,
          "ri" : "ae3394315272",
          "pi" : "mncse8301493",
          "ct" : "20170524T065938",
          "lt" : "20170524T065938",
          "et" : "20270523T065838",
          "api": "C01.com.farm.app01",
          "rr" : true,
          "aei": "S336593381621",
          "poa": ["mqtt://gatewayapp.farm.com"],
          "rn" : "gateway_ae"
      }
   },
   "to"   : "/CSE3409165/farm_gateway/gateway_ae",
   "fr"   : "/CSE3409165"
}
```

8.8 Topics for Request and Response

Topics that AE originator subscribes for listening PUBLISH packets from MQTT server:

- AE sensor_ae01 subscribes to topic /oneM2M/resp/S108653822141/CSE3409165

  Where /S108653822141 is a SP-relative-AE-ID of the sensor_ae01, and /CSE3409165 is a SP-relative-CSE-ID of the MN-CSE

- AE actuator_ae01 subscribes to topic /oneM2M/resp/S208650938180/CSE3409165

  Where /S208650938180 is a SP-relative-AE-ID of the actuator_ae01

- Gateway application gateway_ae subscribes to topic /oneM2M/resp/S336593381621/CSE3409165

  Where /S336593381621 is a SP-relative-AE-ID of the gateway_ae

- IoT application app_ae01 subscribes to topic /oneM2M/resp/S317312761857/CSE1534123

  Where /S317312761857is a SP-relative-AE-ID of the app_ae01

8.9 Initial Resource Generation

8.9.1 Container Creation for Sensors

The AE hoted in a sensor subscribes to topic /oneM2M/resp/S108653822141/CSE3409165 to receive response from the MQTT server. The AE originator sends a PUBLISH packet to the MQTT server, where the payload of the PUBLISH packet contains a container create request primitive that is serialized into JSON format as following:

```
{
   "fr" : "/S108653822141",
   "op" : 1,
   "pc" : {
      "m2m:cnt" : {
          "rn": "cont_monitor01"
      }
   },
   "rqi" : "m_createCont291286",
   "to" : "/CSE3409165/farm_gateway/sensor_ae01",
   "ty" : 3
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to QoS 1 following oneM2M MQTT binding, and the Topic is set to /oneM2M/req/S108653822141/CSE3409165/JSON.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The container creation request message will be delivered by a MQTT server to the MN-CSE (the receiver CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive for the container creation after the MN-CSE accepts the container create request and sends a PUBLISH packet containing response primitive to topic /oneM2M/resp/S108653822141/CSE3409165/JSON.

The response primitive contained in the PUBLISH packet being delivered to the AE originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createCont291286",
   "pc"  : {
```

```
        "m2m:cnt" : {
            "ty" : 3,
            "ri" : "cont92310496",
            "pi" : "ae0349801231",
            "ct" : "20170525T045938",
            "lt" : "20170525T045938",
            "et" : "20270524T045938",
            "st" : 0,
            "cni": 0,
            "cbs": 0,
            "rn" : "cont_monitor01"
        }
    },
    "to"    : "/S108653822141",
    "fr"    : "/CSE3409165"
}
```

This API can be applied to create containers `cont_monitor02`, `cont_monitor03` as a child resource of AE `sensor_ae02`, `sensor_ae03`, respectively.

8.9.2 Container Creation for Actuators

The AE hoted in an actuator subscribes to topic `/oneM2M/resp/S208650938180/CSE3409165` to receive response from the MQTT server. The AE originator sends a PUBLISH packet to the MQTT server, where the payload of the PUBLISH packet contains a container create request primitive that is serialized into JSON format as following:

```
{
    "fr" : "/S208650938180",
    "op" : 1,
    "pc" : {
        "m2m:cnt" : {
            "rn": "cont_ctrl01"
        }
    },
    "rqi" : "m_createCont301272",
    "to" : "/CSE3409165/farm_gateway/actuator_ae01",
    "ty" : 3
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `/oneM2M/req/S208650938180/CSE3409165/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The container creation request message will be delivered by a MQTT server to the MN-CSE (the receiver CSE) that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive for the container creation after the MN-CSE accepts the container create request and sends a PUBLISH packet containing response primitive to topic `/oneM2M/resp/S208650938180/CSE3409165/JSON.`

The response primitive contained in the PUBLISH packet being delivered to the AE originator by the MQTT server is shown as following:

```
{
    "rsc" : 2001,
    "rqi" : "m_createCont301272",
    "pc"  : {
        "m2m:cnt" : {
            "ty" : 3,
            "ri" : "cont95827628",
            "pi" : "ae1359351228",
            "ct" : "20170525T045938",
            "lt" : "20170525T045938",
```

```
            "et" : "20270524T045938",
            "st" : 0,
            "cni": 0,
            "cbs": 0,
            "rn" : "cont_ctrl01"
        }
    },
    "to"   : "/S208650938180",
    "fr"   : "/CSE3409165"
}
```

This API can be applied to create containers `cont_ctrl02`, `cont_ctrl03` as a child resource of AE `actuator_ae02`, `actuator_ae03`, respectively.

8.9.3 Subscription to Monitor Sensors

The AE hoted in an IoT application subscribes to topic `/oneM2M/resp/S317312761857/CSE1534123` to receive subscription response from the MQTT server. The AE originator sends a PUBLISH packet to the MQTT server, where the payload of the PUBLISH packet contains a subscription create request primitive that is serialized into JSON format as following:

```
{
    "fr" : "/S317312761857",
    "op" : 1,
    "pc" : {
        "m2m:sub" : {
            "rn": "sub_monitor01",
            "enc": {
                "net": [3]
            },
            "nu": "/S317312761857",
            "nct": 1
        }
    },
    "rqi" : "m_createSub41226",
    "to" : "/CSE1534123/server/farm_gateway/sensor_ae01/cont_monitor01",
    "ty" : 23
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `/oneM2M/req/S317312761857/CSE1534123/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The subscription creation request message will be delivered by a MQTT server to the IN-CSE (the Transit CSE) then forwarded to MN-CSE (the Hosting CSE) that implements a MQTT client. Then the AE originator will be notified with a PUBLISH packet containing the response primitive for the subscription creation after the MN-CSE accepts the subscription create request and sends a PUBLISH packet containing response primitive to topic `/oneM2M/resp/S317312761857/CSE1534123/JSON.`

The response primitive contained in the PUBLISH packet being delivered to the AE originator by the MQTT server is shown as following:

```
{
    "rsc" : 2001,
    "rqi" : "m_createSub41226",
    "pc"  : {
        "m2m:sub" : {
            "ty" : 23,
            "ri" : "sub956204321",
            "pi" : "cont92310496",
            "ct" : "20170525T065938",
            "lt" : "20170525T065938",
```

```
            "et" : "20270524T065938",
            "enc": {
                "net": [3]
            },
            "nu": "/S317312761857",
            "nct": 1,
            "rn" : "sub_monitor01"
        }
    },
    "to"    : "/S317312761857",
    "fr"    : "/CSE1534123"
}
```

This API can be applied to create subscription `sub_monitor02`, `sub_monitor03` as a child resource of container `cont_monitor02`, `cont_monitor03`, respectively.

8.9.4 Subscription to Control Actuators

The MN-AE hoted in the farm gateway subscribes to topic `/oneM2M/resp/S336593381621/CSE3409165` to receive subscription response from the MQTT server. The AE originator sends a PUBLISH packet to the MQTT server, where the payload of the PUBLISH packet contains a subscription create request primitive that is serialized into JSON format as following:

```
{
    "fr" : "/S336593381621",
    "op" : 1,
    "pc" : {
        "m2m:sub" : {
            "rn": "sub_ctrl01",
            "enc": {
                "net": [3]
            },
            "nu": "/S336593381621",
            "nct": 1
        }
    },
    "rqi" : "m_createSub45826",
    "to" : "/CSE1534123/server/farm_gateway/actuator_ae01/cont_ctrl01",
    "ty" : 23
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `/oneM2M/req/S336593381621/CSE3409165/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The subscription creation request message will be delivered by a MQTT server to the IN-CSE (the Transit CSE) then forwarded to MN-CSE (the Hosting CSE) that implements a MQTT client. Then the AE originator will be notified with a PUBLISH packet containing the response primitive for the subscription creation after the MN-CSE accepts the subscription create request and sends a PUBLISH packet containing response primitive to topic `/oneM2M/resp/S336593381621/CSE3409165/JSON`.

The response primitive contained in the PUBLISH packet being delivered to the AE originator by the MQTT server is shown as following:

```
{
    "rsc" : 2001,
    "rqi" : "m_createSub45826",
    "pc"  : {
        "m2m:sub" : {
            "ty" : 23,
            "ri" : "sub998412563",
            "pi" : "cont95827628",
```

```
            "ct" : "20170525T075938",
            "lt" : "20170525T075938",
            "et" : "20270524T075938",
            "enc": {
                "net": [3]
            },
            "nu": "/S336593381621",
            "nct": 1,
            "rn" : "sub_ctrl01"
        }
    },
    "to"   : "/S336593381621",
    "fr"   : "/CSE3409165"
}
```

This API can be applied to create subscription `sub_ctrl02`, `sub_ctrl03` as a child resource of container `cont_ctrl02`, `cont_ctrl02`, respectively.

8.10 Devices Controlling

The devices controlling in the current use case is designed to be implemented via subscription&notification mechanisms, i.e. the notification receiver e.g. MN-AE in this use case will be notified with the device controlling command to one or multiple target device(s), where the controlling command might be initialized by any authenticated originator, e.g. IoT applications in this use case.

- Step-I: Uploading device controlling command within oneM2M resource primtive

IoT applications (ADN-AE) initializes a contentInstance create request target to the device monitoring container e.g. `cont_ctrl01`, which contains device controlling commands in the primitiveContent of the contentInstance resource.

The ADN-AE hosted in an IoT application subscribes to topic `/oneM2M/resp/S317312761857/CSE1534123` to receive response from the MQTT server. The AE originator sends a PUBLISH packet to the MQTT server, where the payload of the PUBLISH packet contains a contentInstance create request primitive that is serialized into JSON format as following:

```
{
   "fr" : "/S317312761857",
   "op" : 1,
   "pc" : {
      "m2m:cin" : {
          "cnf": "text/plains:0",
          "con": "ON"
      }
   },
   "rqi" : "m_createCin592082",
   "to" : "/CSE1534123/server/farm_gateway/actuator_ae01/cont_ctrl01",
   "ty" : 4
}
```

In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `/oneM2M/req/S317312761857/CSE1534123/JSON`.

As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

The contentInstance creation request message will be delivered by a MQTT server to the MN-CSE that implements a MQTT client. Then the originator will be notified with a PUBLISH packet containing the response primitive for the container creation after the MN-CSE accepts the container create request and sends a PUBLISH packet containing response primitive to topic `/oneM2M/resp/S317312761857/CSE1534123/JSON.`

The response primitive contained in the PUBLISH packet being delivered to the AE originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createCin592082",
   "pc"  : {
       "m2m:cin" : {
          "ty" : 4,
          "ri" : "cin989793432",
          "pi" : "cont95827628",
          "ct" : "20170525T065938",
          "lt" : "20170525T065938",
          "et" : "20270524T065938",
          "st" : 0,
          "cnf": "text/plains:0",
          "cs" : 2,
          "con": "ON",
          "rn" : "cin_d34akgf56sg"
      }
   },
   "to"   : "/S317312761857",
   "fr"   : "/CSE1534123"
}
```

This API can be applied to create contentInstance as a child resource of `cont_ctrl02`, `cont_ctrl03` under AE `actuator_ae02`, `actuator_ae03`, respectively.

- Step II: Notifying farm gateway application to actuate devices

Whenever there is new child resource is created under the subscribed-to resource e.g. `cont_ctrl01`, notification primitive will be delivered to the notification receiver e.g. the MN-AE in this use case by the MQTT server. PrimitiveContent included within the notification primitive is determined by the notificationContentType, while the notification receiver is configured via the notificationURI attribute in the subscription request. Notification event type (short for `net`) field set to `3` to indicate the notification criteria is configured to event of creation of direct child resource of the subscribed-to resource.

The notification primitive contained in the PUBLISH packet being delivered to the MN-AE by the MQTT server is shown as following:

```
{
   "fr" : "/CSE1534123",
   "op" : 5,
   "pc" : {
         "m2m:sgn":
         {
           "nev":{
              "rep":
               {
                 "cin":
                  {
                    "cnf": "text/plain:0",
                    "con": "ON"
                  }
               },
               "net" :[3]
            },
            "sur":"/CSE3409165/sub998412563"
         }
   },
   "rqi" : "m_notify982340",
   "to" : "/S336593381621"
}
```

When the gateway application MN-AE gateway_ae receives notification successfully from IN-CSE, the MN-AE gateway_ae sends a notification response primitive to acknowledge the successful receiving of notification message from IN-CSE.

The notification response primitive contained in the PUBLISH packet being delivered to the IN-CSE by the MQTT server is shown as following:

```
{
      "rsc" : 2000,
      "rqi" : "m_notify982340",
      "fr"  : "/S336593381621",
      "to"  : "/CSE1534123"
}
```

The notification primitive can be parsed to abstract the device controlling command to control specific device. The implementation details is out of the scope of this user guide.

8.11 Devices Sensing Data Monitoring

The sensor monitoring in this use case is subject to the notification for the generation of sensing data in specific sensors, and it is designed to be implemented via subscription&notification mechanisms, i.e. the notification receiver e.g. IoT applications in this use case will be notified with the physical quantity data from sensors e.g. updated indoor temperature value, or pH value in soil of the argriculture farm.

- Step-I: Automatically uploading sensing data by sensors in periodical manner

  Sensors collect and upload the physical quantity into the server platform through farm gateway. The sensing data i.e. physical quantity is represented as a contentInstance in the server platform.

  The ADN-AE hosted in sensors subscribes to topic `/oneM2M/resp/S108653822141/CSE3409165` to receive response from the MQTT server. The AE originator sends a PUBLISH packet to the MQTT server, where the payload of the PUBLISH packet contains a contentInstance create request primitive that is serialized into JSON format as following:

  ```
  {
     "fr" : "/S108653822141",
     "op" : 1,
     "pc" : {
        "m2m:cin" : {
            "cnf": "text/plains:0",
            "con": "25"
        }
     },
     "rqi" : "m_createCin642126",
     "to" : "/CSE3409165/farm_gateway/sensor_ae01/cont_monitor01",
     "ty" : 4
  }
  ```

  In addition, when serializing a PUBLISH packet, DUP, QoS, and Retain, Topic, and Packet Identifier fields need to be set, where the QoS is set to `QoS 1` following oneM2M MQTT binding, and the Topic is set to `/oneM2M/req/S108653822141/CSE3409165/JSON`.

  As a result of QoS 1, PUBACK packet containing the Packet Identifier would be returned to the originator to indicate the receiving status of the PUBLISH packet.

  The contentInstance creation request message will be delivered by a MQTT server to the MN-CSE, and then the IoT application will be notified with a PUBLISH packet containing the response primitive for the contentInstance creation after the MN-CSE accepts the contentInstance create request and sends a PUBLISH packet containing response primitive to topic `/oneM2M/resp/S108653822141/CSE3409165/JSON.`

  The response primitive contained in the PUBLISH packet being delivered to the AE originator by the MQTT server is shown as following:

```
{
   "rsc" : 2001,
   "rqi" : "m_createCin642126",
   "pc"  : {
       "m2m:cin" : {
          "ty" : 4,
          "ri" : "cin990931789",
          "pi" : "cont92310496",
          "ct" : "20170525T075938",
          "lt" : "20170525T075938",
          "et" : "20270524T075938",
          "st" : 0,
          "cnf": "text/plains:0",
          "cs" : 2,
          "con": "25",
          "rn" : "cin_jklqwewe56er"
       }
   },
   "to"   : "/S108653822141",
   "fr"   : "/CSE1534123"
}
```

This API can be applied to create contentInstance as a child resource of `cont_monitor02`, `cont_monitor03` under AE `sensor_ae02`, `sensor_ae03`, respectively.

- Step II: Notifying IoT application for the new sensing data

Whenever there is new child resource is created under the subscribed-to resource e.g. container `cont_monitor01`, notification primitive will be delivered by the MQTT server to the notification receiver e.g. the IoT application ADN-AE app_ae01 (`S317312761857`) in this use case. Notification event type (short for `net`) field set to `3` to indicate the notification criteria is configured to event of creation of direct child resource of the subscribed-to resource.

The notification primitive contained in the PUBLISH packet being delivered to the IoT application by the MQTT server is shown as following:

```
{
   "fr" : "/CSE1534123",
   "op" : 5,
   "pc" : {
         "m2m:sgn":
         {
          "nev":{
             "rep":
              {
                "cin":
                 {
                  "cnf": "text/plain:0",
                  "con": "25"
                 }
              },
              "net":[3]
           },
           "sur":"/CSE3409165/sub956204321"
         }
   },
   "rqi" : "m_notify995826",
   "to" : "/S317312761857"
}
```

When the IoT application ADN-AE app_ae01 receives notification successfully from IN-CSE, the ADN-AE app_ae01 sends a notification response primitive to acknowledge the successful receiving of notification message from IN-CSE.

The notification response primitive contained in the PUBLISH packet being delivered to the IN-CSE by the MQTT server is shown as following:

```
{
```

```
      "rsc" : 2000,
      "rqi" : "m_notify995826",
      "fr"  : "/S317312761857",
      "to"  : "/CSE1534123"
}
```

9 Conclusion

The smart farm monitoring and control use case demonstrated the utilization of a suite of oneM2M high level procedures such as entity registration, initial resource creation, subscription and notifications etc. to implement the smart monitoring and controlling of devices deployed in an agriculture farm scenario.

Those oneM2M high level procedures are implemented via MQTT protocol binding with JSON serialization.

Annex A: Implementation of Connection to MQTT

The two client parties (AE and CSE or CSE and CSE) need to connect to a common MQTT server in order to enable the communication between each other.

The MQTT client sends a CONNECT packet to the MQTT server to initialize the connection with MQTT server. The CONNECT packet contains a suite of paramters such as header, protocol level, user name flag, password flag, will retain flag, will qos flag, will flag, clean session flag, keep alive, as well as connect payload etc. that are defined in MQTT protocol specification. This step is used to establish the connection between MQTT client and the MQTT server through TCP.

Annex B: Using Credential-ID in Initial Registration Progress

During the initial phase, the AE-ID of the devices such as sensors, actuators, and IoT applications might not be known to the receiver CSE. For these scenarios, the originator can use Credential-ID, which is formed by MQTT server during a successful Security Association Establishment process with a MQTT client, for subscribing topics and publishing messages to MQTT server.

The Credential-ID is refered to TS-0003[i.6] Section 10.4. The Credential-ID consists of

- A type-ID part. The type-ID part is a positive integer defined by datatype sec:credIDTypeID.
- A value part which contains a globally-unique identifier for the entity's credential. The value part may use the Roman alphabet, numerals, '.', '_', '-', and '@'.

The Credential-ID is formed by concatenating the type part, the character '-' and the value part.

Annex C: Implementation of Topic Subscription to MQTT Server

Each client party (AE or CSE) need sto subscribe to a specific topic located in a MQTT server in order to receive the published messages that are associated to that topic.

The MQTT client sends a SUBSCRIBE packet to the MQTT server to subscribe a topic that is located in the MQTT server. The SUBSCRIBE packet contains a suite of paramters such as header, packet identifier, payload which contains topic filter and requested qos.

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