|  |
| --- |
| Input Contribution |
| Meeting ID\* | SDS#40 |
| Title:\* | TR-0059-platform\_discovery\_key\_issue |
| Source:\* | JaeSeung Song, KETI, jssong@sejong.ac.kr Youngjin Na, Hyundai Motors, yjra@hyundai.comMinbyeong Lee, Hyundai Motors, minbyeong.lee@hyundai.com  |
| Date:\* | 2019-05-10 |
| Input related to\* | TR-0059 Services and Platform Discovery  |
| Intended purpose ofdocument:\* | [x]  Decision[ ]  Discussion[ ]  Information[ ]  Other <specify> |
| Impacted other TS/TR(s) | N/A |
| Decision requested or recommendation:\* | Introducing Key Issue regarding oneM2M platform discvoery  |
| Template Version: January 2017 (Do not modify) |

**oneM2M Notice**

The document to which this cover statement is attached is submitted to oneM2M. Participation in, or attendance at, any activity of oneM2M, constitutes acceptance of and agreement to be bound by terms of the Working Procedures and the Partnership Agreement, including the Intellectual Property Rights (IPR) Principles Governing oneM2M Work found in Annex 1 of the Partnership Agreement.

# Introduction

**1. Motivation:**

Now a days oneM2M systems are used in many smart city projects and trials. Let us consider a smart city having a large number of oneM2M platforms (i.e., IN-CSEs from different vendors and service providers) providing various IoT services via various IoT service providers. Then we can easily come up with the following platform discovery issues:

* How to find IN-CSEs that an IoT application wants to use among a large amount of IoT service platforms in a city?
* In some cases, IoT applications do not necessarily need to know which provider provides which service. There are plenty of public IoT services from different IoT platforms (e.g., smart parking platforms supported by a local government)
* What if Alice (from SanDiego) visits Seoul. Could Alice discovers available IN-CSEs supporting smart parking with her oneM2M smartphone application? (platforms and services should be discoverable)
	+ Alice may want to discovery platforms then uses available services from the selected oneM2M platform.
	+ Alice may want to discovery a specific IoT service (e.g., smart parking) from where she is located in.
* Could IoT applications know which IN-CSEs provide service for V2X or Smart Parking?
* Do we know how many oneM2M IoT service platforms are running over the world?

**2. Proposed Solutions:**

This contribution includes a proposed solution for oneM2M platform discovery and service provisioning via visited oneM2M CSE. The solution is based on a registry entity managing available oneM2M CSEs. The solution involves a oneM2M CSEs performing registration to a oneM2M DNS-like registry server. By performing these procedures a oneM2M entity can discover available oneM2M platforms whereever it is located in.

### -----------------------Start of change 1-------------------------------------------

### 6.2.x Key Issues on oneM2M CSE discovery and local service provisioning

Discovery of oneM2M CSEs is the capability for oneM2M applications to discover available oneM2M CSEs. In order to support oneM2M CSE discovery, a registry entity can be introduced. Discovery information for a oneM2M CSE can be published to the registry such as point of access, deployed location, supported oneM2M services and features. For example, oneM2M CSE discovery can be based on information such as discovering oneM2M CSEs in Seoul, discover oneM2M CSEs supporting smart parking services, discover oneM2M CSEs supporting MQTT. The registry entity can be queried to check the status and availability of registered oneM2M CSEs. Once a oneM2M CSE is discovered, a oneM2M application can register itself to the discovered oneM2M CSE to receive required oneM2M services.

The following key issues may be summarized:

1. The oneM2M System currently does not support methods to allow oneM2M entities to discover available oneM2M CSEs
2. Entities in the oneM2M System should be able to check the status and availability of other oneM2M CSEs so that reliability of oneM2M CSE discovery can be increased.

### -----------------------End of change 1 -------------------------------------------

### -----------------------Start of change 2 -------------------------------------------

## 8.X Solution: oneM2M CSEs Discovery

### 8.X.1 Overview

To enable a oneM2M entity to discover available oneM2M CSEs based on certain criteria (such as location, supporting services), this solution proposes to introduce a Registry server that stores oneM2M CSE discovery information and that can be queried by entities in the oneM2M System. This solution is ideal for the case where multiple oneM2M CSEs are available in a smart city or state or a country, so that a oneM2M application can select a required oneM2M CSEs even it does not have any prior information about available oneM2M CSEs. Potential solutions should guarantee that only properly working oneM2M CSEs are discoverable based on their availability status.

### 8.X.2 Solution Description using Registry

### This clause describes the procedure for a centralalized CSE discovery registry-based. In order to discover available oneM2M CSEs, a service registry that stores information regarding available oneM2M CSEs is required. A proper description of oneM2M CSEs and registration procedures are needed to be defined.

 

**Figure 8.X.2-1: High-level procedure for a Registry based oneM2M**

**CSE discovery**

CSEs discovery is the process of locating CSE information and retrieving CSE descriptions that have been previously stored within the Registry. oneM2M AEs query the Registry for information regarding available oneM2M CSEs matching the requirements of the AEs. Examples of CSE information that can be stored in the Registry can include information the following:

* Contact of Address: IP Address of a CSE
* Port number: Port number for a CSE
* Name: Name of a CSE
* Status: Liveness information (whether a CSE is running or not)
* Location: Geography information where a CSE is located in.
* Profile: Information about which product profile is referred by a CSE
* Type of CSE (MN-CSE or IN-CSE)
* Supporting public services (e.g., smart parking, smart home)
* Maintenance information (for example, from 01:00 ~ 02:00)
* Access information (or credentials)
* Physical capability (e.g., available storage, memory, etc.)

After the CSE discovery procedure is completed, the oneM2M AE can know the exact location of a needed oneM2M CSE via PoA, its capabilities, and how to communicate with it.

The registry for oneM2M CSEs is providing a similar service to the oneM2M App-ID registry. The PoA of the central Registry can be pre-provisioned.

As there exist oneM2M CSEs which are not available because of various reasons such as maintenance, out of order and temporary disorder, the registry has to check the availability status, which refers to the fact that a CSE is running or not, of the registered oneM2M CSEs periodically.

### 8.X.3 Solution Description using a dedicated resource for CSE discovery

### This clause describes a mechanism for oneM2M CSE discovery based on a dedicated resource named <*cseRegistry*> and <*cseRegistryList*>. In order to discover oneM2M CSEs, the <*cseRegistry*> resource can be used to keep all the available oneM2M CSEs and their description to be discovered either open to the public or have a business relationship. <*cseRegistryList*> resource is used to manage <*cseRegistry*> as shown in Figure 8.x.3-1 For example, if a citizen from smart city A visits smart city B, the citizen may want to find out available parking lots using oneM2M smart parking application. In this case, the oneM2M smart parking application can discover available CSEs supporting smart parking service in smart city B via looking into the <*cseRegistry*> resource. As another example, an oneM2M application running on a moving vehicle tries to find out available CSEs that are covering the routes where the vehicle is going to take and supporting Edge/Fog capability.

In this case, synchronization between the attributes of remote CSEs in the <*cseRegistry*> resource and the original resource (i.e., available oneM2M CSEs) shall be the responsibility of the <*cseRegistry*> resource hosting CSE. In addition, the hosting CSE performs the liveness check for the all the remoteCSEs managed in the <*cseRegistry*> resource.

An AE or other CSE can request the source CSE for adding its information to the <*caseRegistry*> resource of the target Hosing CSE.

**Step 001**: The Originator of a Request initiating the publication of the description of a CSE to a target CSE. The request should include the target CSE address and the indication of publishing CSE description. The target CSE can also be the Originator of a Request.

**Step 002**: The Hosting CSE then prepares a CREATE request message of itself to the given target CSE’s <*cseRegistry*> resource. The message is composed of the address of itself, access token to be used for basic authentication, supporting services and features.

**Step 003**: The Hosting CSE sends the CREATE request message to the target CSE

**Step 004**: The Target CSE adds a new CSE description to the <*cseRegistry*> resource

 

Figure 8.x.3-1: Resource structure showing how cseRegistry and cseRegistryList are composed of

### Editor’s note: It is FFS to compare CSE information managed by <cseRegistry> with the one from existing CSE related resources (e.g., remoteCSE, decendentCSE).

### -----------------------End of change 2 -------------------------------------------