Service Layer API for oneM2M

Final Draft

93 Pages

~~*Text in Red is here to help you. Delete it when you have followed the instructions. The <RFC Title> can be set from the File>Properties:User Defined menu. To update it onscreen, press F9. To update all of the fields in the document Select All (CTRL-A), then hit F9. Set the release level by selecting one from: Draft, Final Draft, Release. The date is set automatically when the document is saved.*~~

Abstract

~~10 point Arial Centered.~~

oneM2M is standard organization and specifies middleware for IoT, called Common Services Entities (CSE). Application can access functionality in CSE with RESTful operations, which are Create, Retrieve, Update, Delete and Notify. oneM2M allows variety of communication methods, 4 protocol bindings (HTTP, MQTT, CoAP, Websocket) and 3 serializations (XML, JSON, CBOR). This RFC describes the way to provide high level API for oneM2M RESTful operations hiding the difference of variety of communication methods.

0 Document Information

0.1 License

DISTRIBUTION AND FEEDBACK LICENSE, Version 2.0

The OSGi Alliance hereby grants you a limited copyright license to copy and display this document (the “Distribution”) in any medium without fee or royalty. This Distribution license is exclusively for the purpose of reviewing and providing feedback to the OSGi Alliance. You agree not to modify the Distribution in any way and further agree to not participate in any way in the making of derivative works thereof, other than as a necessary result of reviewing and providing feedback to the Distribution. You also agree to cause this notice, along with the accompanying consent, to be included on all copies (or portions thereof) of the Distribution. The OSGi Alliance also grants you a perpetual, non-exclusive, worldwide, fully paid-up, royalty free, limited license (without the right to sublicense) under any applicable copyrights, to create and/or distribute an implementation of the Distribution that: (i) fully implements the Distribution including all its required interfaces and functionality; (ii) does not modify, subset, superset or otherwise extend the OSGi Name Space, or include any public or protected packages, classes, Java interfaces, fields or methods within the OSGi Name Space other than those required and authorized by the Distribution. An implementation that does not satisfy limitations (i)-(ii) is not considered an implementation of the Distribution, does not receive the benefits of this license, and must not be described as an implementation of the Distribution. "OSGi Name Space" shall mean the public class or interface declarations whose names begin with "org.osgi" or any recognized successors or replacements thereof. The OSGi Alliance expressly reserves all rights not granted pursuant to these limited copyright licenses including termination of the license at will at any time.

EXCEPT FOR THE LIMITED COPYRIGHT LICENSES GRANTED ABOVE, THE OSGi ALLIANCE DOES NOT GRANT, EITHER EXPRESSLY OR IMPLIEDLY, A LICENSE TO ANY INTELLECTUAL PROPERTY IT, OR ANY THIRD PARTIES, OWN OR CONTROL. Title to the copyright in the Distribution will at all times remain with the OSGi Alliance. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted therein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

THE DISTRIBUTION IS PROVIDED "AS IS," AND THE OSGi ALLIANCE (INCLUDING ANY THIRD PARTIES THAT HAVE CONTRIBUTED TO THE DISTRIBUTION) MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DISTRIBUTION ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.
NEITHER THE OSGi ALLIANCE NOR ANY THIRD PARTY WILL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THE DISTRIBUTION.

Implementation of certain elements of this Distribution may be subject to third party intellectual property rights, including without limitation, patent rights (such a third party may or may not be a member of the OSGi Alliance). The OSGi Alliance is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

The Distribution is a draft. As a result, the final product may change substantially by the time of final publication, and you are cautioned against relying on the content of this Distribution. You are encouraged to update any implementation of the Distribution if and when such Distribution becomes a final specification.

The OSGi Alliance is willing to receive input, suggestions and other feedback (“Feedback”) on the Distribution. By providing such Feedback to the OSGi Alliance, you grant to the OSGi Alliance and all its Members a non-exclusive, non-transferable,

worldwide, perpetual, irrevocable, royalty-free copyright license to copy, publish, license, modify, sublicense or otherwise distribute and exploit your Feedback for any purpose. Likewise, if incorporation of your Feedback would cause an implementation of the Distribution, including as it may be modified, amended, or published at any point in the future ("Future Specification"), to necessarily infringe a patent or patent application that you own or control, you hereby commit to grant to all implementers of such Distribution or Future Specification an irrevocable, worldwide, sublicenseable, royalty free license under such patent or patent application to make, have made, use, sell, offer for sale, import and export products or services that implement such Distribution or Future Specification. You warrant that (a) to the best of your knowledge you have the right to provide this Feedback, and if you are providing Feedback on behalf of a company, you have the rights to provide Feedback on behalf of your company; (b) the Feedback is not confidential to you and does not violate the copyright or trade secret interests of another; and (c) to the best of your knowledge, use of the Feedback would not cause an implementation of the Distribution or a Future Specification to necessarily infringe any third-party patent or patent application known to you. You also acknowledge that the OSGi Alliance is not required to incorporate your Feedback into any version of the Distribution or a Future Specification.

I HEREBY ACKNOWLEDGE AND AGREE TO THE TERMS AND CONDITIONS DELINEATED ABOVE.

0.2 Trademarks

OSGi™ is a trademark, registered trademark, or service mark of the OSGi Alliance in the US and other countries. Java is a trademark, registered trademark, or service mark of Oracle Corporation in the US and other countries. All other trademarks, registered trademarks, or service marks used in this document are the property of their respective owners and are hereby recognized.

0.3 Feedback

This document can be downloaded from the OSGi Alliance design repository at https://github.com/osgi/design
The public can provide feedback about this document by opening a bug at https://www.osgi.org/bugzilla/.

0.4 Table of Contents

0.5 Terminology and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in 10.1.

`Source code is shown in this typeface.`

0.6 Revision History

The last named individual in this history is currently responsible for this document.

Revision	Date	Comments
Initial	*SEP 15 2017*	Initial Contribution. Hiroyuki Maeomichi, NTT, maeomichi.hiroyuki@lab.ntt.co.jp
0.0.1	*SEP 21 2017*	Updated alternatives, some figures, added description on validator. Hiroyuki Maeomichi, NTT, maeomichi.hiroyuki@lab.ntt.co.jp

Revision	Date	Comments
0.0.2	*April 17 2018*	Update based on discussion in Washington meeting. Hiroyuki Maeomichi, NTT, maeomichi.hiroyuki@lab.ntt.co.jp
0.0.3	*June 22 2018*	Add new fields and class reflecting R3 draft of oneM2M: Added fields in RequestPrimitiveDTO, ResponsePrimitiveDTO, and FilterCriteriaDTO, and ReleaseVersion enum. Organize DTOs: Added AttributeDTO, LocalIdTokenIdAssigmentDTO, and DasInfoDTO and remove DynAuthLocalIdAssignmentsDTO and DynAuthReqInfoDTO Introduce OperationIF interface as a super interface of ProtocolBinding interface and CSE interface for enabling concise application code. This replaces former simple.Client. Organize Introspection interfaces with less methods. They are moved to dedicated package.
0.0.4	*June 25 2018*	Add section 'Mapping Rules for Generic DTO' Update Javadoc with more explanations. (moved old classes to org.osgi.service.onem2m.old package for preparing deletion.) Add description to Security Consideration section. Add oneM2M R3 specs and XSD to references.
0.0.5	*Jun 27 2018*	Modified after discussion in Washington DC F2F. Restructure service interfaces; now 2 interface is remaining. For receiving notification, dedicated interface is prepared. Remove Introspection interfaces. Reduce service properties by removing ones for informative purpose. Add example flow to control devices. Modify security consideration in section 9.2.
0.06	*Jun 29 2018*	Add examples, with code snipets. Add 'Data Modification in Protocol Binding' section. Add discovery() method with additional parameter.
0.07	*Sep 17 2018*	Remove all service properties. Add some reasons to considered alternative section.
0.0.8	*May 22 2019*	Add configuration chapter.(5.4) Fix API parameter.

Revision	Date	Comments
0.1	*Feb 04 2020*	Remove configuration chapter (introduce 0.0.8) Fix Figure, typo. Add GenericDTO Remove Instruction(in red text)

1 Introduction

~~Introduce the RFC. Discuss the origins and status of the RFC and list any open items to do.~~

oneM2M is standard organization and specifies middleware for Internet of Things (IoT), called Common Services Entities (CSE). Applications can access CSE's functionality with RESTful operations, which are Create, Retrieve, Update, Delete and Notify. TS-0001 [2] defines more than 40 resource types to expose CSE's functionalities. oneM2M allows variety of communication methods, combination of 4 protocol bindings (HTTP, MQTT, CoAP, Websocket) and 3 serializations (XML, JSON, CBOR).

This RFP discuss the way to provide high level API (namely service layer API) for oneM2M RESTful operations hiding the difference of variety of communication methods.

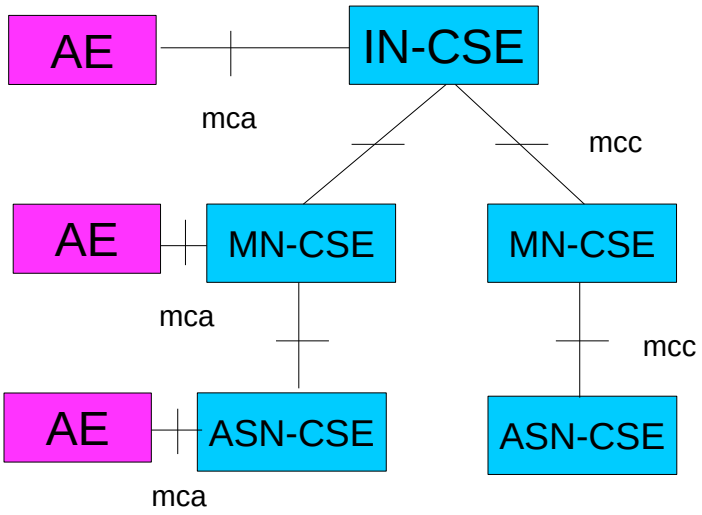
2 Application Domain

~~*This section should be copied from the appropriate RFP(s). It is repeated here so it can be extended while the RFC authors learn more subtle details.*~~

2.1 IoT Application configuration using oneM2M

oneM2M's middleware, called CSE can be deployed in different locations and they are connected each other forming tree topology. Depending on deployed location, CSEs are categorized to 3 types, IN-CSE, MN-CSE and

ASN-CSE. IN-CSE is located top of tree, ASN-CSE is located at leaf and MN-CSE is located and MN-CSE is located on middle.

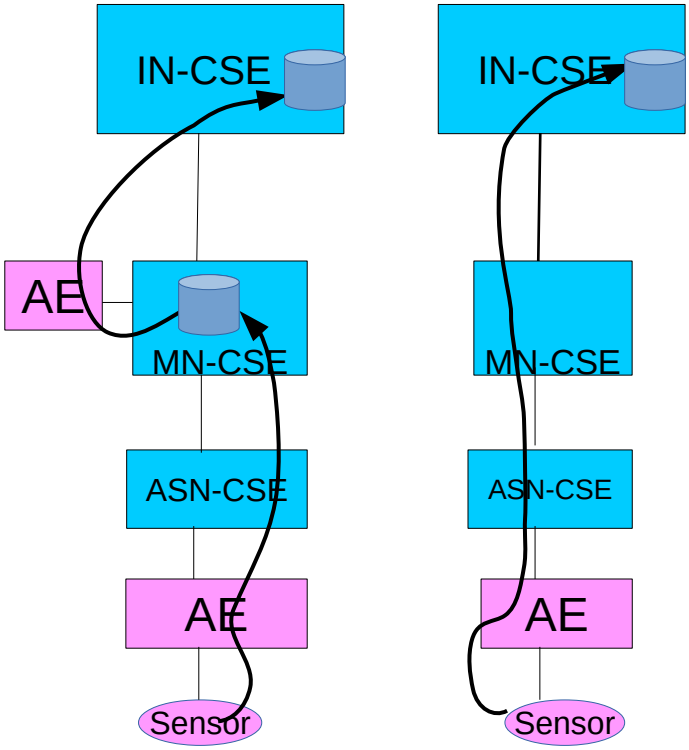


oneM2M's application, called Application Entity (AE) connects to one of CSEs. After AE connecting to the CSE, AE can access to all of CSEs, by retargeting function (similar to routing) of CSEs.

AE accesses to CSE's functionality through RESTful API, which consists of Create, Retrieve, Update, Delete and Notify in targeting more than 40 types of resources. For examples, typical resources are <*contentInstance*> that expresses IoT data and <*container*> that holds set of <*contentInstance*>s. AE can create or retrieve the <contentInstance> on any CSE by the retargeting functionality, as far as permission is allowed. Interface between CSEs is called *mcc* and interface between CSE and AE is called *mca,* both interfaces have almost same interface.

It is possible to develop variety types of distributed applications using the architecture. For example for IoT data aggregation applications, it is possible to develop gradual aggregation type or direct aggregation type. In gradual aggregation type, AE connected to ASN-CSE creates <*conentInstance*>s in ASN-CSE, and intermediate applications calculate statistics and put the result on IN-CSE as a <*contentInstance*>, while, in direct aggregation type, AE connected to ASN-CSE creates <*contentInstance*>s in IN-CSE directly.

Under CSE layer, oneM2M specifies NSE(Network Services Entity), but this RFC doesn't cover the NSE layer.



2.2 Communication methods used in oneM2M

oneM2M allows variety of communication methods, combination of 4 protocol bindings (HTTP, MQTT, CoAP, Websocket) and 3 serializations (XML, JSON, CBOR). It might be added in future. oneM2M specifies specification in different level.

Firstly TS-0001[2] specifies high level resource definitions, it defines more than 40 resource types, such as <contentInstance> for storing IoT data, <timeSeriesInstance> for periodic sensor measurement with leap detection mechanism.

Secondly TS-0004[3] specifies procedures and serializations in independent manner from protocol bindings. Resource type and protocol data unit are defined using XSD for XML serialization. Mapping between XML and other serializations are also specified.

Thirdly TS-0008, TS-0009, TS-0010, TS-0020 specify protocol specific details for CoAP, HTTP, MQTT and Web Socket respectively.

2.3 Long name and short name

oneM2M introduced two types of notation, called long name and short name for resource types, attribute and so on. Long name is human friendly string and specifications mainly use this notation, while short name is short string consist of typically 2 or 3 characters (but not limited and sometimes longer) and communication protocol use this notation. In most cases, the initial characters of long name are assigned as short name, for examples, ct for CreationTime and at for AnnounceTo.

3 Problem Description

~~*This section should be copied from the appropriate RFP(s). It is repeated here so it can be extended while the RFC authors learn more subtle details.*~~

oneM2M specifies protocol based interface, but doesn't specify programing level API. As previously mentioned oneM2M allows variety of communication methods which are the combinations of 4 protocol bindings (HTTP, MQTT, CoAP, Websocket) and 3 serializations (XML, JSON, CBOR).

First problem is application portability. Without standardized API, application program tends to depend on the communication method initially intend to use and it will became hard to run another environment in which uses another communication method. (For example, an application designed for XML/HTTP, tend to run on environment use JSON/Websocket)

Second problem is the latency of the communication between CSE and application. Even if CSE and application is located in the same box, current oneM2M specifications define methods through protocols which requires serialization/deserialization of data, context-switch of applications, validation of incoming data and resulted in large latency compared to the situation both CSE and Application resides in the same Java VM and communicate with Java interfaces. Large latency reduces applicable area of oneM2M based solution.

Third problem is the complexity of handling of long name and short name. Even if short name is defined by trying to use initial characters, it is not straight forward to translate them in head.

4 Requirements

~~*This section should be copied from the appropriate RFP(s)*~~

- R0010 – The solution MUST provide means to access outer CSE from application.
- R0011 – The solution MUST provide means to access outer CSE from client CSE.
- R0012 – The solution MUST provide means to select a communication method for application.
- R0013 – The solution MUST provide means to select a communication method for client CSE.
- R0020 – The solution MUST provide means for CSE to accept requests form outer CSE.
- R0020 – The solution MUST provide means for CSE to accept requests form outer application.

- R0030 – The solution MUST provide means to communicate through Java interface between CSE and application that are located in the same OSGi framework.
- R0040 – The solution SHOULD hide differences of communication methods, which are combinations of 4 protocol bindings and 3 serializations (XML, JSON, CBOR).
- R0050 – The solution SHOULD provide developer friendly way for handling short names.
- R0060 – The solution MUST provide asynchronous interface using 'call by value', such as DTO.

5 Technical Solution

~~*First give an architectural overview of the solution so the reader is gently introduced in the solution (Javadoc is not considered gently). What are the different modules? How do the modules relate? How do they interact? Where do they come from? This section should contain a class diagram. Then describe the different modules in detail. This should contain descriptions, Java code, UML class diagrams, state diagrams and interaction diagrams. This section should be sufficient to implement the solution assuming a skilled person.*~~

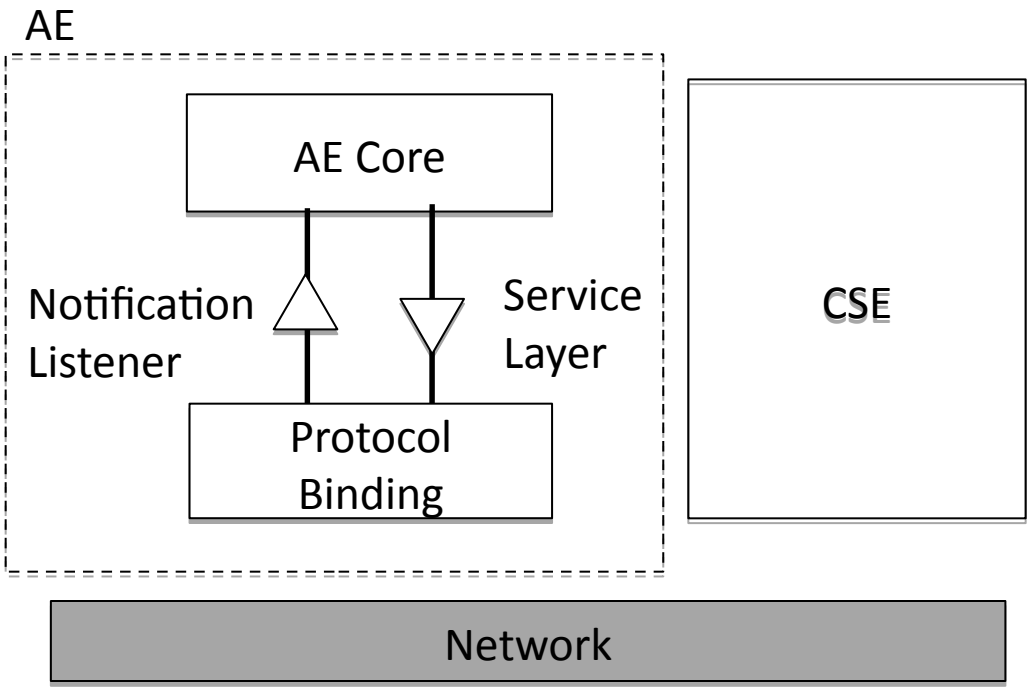
~~*Strictly use the terminology a defined in the Problem Context.*~~

~~*On each level, list the limitations of the solutions and any rationales for design decisions. Almost every decision is a trade off so explain what those trade offs are and why a specific trade off is made.*~~

~~*Address what security mechanisms are implemented and how they should be used.*~~

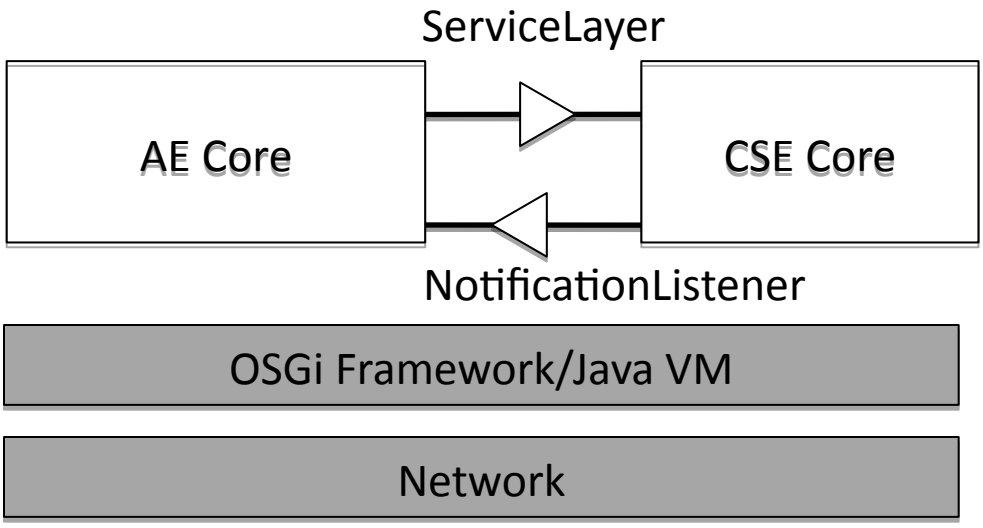
5.1 Overview for the solution

Protocol binding service is introduced to handle different protocols and serializations. oneM2M application uses the protocol binding service through Service Layer Interface to communicate CSE. The interface is protocol and serialization agnostic interface; it has no protocol and serialization specific methods, parameters, so that application can communicate to CSE without knowing which protocol is actually used.

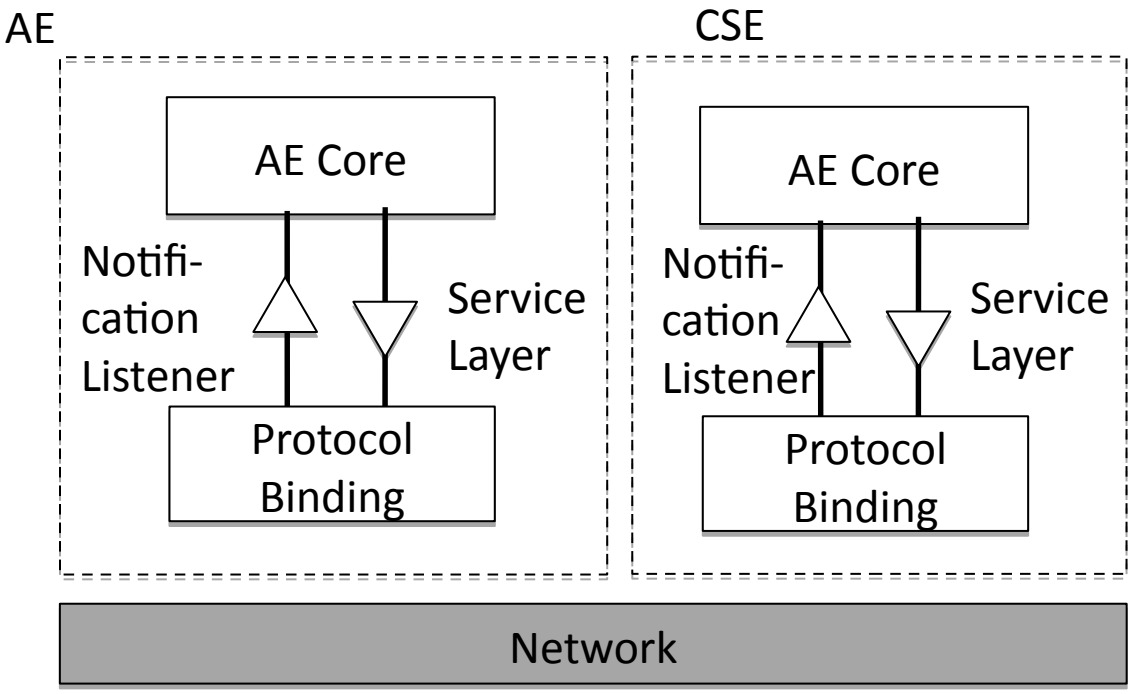


In figure, the term of Core is introduced for AE Core and CSE Core. This is for specifying parts which does not contain ProtocolBinding of AE and CSE.

Another use case is that the AE Core and the CSE core are located on the same OSGi Framework. In this use case, the AE Core and the CSE core communicate directly with ServiceLayer API, without inter-mediating ProtocolBinding Services. Following figure depicts overall configuration. Though this type of communication is not clearly defined in oneM2M specification, communicating directly without serializing data between AE and CSE allows shorter latency and less computational resources.



Following figure (right hand side) shows potential implementation of CSE, which are consisted fo CSE core and Protocol Binding Service, as symmetric to AE side. APIs defined in this RFC is consistent with the usage, but this RFC does not mandate that usage and it is left to implementor's choice.



5.2 Service Layer Interfaces

Service Layer Interface is for allowing AE to send request and get response.

request() method allows very raw data type access and it enables all possible message exchanges among oneM2M entities,

```
Promise<ResponseDTO> request(RequestDTO request);
```

Meanwhile, it can be redundant to application developers, because they need to write composition of requestPrimitive and decomposition of responsePrimitive. This interface is provided for application developer allowing less application codes. It provides methods with higher level of abstraction; operation level of resource such as create, retrieve, update, delete and so on. They do not cover all of possible message exchange but do typical ones.

Note: If this RFC doesn't provide these methods, developers likely to create similar ones in their own (various) way.

```
package org.osgi.service.onem2m.servicelayer;
```

```
import org.osgi.onem2m.dto.RequestDTO;
import org.osgi.onem2m.dto.ResponseDTO;
import org.osgi.util.promise.Promise;

/**
 * Service Layer Interface, which locates between AE and Protocol Binding Service.
 */
public interface ServiceLayer {
	/**
	 * send a request.
	 *
	 * @param request request
	 * @return promise for ResponseDTO.
	 */
	Promise<ResponseDTO> request(RequestDTO request);

	/**
	 * create resource
	 *
	 * @param uri URI for parent resource
	 * @param resource resource data
	 * @return Promise of created resource
	 */
	public Promise<ResourceDTO> create(String uri, ResourceDTO resource);

	/**
	 * retrieve resource
	 *
	 * @param uri URI for retrieving resource
	 * @return retrieved resource data
	 */
	public Promise<ResourceDTO> retrieve(String uri);

	/**
	 * retrieve subset of attributes.
	 *
	 * @param uri URI for retrieving resource
	 * @param targetNames attribute names for retrival
	 * @return retrieved resource data
	 */
	public Promise<ResourceDTO> retrieve(String uri, List<String>
targetAttributes);

	/**
	 * update resource
	 *
	 * @param uri URI for updating resource
	 * @param resource data resource
```

```
     * @return updated resource
     */
    public Promise<ResourceDTO> update(String uri, ResourceDTO resource);

    /**
     * delete resource
     *
     * @param uri target URI for deleting resource
     */
    public Promise<Boolean> delete(String uri);

    /**
     * find resources
     *
     * @param uri URI for top of search
     * @param fc filter criteria
     * @return list of URIs matching the condition specified in fc
     */
    public Promise<List<String>> discovery(String uri, FilterCriteriaDTO fc);

         /**
     * find resources
     *
     * @param uri URI for top of search
     * @param fc filter criteria
     * @param drt Discovery Result Type (structured/unstructured)
     * @return list of URIs matching the condition specified in fc
     */
    public Promise<List<String>> discovery(String uri, FilterCriteriaDTO
fc, RequestPrimitiveDTO.DiscoveryResultType drt);

    /**
     * send notification
     *
     * @param notification
     */
    public Promise<Boolean> notify(String uri, NotificationDTO notification );

}
```

5.3 Service Property for Interfaces

No service properties are defined for ServiceLayer interface and NotificagtionListener interface.

5.4 Service Binding

Proper instance of ServiceLayer~~Protocol Binding~~ Service must be bound to proper AE Core. Implementation of ServiceLayer ~~be may registered as~~ should created by ServiceFactory ~~on the service registry and implementation should create~~to provide a proper service instance depending on calling AE Core.

~~Managed Service Factory shall be used for creating and configuring service objects for ServiceLayer. "org.osgi.service.onem2m.servicelayer.pid" is used as PID. Following table summarizes key and value information for configuration object.~~

5.5 Data Modification in ServiceLayer Service~~Protocol Binding~~

Usually ServiceLayer service doesn't change any logical information on passing data, but a~~A~~s an ~~only~~ exception ~~of the behavior of Protocol Binding entity (implements~~ ServiceLayer service modifies the passing data by adding pointOfAccess information on following case.~~, in following condition,), it MUST add pointOfAccess attribute using configured value~~

1. create() for <AE> resource

2. update() for <AE> only when ~~'~~pointOfAccess'~~ ~~is specified.

3. request() when content is equivalent to above.

This is because AE Core entity doesn't know the pointOfAccess information and processing show above simply solve the problem.

5.6 Example: Registration

This section explains how application registers to its hosting CSE. In order to interact with the oneM2M system the bundle obtains a reference to the `ServiceLayer` service from the service registry

```
@Component
public class MyLightSwitchComponent {
@Reference
ServiceLayer client
}
```

After getting client, it starts registration by creating <AE> resource.<AE> resource creation requires App-ID, AE-ID, requestReachability attributes. In the following code, "C" is passed for AE-ID, this means asking CSE for assigning the value. Assigned value is is included in returned resource

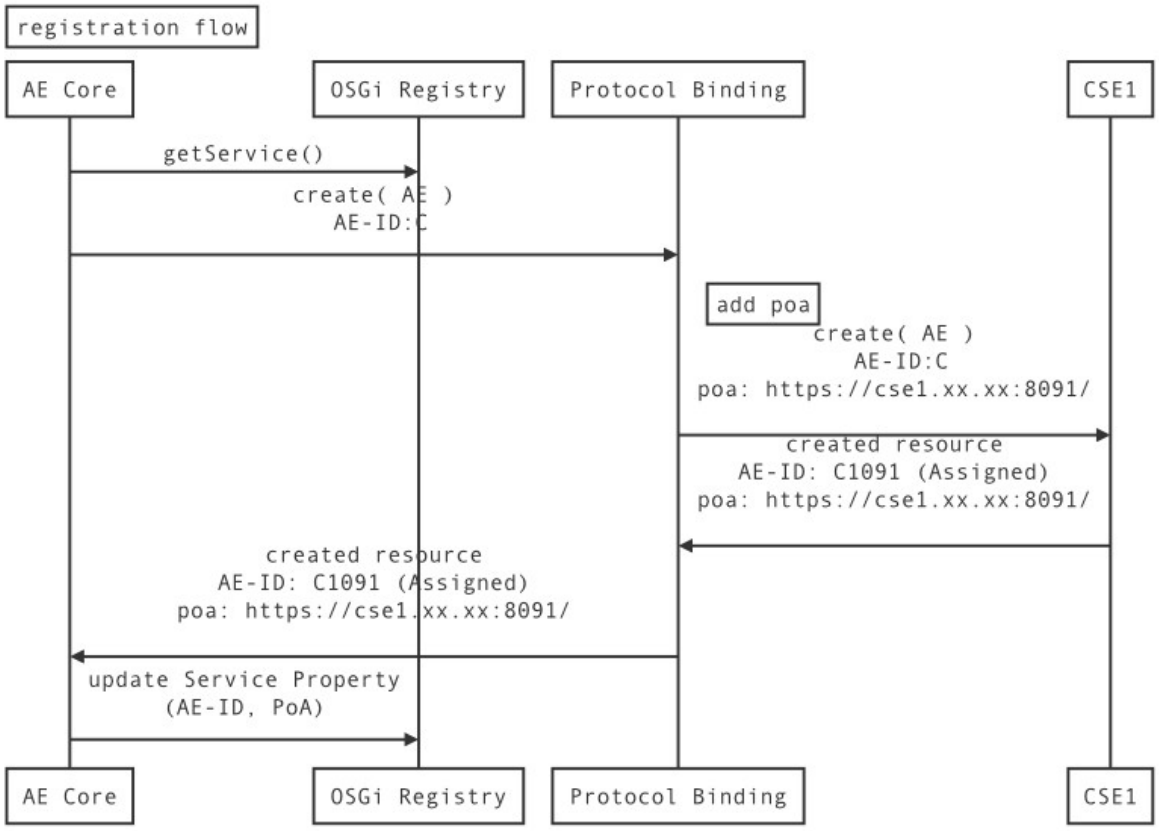
The information of pointOfAccess is kept in ~~Protocol Binding entity~~ServiceLayer Service, it is assigned by ~~Protocol Binding entity~~ServiceLayer Service. before sending the request message to network. The value is also returned in response.

After success response of <AE> resource, it registers NotificationListener with returned AE-ID and pointOfAccess as the service property

```
@Component
public class RegistrationExample implements NotificationListener {
	@Reference
	ServiceLayer client;

	BundleContext context;
	 @Activate
	 void start(BundleContext context) {
		this.context = context;

		// create AE: This means registration.
		ResourceDTO dto = new ResourceDTO();
		dto.resourceType = ResourceType.AE.getValue();
		dto.attribute.put("App-ID", "01.com.company.lightApp1<");
		dto.attribute.put("AE-ID", "C");
		dto.attribute.put("requestReachability", Boolean.TRUE);
		final Promise<ResourceDTO> ret = client.create("/CSE1/-", dto);

		ret.onResolve(new Runnable() {
			public void run() {
				 ResourceDTO ae = null;
				try {
					ae = ret.getValue();
				} catch (Exception e) {
					e.printStackTrace();
					return;
				}
				 Dictionary prop = new Properties();
				 prop.put("AE-ID",ae.attribute.get("AE-ID"));
				 prop.put("PoA",ae.attribute.get("pointOfAccess");
				 context.registerService(new String[] {"org.osgi.service.onem2m.
NotificationListener"}, this, prop);
			}
```

```
            });
     }

    @Override
    public void notified(RequestPrimitiveDTO request) {
          //....
    }
```

Following figure shows sequence diagram of the registration flow.



5.7 Example: Turning Light ON

This section explains how application turns on the lights located on remote site. There are mainly two ways to represent devices in oneM2M.

First way uses <flecContainer> resource type for representing device, its status, and so on. The resource type is introduced in oneM2M release 2 and it allows having custom attributes in it. Based on the resource, variety of the

data model for devices, especially of home domain, are specified in TS-0023. With this way application can use standardized data model and operate device status in commonly used manner.

Second way uses <container> and <contentInstance> resource types for representing device and its status. These resource types are introduced very beginning of oneM2M and this approach is well explained in developer guide (TR-0025 [TODO ref. WEB]) Meanwhile, the resource type is not primarily designed for this purpose, but for storing data, so that how to operate device could be different from usual manner. To change state, new <contentInstance> is created with new status in its content attribute. <container> usually has multiple <contentInstance>s underneath and the latest one is supposed to be the latest status.

5.7.1 Example (Using <flexContainer>

In this example a simple bundle wishes to swtich light devices on when the bundle is started, and switch them off again when the bundle is stopped. Here it is assumed that registration process described in the previous section is done.

1. When the bundle has obtained a reference to the ServiceLayer then it can use the `discover` method to find all of the lightbulbs in the system

```
public class MyLightSwitchComponent {
@Reference
ServiceLayer client;

Promise<List<String>> discoveredLightbulbs;

@Activate
void start() {
  discoveredLightbulbs = findLightBulbs();
}

private Promise<List<String>> findLightBulbs() {
    String baseURI = "/homegateway/-/"; // - means CSE Base ( kaid of top directory )

    FilterCriteriaDTO filter = new FilterCriteriaDTO();
    filter.resourceType = Collections.singletonList(
      ResourceType.flexContainer.getValue());
    AttributeDTO attr = new AttributeDTO();
    attr.name = "contentDefinition";
    attr.value = "org.onem2m.home.device.light";
    filter.attribute = Collections.singletonList(attr);
    filter.filterOperation = FilterCriteriaDTO.FilterOperation.AND;
```

```
    return client.discovery(baseURI, filter);
}
}
```

2. Once the lightbulbs are discovered then the bundle can switch on the bulbs by creating a content instance using the `update` method

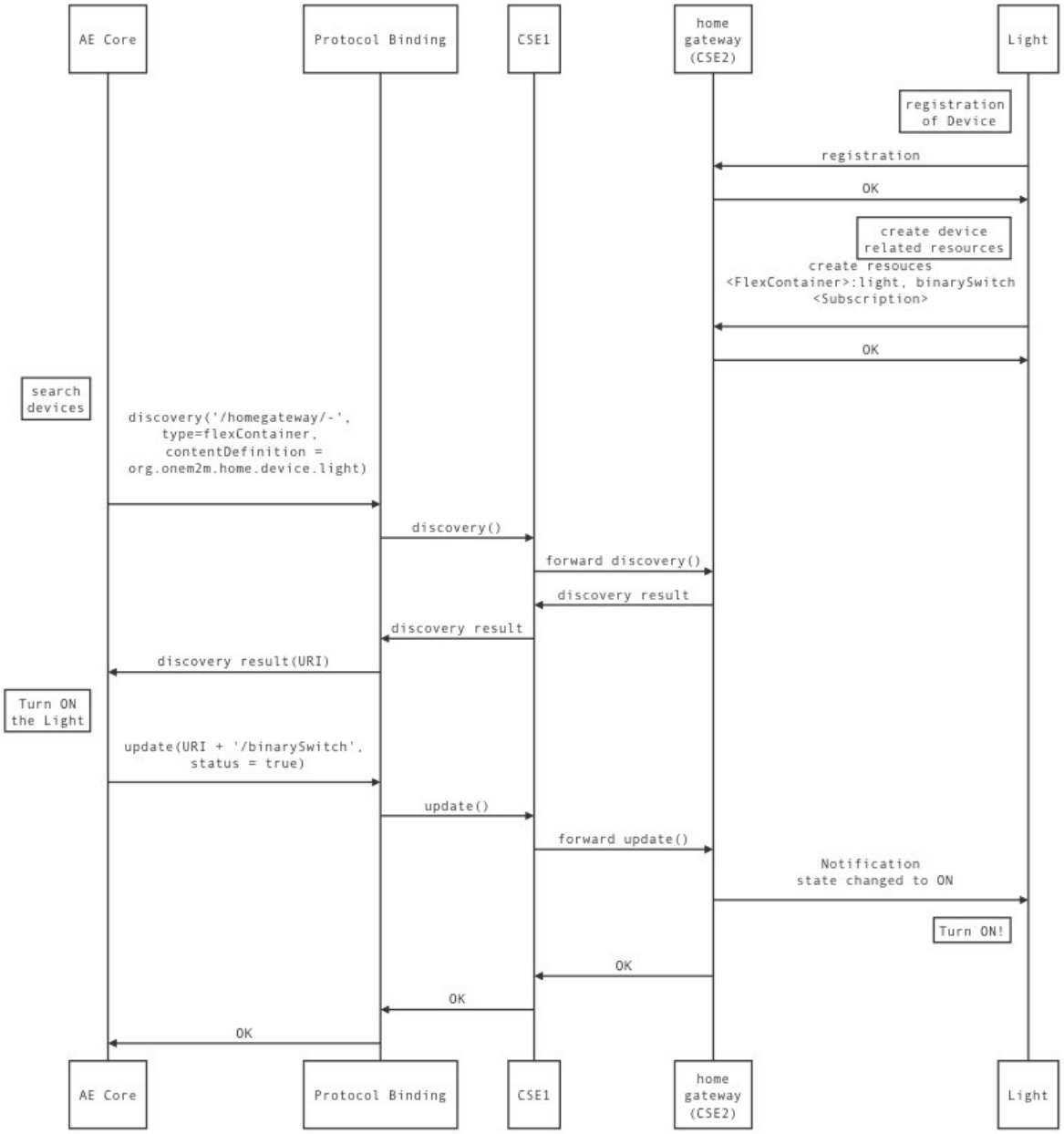
```
public class MyLightSwitchComponent {
@Reference
ServiceLayer client;
Promise<List<ResourceDTO> allTurnedOn;

@Activate
void start() {
  allTurnedOn = findLightBulbs().
        flatMap(l -> Promises.all(
            l.stream()
                .map(this::turnOn)
                .collect(toList())));
}

private Promise<ResourceDTO> turnOn(String bulbUri) {
  ResourceDTO dto = new ResourceDTO();
  dto.attribute.put("powerStatus", Boolean.TRUE);
  return client.update(bulbUri + "/binarySwitch", dto );
}
}
```

3. Finally, the bulbs can be turned off again when the bundle is stopped.

```
public class MyLightSwitchComponent {
@Reference
ServiceLayer client;
Promise<List<ResourceDTO> allTurnedOn;

@Deactivate
void stop() {
    findLightBulbs().flatMap(l -> Promises.all(
        l.stream()
```

```
            .map(this::turnOff)
        .collect(toList())));
}
}
```

Following figure shows the example showing how application turn on the light device on the remote.



5.7.2 Example (Using <container> and <contentInstance>

In this example, device is expressed as <container> resource type in remote CSE (Home Gateway). Discovery is changed as follows. Here it assumes that all <container>s representing lightbulb have label of “lightBulb”.

```
private Promise<List<String>> findLightBulbs() {
        String baseURI = "/homegateway/-/"; // - means CSEBase ( kind of top directory )

        FilterCriteriaDTO filter = new FilterCriteriaDTO();
        filter.resourceType = Collections.singletonList(
            ResourceType.container.getValue());
        filter.labels = Collections.singletonList("lightBulb");
        filter.filterOperation = FilterCriteriaDTO.FilterOperation.AND;
        return client.discovery(baseURI, filter);
}
```

Light can be controlled by creating new <contentInstance> resource as follows.

```
   private Promise<ResourceDTO> turnOn(String bulbUri) {
        ResourceDTO dto = new ResourceDTO();
        dto.resourceType = ResourceType.container.getValue();
        dto.attribute.put("content", "ON");
        return client.create(bulbUri , dto );
    }
```

6 Data Transfer Objects

~~*RFC 185 defines Data Transfer Objects as a generic means for management solutions to interact with runtime entities in an OSGi Framework. DTOs provides a common, easily serializable representation of the technology.*~~

~~*For all new functionality added to the OSGi Framework the question should be asked: would this feature benefit from a DTO? The expectation is that in most cases it would.*~~

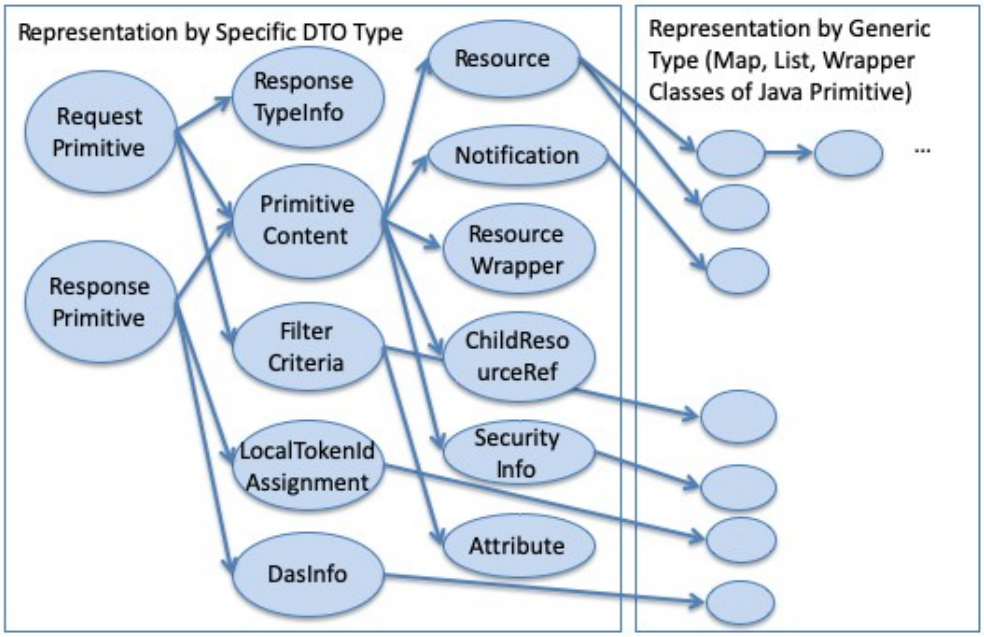
~~*The DTOs for the design in this RFC should be described here and if there are no DTOs being defined an explanation should be given explaining why this is not applicable in this case.*~~

~~*This section is optional and could also be provided in a separate RFC.*~~

6.1 Design Policy of DTOs

Data transfer Object was chosen as data object passing by the interfaces, following OSGi's design convention. Though data structure near root structures are designed specific DTO, deeper data types are to be stored as generic types, such as Map, List and Wrapper classes of Java Primitives. Following figure depict reference relationship among object. (See also the considered Alternatives)

In the class definition, some data types are shown as Object, but the assigned value shall be a type that is allowed for OSGi DTO.



In figure, followed 'DTO' is committed.~~ommited.~~

6.2 RequestPrimitiveDTO

RequestPrimitiveDTO holds a Request Information used for oneM2M communication.

```
package org.osgi.service.onem2m.dto;

import java.util.*;

public class RequestPrimitiveDTO extends org.osgi.dto.DTO {
	@javax.xml.bind.annotation.XmlElement(required = true)
	public Operation operation;
	@javax.xml.bind.annotation.XmlElement(required = true)
	public String to;
```

```
	public String from;
	@javax.xml.bind.annotation.XmlElement(required = true)
	public String requestIdentifier;
	@javax.xml.bind.annotation.XmlElement(required = false)
	public Integer resourceType;
	public PrimitiveContentDTO content;
	public List<String> roleIDs;

	public String originatingTimestamp;
	public String requestExpirationTimestamp;
	public String resultExpirationTimestamp;
	public String operationExecutionTime;

	public ResponseTypeInfoDTO responseType;
	public String resultPersistence;
	@javax.xml.bind.annotation.XmlElement(required = false)
	public ResultContent resultContent;
	public String eventCategory;
	@javax.xml.bind.annotation.XmlElement(required = false)
	public Boolean deliveryAggregation;
	public String groupRequestIdentifier;
	public FilterCriteriaDTO filterCriteria;
	@javax.xml.bind.annotation.XmlElement(required = false)
	public DiscoveryResultType discoveryResultType;
	public String tokens;
	public List<String> tokenIDs;
	public List<String> localTokenIDs;
	@javax.xml.bind.annotation.XmlElement(required = false)
	public Boolean tokenReqIndicator;

	// Added at R3.0
	public List<String> groupRequestTargetMembers;
	public Boolean authSignatureIndicator;
	public List<String> authSignature;
	public Boolean authRelationshipIndicator;
	public Boolean semanticQueryIndicator;
	public ReleaseVersion releaseVersion;
	public String verndorInformation;

	public static enum DiscoveryResultType {
		structured(1), unstructured(2);
		// omitted
	}

	public static enum ResultContent {
		nothing(1), attributes(2), hierarchicalAddress(3),
		hierarchicalAddressAndAttributes(4),
```

```
attributesAndChildResources(5),
attributesAndChildResourceReferences(6),
childResourceReferences(7), originalResource(8), childResources(9);

            // omitted
      }

      public static enum Operation {
            Create(1), Retrieve(2), Update(3), Delete(4), Notify(5);
            // omitted
      }

}
```

6.3 ResponsePrimitiveDTO

ResponsePrimitiveDTO holds a Response Information used for oneM2M communication.

```
package org.osgi.service.onem2m.dto;
import java.util.*;

public class ResponsePrimitiveDTO extends org.osgi.dto.DTO{
      @javax.xml.bind.annotation.XmlElement( required = true)
      public Integer responseStatusCode;
      @javax.xml.bind.annotation.XmlElement( required = true)
      public String requestIdentifier;
      public PrimitiveContentDTO content;
      public String to;
      public String from;
      public String originatingTimestamp;
      public String resultExpirationTimestamp;
      public String eventCategory;
      @javax.xml.bind.annotation.XmlElement( required = false)
      public ContentStatus contentStatus;
      @javax.xml.bind.annotation.XmlElement( required = false)
      public Integer contentOffset;
      public List<LocalTokenIdAssignmentDTO>_-
assignedTokenIdentifiers;//Map<String,Object>
      public List<DasInfoDTO> tokenReqInfo;//DynAuthTokenReqInfoDTO

      // Added R3.0
      public Boolean AuthSignatureReqInfo;
      public ReleaseVersion releaseVersionIndicator;
      public String vendorInformation;
```

```
	public static enum ContentStatus{
		PARTIAL_CONTENT, // 1
		FULL_CONTENT;  //2

	}
}
```

6.4 ResponseTypeInfoDTO

```
package org.osgi.service.onem2m.dto;

import java.util.*;

public class ResponseTypeInfoDTO extends org.osgi.dto.DTO {
	@javax.xml.bind.annotation.XmlElement(required = true)
	public java.lang.Integer responseTypeValue;
	@javax.xml.bind.annotation.XmlElement(required = true)
	public List<java.lang.String> notificationURI;

	public static enum ResponseType {
		nonBlockingRequestSynch(1), nonBlockingRequestAsynch(2),
blockingRequest(3), flexBlocking(4);
		// omitted.
	}
}
```

6.5 FilterCriteriaDTO

```
package org.osgi.service.onem2m.dto;
import java.util.*;

public class FilterCriteriaDTO extends org.osgi.dto.DTO{
	public String createdBefore;
	public String createdAfter;
	public String modifiedSince;
	public String unmodifiedSince;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer stateTagSmaller;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer stateTagBigger;
	public String expireBefore;
	public String expireAfter;
```

```
	public List<String> labels;
	public List<Integer> resourceType;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer sizeAbove;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer sizeBelow;
	public List<String> contentType;
	public AttributeDTO attribute;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public FilterUsage filterUsage;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer limit;
	public String semanticsFilter;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public FilterOperation filterOperation;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer contentFilterSyntax;
	public String contentFilterQuery;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer level;
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Integer offset;

	// added in R3
	public List<String> childLabels;
	public List<String> parentLabels;
	public String labelsQuery;
	public Integer childResourceType;
	public Integer parentResourceType;
	public AttributeDTO childAttribute;
	public AttributeDTO parentAttribute;
	public String applyRelativePath;


	public static enum FilterOperation {
		AND(1), OR(2);
		// omitted...
	}

	public static enum FilterUsage {
		DiscoveryCriteria(1), ConditionalRetrival(2), IPEOndemandDiscovery(3);
		// omitted...

	}
}
```

6.6 ResourceDTO

```
package org.osgi.service.onem2m.dto;
import java.util.*;

public class ResourceDTO extends org.osgi.dto.DTO{

	// Universal Attribute, which can be held by all resources.
	@javax.xml.bind.annotation.XmlElement( required  = true)
	public Integer resourceType;
	@javax.xml.bind.annotation.XmlElement( required  = true)
	public String resourceID;
	@javax.xml.bind.annotation.XmlElement( required  = true)
	public String parentID;
	@javax.xml.bind.annotation.XmlElement( required  = true)
	public String creationTime;
	@javax.xml.bind.annotation.XmlElement( required  = true)
	public String lastModifiedTime;

	public String resourceName;

	// optional, Universal Attributes
	public List<String> labels;

	/**
	 * Non Universal Attribute.
	 * Value Part must be the types that are allowed for OSGi DTO.
	 */
	public Map<String, Object> attribute;
}
```

6.7 NotificationDTO

NotificationDTO has information of notification.

```
package org.osgi.service.onem2m.dto;
import java.util.*;

public class NotificationDTO extends org.osgi.dto.DTO{
	public Map<String,Object> notificationEvent;//NotificationEventDTO
	@javax.xml.bind.annotation.XmlElement( required  = false)
	public Boolean verificationRequest;
	@javax.xml.bind.annotation.XmlElement( required  = false)
```

```
        public Boolean subscriptionDeletion;
        public String subscriptionReference;
        public String creator;
        public String notificationForwardingURI;
        @javax.xml.bind.annotation.XmlElement( required = false)
        public Map<String,Object> ipeDiscoveryRequest;//IPEDiscoveryRequestDTO
}
```

6.8 Other DTOs

There are some other DTOs, please refer Javadoc section for them.

6.9 Mapping Rules for Generic DTO

Following table summarizes mapping rule between oneM2M data types and Generic types used in DTOs. There are two types of XSD are defined in oneM2M, which are longname version and shortname version. The longname version should be referred.~~refered.~~

oneM2M Types (XML Schema)	Type of OSGi DTO	
Basic Types of XML Schema	Wrapper Object of Java primitive	For example: xs:integer, xs:float
xs:anyURI, m2m:ID,	String	
m2m:timestamp	String	YYYYMMDDThhmmss,ssssss
m2m:absRelTimestamp	String	Union of m2m:timestamp and xs:long. This is exception of union rule above. Distinction is done by existence of 'T'
xs:sequence (as complexType)	Map	Name of element is used for key of map.
xs:list, xs:sequence (as list)	List	
xs:union	Map	Base attribute of restriction tag is used for key of map. Only one key is allowed. See Example of missingDataList:

Following XML is an example of missingData.

```
<xs:simpleType name="missingDataList">
      <xs:union>
        <simpleType>
            <restriction base='m2m:listOfTimeStamp' />
```

```
        </simpleType>
        <simpleType>
            <restriction base='m2m:listOfRelTimeStamp' />
        </simpleType>
      </xs:union>
    </xs:simpleType>
```

7 Javadoc

Please include Javadoc of any new APIs here, once the design has matured. Instructions on how to export Javadoc for inclusion in the RFC can be found here: https://www.osgi.org/members/RFC/Javadoc

OSGi Javadoc

20/02/05 16:13

Package Summary		Page
org.osgi.service.onem2m		32
org.osgi.service.onem2m.dto		38

Package org.osgi.service.onem2m

Interface Summary		Page
NotificationListener	Primary Interface for an oneM2M entity to send request and get response to/from other oneM2M entity.	33
ServiceLayer	Primary Interface for an oneM2M entity to send request and get response to/from other oneM2M entity.	35

Exception Summary		Page
OneM2MException	General Exception for oneM2M.	34

Interface NotificationListener

org.osgi.service.onem2m

`public interface` **`NotificationListener`**

Primary Interface for an oneM2M entity to send request and get response to/from other oneM2M entity.

Method Summary		Page
`void`	**`notified`**`(RequestPrimitiveDTO request)` receive notification.	33

Method Detail

notified

`void` **`notified`**`(RequestPrimitiveDTO request)`

receive notification.

Parameters:

`request` - request primitive

Class OneM2MException

org.osgi.service.onem2m

```
java.lang.Object
  └java.lang.Throwable
    └java.lang.Exception
      └java.io.IOException
        └org.osgi.service.onem2m.OneM2MException
```

All Implemented Interfaces:

Serializable

```
public class OneM2MException
extends IOException
```

General Exception for oneM2M.

Field Summary		Page
	cause Cause of Exception	34
	errorCode Error Code	34

Constructor Summary	Page
OneM2MException()	34

Field Detail

errorCode

```
public errorCode
```

Error Code

~~cause~~

```
public cause
```

Cause of Exception

Constructor Detail

OneM2MException

```
public OneM2MException()
```

Interface ServiceLayer

org.osgi.service.onem2m

`public interface` **`ServiceLayer`**

Primary Interface for an oneM2M entity to send request and get response to/from other oneM2M entity.

Method Summary		Page
org.osgi.util.promise.Promise	**create**(String uri, ResourceDTO resource) create resource	35
org.osgi.util.promise.Promise	**delete**(String uri) delete resource	36
org.osgi.util.promise.Promise	**discovery**(String uri, FilterCriteriaDTO fc) find resources.	36
org.osgi.util.promise.Promise	**discovery**(String uri, FilterCriteriaDTO fc, RequestPrimitiveDTO.DiscoveryResultType drt) find resources	37
org.osgi.util.promise.Promise	**notify**(String uri, NotificationDTO notification) send notification	37
org.osgi.util.promise.Promise	**request**(RequestPrimitiveDTO request) send a request.	35
org.osgi.util.promise.Promise	**retrieve**(String uri) retrieve resource	36
org.osgi.util.promise.Promise	**retrieve**(String uri, List targetAttributes) retrieve subset of attributes.	36
org.osgi.util.promise.Promise	**update**(String uri, ResourceDTO resource) update resource	36

Method Detail

request

`org.osgi.util.promise.Promise` **`request`**`(RequestPrimitiveDTO request)`

send a request.

Parameters:
`request` - request primitive
Returns:
promise of ResponseDTO.

---

create

`org.osgi.util.promise.Promise` **`create`**`(String uri, ResourceDTO resource)`

create resource

Parameters:
`uri` - URI for parent resource

`resource` - resource data

Returns:

Promise of created resource

---

retrieve

```
org.osgi.util.promise.Promise retrieve(String uri)
```

retrieve resource

Parameters:

`uri` - URI for retrieving resource

Returns:

retrieved resource data

---

retrieve

```
org.osgi.util.promise.Promise retrieve(String uri,
                                       List targetAttributes)
```

retrieve subset of attributes.

Parameters:

`uri` - URI for retrieving resource

`targetAttributes` - names of the target attribute

Returns:

retrieved resource data

---

update

```
org.osgi.util.promise.Promise update(String uri,
                                     ResourceDTO resource)
```

update resource

Parameters:

`uri` - URI for updating resource

`resource` - data resource

Returns:

updated resource

---

delete

```
org.osgi.util.promise.Promise delete(String uri)
```

delete resource

Parameters:

`uri` - target URI for deleting resource

---

discovery

```
org.osgi.util.promise.Promise discovery(String uri,
                                        FilterCriteriaDTO fc)
```

find resources. Discovery Result Type is kept as blank and default value of target CSE is used for the parameter.

Parameters:
`uri` - URI for top of search
`fc` - filter criteria
Returns:
list of URIs matching the condition specified in fc

---

discovery

```
org.osgi.util.promise.Promise discovery(String uri,
                                        FilterCriteriaDTO fc,
                                        RequestPrimitiveDTO.DiscoveryResultType drt)
```

find resources

Parameters:
`uri` - URI for top of search
`fc` - filter criteria
`drt` - Discovery Result Type (structured/unstructured)
Returns:
list of URIs matching the condition specified in fc

---

notify

```
org.osgi.util.promise.Promise notify(String uri,
                                     NotificationDTO notification)
```

send notification

Package org.osgi.service.onem2m.dto

Class Summary		Page
AttributeDTO	DTO expresses Attribute.	40
ChildResourceRefDTO	DTO expressing ChildResourceRef.	41
DasInfoDTO	DTO expressing DasInfo.	43
FilterCriteriaDTO	DTO expressing FilterCriteria.	45
FilterCriteriaDTO.FilterOperation	Enum FilterOperation	50
FilterCriteriaDTO.FilterUsage	Enum FilterUsage	52
GenericDTO	GenericDTO expresses miscellaneous data structures.	54
LocalTokenIdAssignmentDTO	DTO expressing LocalTokenIdAssignment.	55
NotificationDTO	DTO expressing Notification.	56
PrimitiveContentDTO	DTO expressing Primitive Content.	58
ReleaseVersion	Enum expressing oneM2M specification version.	61
RequestPrimitiveDTO	DTO expresses Request Primitive.	63
RequestPrimitiveDTO.DiscoveryResultType		68
RequestPrimitiveDTO.Operation	enum type for Operation	70
RequestPrimitiveDTO.ResultContent	enum type for Result Content	72
ResourceDTO	DTO expressing Resource.	75
ResourceWrapperDTO	DTO expressing ResourceWrapper.	77
ResponsePrimitiveDTO	DTO expressing Response Primitive.	78
ResponsePrimitiveDTO.ContentStatus	Enum ContentStatus	81
ResponseTypeInfoDTO	Expressing ResponseTypeInfo	83
ResponseTypeInfoDTO.ResponseType	enum ResponseType	84
SecurityInfoDTO	DTO expressing Security Info.	86

SecurityInfoDTO.SecurityInfoType	Enum SecurityInfoType	88

Class AttributeDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.AttributeDTO
```

```
public class AttributeDTO
extends org.osgi.dto.DTO
```

DTO expresses Attribute. This is typically used in FilterCriteriaDTO for expressing matching condition.

Field Summary		Page
	name Attribute name	40
	value Supposed value of the attribute	40

Constructor Summary	Page
AttributeDTO()	40

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

name

```
public name
```

Attribute name

value

```
public value
```

Supposed value of the attribute

Constructor Detail

AttributeDTO

```
public AttributeDTO()
```

Class ChildResourceRefDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └org.osgi.dto.DTO
    └org.osgi.service.onem2m.dto.ChildResourceRefDTO
```

```
public class ChildResourceRefDTO
extends org.osgi.dto.DTO
```

DTO expressing ChildResourceRef.

Field Summary		Page
	name name of the child resource pointed to by the URI	41
	specializationID resource type specialization of the child resource pointed to by the URI in case @type represents a flexContainer.	42
	type resourceType of the child resource pointed to by the URI	41
	uri URI to the child resource.	41

Constructor Summary	Page
ChildResourceRefDTO()	42

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

uri

```
public uri
```

URI to the child resource.

name

```
public name
```

name of the child resource pointed to by the URI

type

```
public type
```

resourceType of the child resource pointed to by the URI

---

specializationID

`public` **`specializationID`**

resource type specialization of the child resource pointed to by the URI in case @type represents a flexContainer. This is an optional field.

Constructor Detail

ChildResourceRefDTO

`public` **`ChildResourceRefDTO`**`()`

Class DasInfoDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.DasInfoDTO
```

```
public class DasInfoDTO
extends org.osgi.dto.DTO
```

DTO expressing DasInfo. DAS is short for Dynamic Authorization Server.

Field Summary		Page
	dasRequest Information to send to the Dynamic Authorization Server	43
	securedDasRequest Secured Information to send to the Dynamic Authorization Server.	43
	uri Dynamic Authorization Server URI	43

Constructor Summary	Page
DasInfoDTO()	44

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

uri

```
public uri
```

Dynamic Authorization Server URI

dasRequest

```
public dasRequest
```

Information to send to the Dynamic Authorization Server

securedDasRequest

```
public securedDasRequest
```

Secured Information to send to the Dynamic Authorization Server. JWS or JWE is assigned to this field.

Constructor Detail

DasInfoDTO

```
public DasInfoDTO()
```

Class FilterCriteriaDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.FilterCriteriaDTO
```

```
public class FilterCriteriaDTO
extends org.osgi.dto.DTO
```

DTO expressing FilterCriteria. This data structure is used for searching resources.

Nested Class Summary		Page
final static class	FilterCriteriaDTO.FilterOperation Enum FilterOperation	50
final static class	FilterCriteriaDTO.FilterUsage Enum FilterUsage	52

Field Summary		Page
	applyRelativePath Apply Relative Path	49
	attribute Attribute	48
	childAttribute Child Attribute	49
	childLabels Child Labels	49
	childResourceType Child Resource Type	49
	contentFilterQuery Content Filter Query	48
	contentFilterSyntax Content Filter Syntax	48
	contentType Content Type	48
	createdAfter Created After	46
	createdBefore Created Before	46
	expireAfter Expire After	47
	expireBefore Expire Before	47
	filterOperation Filter Operation	48
	filterUsage Filter Usage	48

	labels Labels	47
	labelsQuery Label Query	49
	level Level	48
	limit Limit number of Answers	48
	modifiedSince Modified Since	47
	offset Offset	48
	parentAttribute Parent Attribute	49
	parentLabels Parent Labels	49
	parentResourceType Parent Resource Type	49
	resourceType Resource Type	47
	semanticsFilter Semantic Filter	48
	sizeAbove Size Above	47
	sizeBelow Size Below	47
	stateTagBigger State Tag Bigger	47
	stateTagSmaller State Tag Smaller	47
	unmodifiedSince Unmodified Since	47

Constructor Summary	Page
FilterCriteriaDTO()	49

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

createdBefore

public **createdBefore**

Created Before

createdAfter

public **createdAfter**

Created After

modifiedSince

`public` **`modifiedSince`**

Modified Since

unmodifiedSince

`public` **`unmodifiedSince`**

Unmodified Since

stateTagSmaller

`public` **`stateTagSmaller`**

State Tag Smaller

stateTagBigger

`public` **`stateTagBigger`**

State Tag Bigger

expireBefore

`public` **`expireBefore`**

Expire Before

expireAfter

`public` **`expireAfter`**

Expire After

labels

`public` **`labels`**

Labels

resourceType

`public` **`resourceType`**

Resource Type

sizeAbove

`public` **`sizeAbove`**

Size Above

sizeBelow

`public` **`sizeBelow`**

Size Below

contentType

public **contentType**

Content Type

attribute

public **attribute**

Attribute

filterUsage

public **filterUsage**

Filter Usage

limit

public **limit**

Limit number of Answers

semanticsFilter

public **semanticsFilter**

Semantic Filter

filterOperation

public **filterOperation**

Filter Operation

contentFilterSyntax

public **contentFilterSyntax**

Content Filter Syntax

contentFilterQuery

public **contentFilterQuery**

Content Filter Query

level

public **level**

Level

offset

public **offset**

Offset

childLabels

public **childLabels**

Child Labels

parentLabels

public **parentLabels**

Parent Labels

labelsQuery

public **labelsQuery**

Label Query

childResourceType

public **childResourceType**

Child Resource Type

parentResourceType

public **parentResourceType**

Parent Resource Type

childAttribute

public **childAttribute**

Child Attribute

parentAttribute

public **parentAttribute**

Parent Attribute

applyRelativePath

public **applyRelativePath**

Apply Relative Path

Constructor Detail

FilterCriteriaDTO

public **FilterCriteriaDTO**()

Class FilterCriteriaDTO.FilterOperation

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.FilterCriteriaDTO.FilterOperation
```

All Implemented Interfaces:

Comparable, Serializable

Enclosing class:

FilterCriteriaDTO

```
final public static class FilterCriteriaDTO.FilterOperation
extends Enum
```

Enum FilterOperation

Field Summary		Page
static	**AND** AND	50
static	**OR** OR	50

Method Summary		Page
int	**getValue**() get assigned value	51
static FilterCriteriaDTO.FilterOperation	**valueOf**(String name)	51
static FilterCriteriaDTO.FilterOperation[]	**values**()	51

Field Detail

AND

```
public static final AND
```

AND

OR

```
public static final OR
```

OR

Method Detail

values

```
public static FilterCriteriaDTO.FilterOperation[] values()
```

valueOf

```
public static FilterCriteriaDTO.FilterOperation valueOf(String name)
```

getValue

```
public int getValue()
```

get assigned value

Returns:

assigned integer value

Class FilterCriteriaDTO.FilterUsage

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.FilterCriteriaDTO.FilterUsage
```

All Implemented Interfaces:
Comparable, Serializable

Enclosing class:
FilterCriteriaDTO

```
final public static class FilterCriteriaDTO.FilterUsage
extends Enum
```

Enum FilterUsage

Field Summary		Page
static	**ConditionalRetrival** Conditional Retrieve	52
static	**DiscoveryCriteria** Discovery Criteria	52
static	**IPEOndemandDiscovery** IPE on Demand Discovery	53

Method Summary		Page
int	**getValue**() get assigned integer value	53
static FilterCriteriaDTO.FilterUsage	**valueOf**(String name)	53
static FilterCriteriaDTO.FilterUsage[]	**values**()	53

Field Detail

DiscoveryCriteria

```
public static final DiscoveryCriteria
```

Discovery Criteria

ConditionalRetrival

```
public static final ConditionalRetrival
```

Conditional Retrieve

IPEOndemandDiscovery

```
public static final IPEOndemandDiscovery
```

IPE on Demand Discovery

Method Detail

values

```
public static FilterCriteriaDTO.FilterUsage[] values()
```

valueOf

```
public static FilterCriteriaDTO.FilterUsage valueOf(String name)
```

getValue

```
public int getValue()
```

get assigned integer value

Returns:
assigned integer value

Class GenericDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.GenericDTO
```

```
public class GenericDTO
extends org.osgi.dto.DTO
```

GenericDTO expresses miscellaneous data structures.

Field Summary		Page
	element Substructure of DTO.	54
	type type of data structure, which is represented by this DTO.	54

Constructor Summary	Page
GenericDTO()	54

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

type

```
public type
```

type of data structure, which is represented by this DTO. This is optional field. The creator of the DTO may use the field for clarification purpose. Receiver should not rely on this information to analyze data structure, since this information may not provided.

element

```
public element
```

Substructure of DTO. Type of the value part should be one of types allowed as OSGi DTO.

Constructor Detail

GenericDTO

```
public GenericDTO()
```

Class LocalTokenIdAssignmentDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.LocalTokenIdAssignmentDTO
```

```
public class LocalTokenIdAssignmentDTO
extends org.osgi.dto.DTO
```

DTO expressing LocalTokenIdAssignment.

Field Summary		Page
	localTokenID local token ID	55
	tokenID token ID	55

Constructor Summary	Page
LocalTokenIdAssignmentDTO()	55

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

localTokenID

```
public localTokenID
```

local token ID

tokenID

```
public tokenID
```

token ID

Constructor Detail

LocalTokenIdAssignmentDTO

```
public LocalTokenIdAssignmentDTO()
```

Class NotificationDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.NotificationDTO
```

```
public class NotificationDTO
extends org.osgi.dto.DTO
```

DTO expressing Notification.

Field Summary		Page
	creator creator	57
	ipeDiscoveryRequest IPE Discovery Request.	57
	notificationEvent Notification Event	56
	notificationForwardingURI URI for notification target	57
	subscriptionDeletion Flag showing subscription deletion This field is optional.	57
	subscriptionReference URI referring subscription resource.	57
	verificationRequest Flag showing verification request.	56

Constructor Summary	Page
NotificationDTO()	57

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

notificationEvent

```
public notificationEvent
```

Notification Event

verificationRequest

```
public verificationRequest
```

Flag showing verification request. This field is optional.

---

subscriptionDeletion

`public` **`subscriptionDeletion`**

Flag showing subscription deletion This field is optional.

subscriptionReference

`public` **`subscriptionReference`**

URI referring subscription resource.

creator

`public` **`creator`**

creator

notificationForwardingURI

`public` **`notificationForwardingURI`**

URI for notification target

ipeDiscoveryRequest

`public` **`ipeDiscoveryRequest`**

IPE Discovery Request.

Constructor Detail

NotificationDTO

`public` **`NotificationDTO`**`()`

Class PrimitiveContentDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.PrimitiveContentDTO
```

```
public class PrimitiveContentDTO
extends org.osgi.dto.DTO
```

DTO expressing Primitive Content. This Data structure is used as union. Only one field MUST have a value, the others MUST be null.

Field Summary		Page
	aggregatedNotification Aggregated Notification	59
	aggregatedResponse Aggregated Response	59
	attributeList Attribute List	60
	childResourceRefList Child Resource RefList	60
	debugInfo Debug Info	59
	listOfURIs List Of URIs	59
	notification Notification	60
	queryResult Query Result	60
	requestPrimitive Request Primitive	60
	resource Resource	59
	resourceWrapper Resource Wrapper	59
	responsePrimitive Response Primitive	59
	securityInfo Security Info	59
	uri URI	59

Constructor Summary	Page
PrimitiveContentDTO()	60

Methods inherited from class org.osgi.dto.DTO

toString

Field Detail

resource

public **resource**

Resource

resourceWrapper

public **resourceWrapper**

Resource Wrapper

aggregatedNotification

public **aggregatedNotification**

Aggregated Notification

securityInfo

public **securityInfo**

Security Info

responsePrimitive

public **responsePrimitive**

Response Primitive

debugInfo

public **debugInfo**

Debug Info

listOfURIs

public **listOfURIs**

List Of URIs

uri

public **uri**

URI

aggregatedResponse

public **aggregatedResponse**

Aggregated Response

childResourceRefList

`public` **`childResourceRefList`**

Child Resource RefList

notification

`public` **`notification`**

Notification

attributeList

`public` **`attributeList`**

Attribute List

requestPrimitive

`public` **`requestPrimitive`**

Request Primitive

queryResult

`public` **`queryResult`**

Query Result

Constructor Detail

PrimitiveContentDTO

`public` **`PrimitiveContentDTO`**`()`

Class ReleaseVersion

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
      └ org.osgi.service.onem2m.dto.ReleaseVersion
```

All Implemented Interfaces:

Comparable, Serializable

```
final public class ReleaseVersion
extends Enum
```

Enum expressing oneM2M specification version. This information is introduced after Release 2.0 and oneM2M uses only R2A,R3_0 (as 2a and 3).

Field Summary		Page
static	**R1_0** Release 1	61
static	**R1_1** Release 1.1	61
static	**R2_0** Release 2	62
static	**R2A** Release 2A	62
static	**R3_0** Release 3	62

Method Summary		Page
static ReleaseVersion	**valueOf**(String name)	62
static ReleaseVersion[]	**values**()	62

Field Detail

R1_0

```
public static final R1_0
```

Release 1

R1_1

```
public static final R1_1
```

Release 1.1

R2_0

```
public static final R2_0
```

Release 2

R2A

```
public static final R2A
```

Release 2A

R3_0

```
public static final R3_0
```

Release 3

Method Detail

values

```
public static ReleaseVersion[] values()
```

valueOf

```
public static ReleaseVersion valueOf(String name)
```

Class RequestPrimitiveDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.RequestPrimitiveDTO
```

```
public class RequestPrimitiveDTO
extends org.osgi.dto.DTO
```

DTO expresses Request Primitive.

Nested Class Summary		Page
final static class	RequestPrimitiveDTO.DiscoveryResultType	68
final static class	RequestPrimitiveDTO.Operation enum type for Operation	70
final static class	RequestPrimitiveDTO.ResultContent enum type for Result Content	72

Field Summary		Page
	authorRelIndicator Author Relation Indicator	67
	authorSignIndicator Author Sign Indicator	67
	authorSigns Author Signs	67
	content Primitive Content	65
	deliveryAggregation Delivery Aggregation	66
	discoveryResultType Discovery Result Type	66
	eventCategory Event Category	66
	filterCriteria Filter Criteria	66
	from From Parameter.	65
	groupRequestIdentifier Group Request Identifier	66
	groupRequestTargetMembers Group Request Target Members	67
	localTokenIDs Local Token Identifiers	67
	operation Operation	64

	operationExecutionTime Operation Execution Time	65
	originatingTimestamp Originating Timestamp	65
	releaseVersionIndicator Release Version	67
	requestExpirationTimestamp Request Expiration Timestamp	65
	requestIdentifier Request Identifier	65
	resourceType Resource Type	65
	responseType Response Type Info	66
	resultContent Result Content	66
	resultExpirationTimestamp Result Expiration Timestamp	65
	resultPersistence Result Persistence	66
	roleIDs Role IDs	65
	semanticQueryIndicator Semantic Query Indicator	67
	to To Parameter	65
	tokenIDs Token Identifiers	66
	tokenRequestIndicator Token Request Indicator	67
	tokens tokens	66
	vendorInformation Vendor Information	67

Constructor Summary	***Page***
RequestPrimitiveDTO()	67

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

operation

public **operation**

Operation

to

`public to`

To Parameter

from

`public from`

From Parameter. In other word, originator of request is stored.

requestIdentifier

`public requestIdentifier`

Request Identifier

resourceType

`public resourceType`

Resource Type

content

`public content`

Primitive Content

roleIDs

`public roleIDs`

Role IDs

originatingTimestamp

`public originatingTimestamp`

Originating Timestamp

requestExpirationTimestamp

`public requestExpirationTimestamp`

Request Expiration Timestamp

resultExpirationTimestamp

`public resultExpirationTimestamp`

Result Expiration Timestamp

operationExecutionTime

`public operationExecutionTime`

Operation Execution Time

responseType

`public` **`responseType`**

Response Type Info

resultPersistence

`public` **`resultPersistence`**

Result Persistence

resultContent

`public` **`resultContent`**

Result Content

eventCategory

`public` **`eventCategory`**

Event Category

deliveryAggregation

`public` **`deliveryAggregation`**

Delivery Aggregation

groupRequestIdentifier

`public` **`groupRequestIdentifier`**

Group Request Identifier

filterCriteria

`public` **`filterCriteria`**

Filter Criteria

discoveryResultType

`public` **`discoveryResultType`**

Discovery Result Type

tokens

`public` **`tokens`**

tokens

tokenIDs

`public` **`tokenIDs`**

Token Identifiers

localTokenIDs

`public` **`localTokenIDs`**

Local Token Identifiers

tokenRequestIndicator

`public` **`tokenRequestIndicator`**

Token Request Indicator

groupRequestTargetMembers

`public` **`groupRequestTargetMembers`**

Group Request Target Members

authorSignIndicator

`public` **`authorSignIndicator`**

Author Sign Indicator

authorSigns

`public` **`authorSigns`**

Author Signs

authorRelIndicator

`public` **`authorRelIndicator`**

Author Relation Indicator

semanticQueryIndicator

`public` **`semanticQueryIndicator`**

Semantic Query Indicator

releaseVersionIndicator

`public` **`releaseVersionIndicator`**

Release Version

vendorInformation

`public` **`vendorInformation`**

Vendor Information

Constructor Detail

RequestPrimitiveDTO

`public` **`RequestPrimitiveDTO`**`()`

Class RequestPrimitiveDTO.DiscoveryResultType

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.RequestPrimitiveDTO.DiscoveryResultType
```

All Implemented Interfaces:

Comparable, Serializable

Enclosing class:

RequestPrimitiveDTO

```
final public static class RequestPrimitiveDTO.DiscoveryResultType
extends Enum
```

Field Summary		Page
static	**structured** structured	68
static	**unstructured** unstructured	68

Method Summary		Page
int	**getValue**()	69
static RequestPrimitiveDTO.DiscoveryResultType	**valueOf**(String name)	69
static RequestPrimitiveDTO.DiscoveryResultType[]	**values**()	69

Field Detail

structured

```
public static final structured
```

structured

unstructured

```
public static final unstructured
```

unstructured

Method Detail

values

public static RequestPrimitiveDTO.DiscoveryResultType[] **values**()

valueOf

public static RequestPrimitiveDTO.DiscoveryResultType **valueOf**(String name)

getValue

public int **getValue**()

Class RequestPrimitiveDTO.Operation

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.RequestPrimitiveDTO.Operation
```

All Implemented Interfaces:

Comparable, Serializable

Enclosing class:

RequestPrimitiveDTO

```
final public static class RequestPrimitiveDTO.Operation
extends Enum
```

enum type for Operation

Field Summary		Page
static	**Create** Create	70
static	**Delete** Delete	71
static	**Notify** Notify	71
static	**Retrieve** Retrieve	71
static	**Update** Update	71

Method Summary		Page
int	**getValue**() get assigned integer value	71
static RequestPrimitiveDTO.Operation	**valueOf**(String name)	71
static RequestPrimitiveDTO.Operation[]	**values**()	71

Field Detail

Create

```
public static final Create
```

Create

Retrieve

```
public static final Retrieve
```

Retrieve

Update

```
public static final Update
```

Update

Delete

```
public static final Delete
```

Delete

Notify

```
public static final Notify
```

Notify

Method Detail

values

```
public static RequestPrimitiveDTO.Operation[] values()
```

valueOf

```
public static RequestPrimitiveDTO.Operation valueOf(String name)
```

getValue

```
public int getValue()
```

get assigned integer value

Returns:
assigned integer value

Class RequestPrimitiveDTO.ResultContent

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.RequestPrimitiveDTO.ResultContent
```

All Implemented Interfaces:
Comparable, Serializable

Enclosing class:
RequestPrimitiveDTO

```
final public static class RequestPrimitiveDTO.ResultContent
extends Enum
```

enum type for Result Content

Field Summary		Page
static	**attributes** attributes	73
static	**attributesAndChildResourceReferences** attributesAndChildResourceReferences	73
static	**attributesAndChildResources** attributesAndChildResources	73
static	**childResourceReferences** childResourceReferences	73
static	**childResources** childResources	73
static	**hierarchicalAddress** hierarchicalAddress	73
static	**hierarchicalAddressAndAttributes** hierarchicalAddressAndAttributes	73
static	**nothing** nothing	73
static	**originalResource** originalResource	73

Method Summary		Page
int	**getValue**() get assigned integer value	74
static RequestPrimitiveDTO.ResultContent	**valueOf**(String name)	74
static RequestPrimitiveDTO.ResultContent[]	**values**()	74

Field Detail

nothing

```
public static final nothing
```

nothing

attributes

```
public static final attributes
```

attributes

hierarchicalAddress

```
public static final hierarchicalAddress
```

hierarchicalAddress

hierarchicalAddressAndAttributes

```
public static final hierarchicalAddressAndAttributes
```

hierarchicalAddressAndAttributes

attributesAndChildResources

```
public static final attributesAndChildResources
```

attributesAndChildResources

attributesAndChildResourceReferences

```
public static final attributesAndChildResourceReferences
```

attributesAndChildResourceReferences

childResourceReferences

```
public static final childResourceReferences
```

childResourceReferences

originalResource

```
public static final originalResource
```

originalResource

childResources

```
public static final childResources
```

childResources

Method Detail

values

```
public static RequestPrimitiveDTO.ResultContent[] values()
```

valueOf

```
public static RequestPrimitiveDTO.ResultContent valueOf(String name)
```

getValue

```
public int getValue()
```

get assigned integer value

Returns:

assigned integer value

Class ResourceDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.ResourceDTO
```

```
public class ResourceDTO
extends org.osgi.dto.DTO
```

DTO expressing Resource.

Field Summary		Page
	attribute Non Universal Attribute.	76
	creationTime Creation time	76
	labels Labels This field is optional.	76
	lastModifiedTime last modified time	76
	parentID Parent ID Resource ID of parent resource.	76
	resourceID Resource ID	75
	resourceName Resource name	76
	resourceType Resource Type	75

Constructor Summary	Page
ResourceDTO()	76

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

resourceType

```
public resourceType
```

Resource Type

resourceID

```
public resourceID
```

Resource ID

parentID

`public` **`parentID`**

Parent ID Resource ID of parent resource.

creationTime

`public` **`creationTime`**

Creation time

lastModifiedTime

`public` **`lastModifiedTime`**

last modified time

resourceName

`public` **`resourceName`**

Resource name

labels

`public` **`labels`**

Labels This field is optional.

attribute

`public` **`attribute`**

Non Universal Attribute. Value Part must be the types that are allowed for OSGi DTO.

Constructor Detail

ResourceDTO

`public` **`ResourceDTO`**`()`

Class ResourceWrapperDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.ResourceWrapperDTO
```

```
public class ResourceWrapperDTO
extends org.osgi.dto.DTO
```

DTO expressing ResourceWrapper.

Field Summary		Page
	resource Resource	77
	uri URI for the resource	77

Constructor Summary	Page
ResourceWrapperDTO()	77

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

uri

```
public uri
```

URI for the resource

resource

```
public resource
```

Resource

Constructor Detail

ResourceWrapperDTO

```
public ResourceWrapperDTO()
```

Class ResponsePrimitiveDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.ResponsePrimitiveDTO
```

```
public class ResponsePrimitiveDTO
extends org.osgi.dto.DTO
```

DTO expressing Response Primitive.

Nested Class Summary		Page
final static class	ResponsePrimitiveDTO.ContentStatus Enum ContentStatus	81

Field Summary		Page
	assignedTokenIdentifiers Assigned Token Identifiers	80
	AuthSignatureReqInfo AuthSignatureReqInfo	80
	content Primitive Content	79
	contentOffset Content Offset	80
	contentStatus Content Status	80
	eventCategory Event Category	79
	from From Parameter	79
	originatingTimestamp Originating Timestamp	79
	releaseVersionIndicator Release Version Indicator	80
	requestIdentifier Request Identifier	79
	responseStatusCode Response Status Code	79
	resultExpirationTimestamp ResultExpiration Timestamp	79
	to To Parameter	79
	tokenReqInfo Token Request Info	80
	vendorInformation Vendor Information	80

Constructor Summary	Page
ResponsePrimitiveDTO()	80

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

responseStatusCode

public **responseStatusCode**

Response Status Code

requestIdentifier

public **requestIdentifier**

Request Identifier

content

public **content**

Primitive Content

to

public **to**

To Parameter

from

public **from**

From Parameter

originatingTimestamp

public **originatingTimestamp**

Originating Timestamp

resultExpirationTimestamp

public **resultExpirationTimestamp**

ResultExpiration Timestamp

eventCategory

public **eventCategory**

Event Category

contentStatus

`public` **`contentStatus`**

Content Status

contentOffset

`public` **`contentOffset`**

Content Offset

assignedTokenIdentifiers

`public` **`assignedTokenIdentifiers`**

Assigned Token Identifiers

tokenReqInfo

`public` **`tokenReqInfo`**

Token Request Info

AuthSignatureReqInfo

`public` **`AuthSignatureReqInfo`**

AuthSignatureReqInfo

releaseVersionIndicator

`public` **`releaseVersionIndicator`**

Release Version Indicator

vendorInformation

`public` **`vendorInformation`**

Vendor Information

Constructor Detail

ResponsePrimitiveDTO

`public` **`ResponsePrimitiveDTO`**`()`

Class ResponsePrimitiveDTO.ContentStatus

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.ResponsePrimitiveDTO.ContentStatus
```

All Implemented Interfaces:

Comparable, Serializable

Enclosing class:

ResponsePrimitiveDTO

```
final public static class ResponsePrimitiveDTO.ContentStatus
extends Enum
```

Enum ContentStatus

Field Summary		Page
static	**FULL_CONTENT** FULL_CONTENT	81
static	**PARTIAL_CONTENT** PARTIAL_CONTENT	81

Method Summary		Page
static ResponsePrimitiveDTO.ContentStatus	**valueOf**(String name)	82
static ResponsePrimitiveDTO.ContentStatus[]	**values**()	82

Field Detail

PARTIAL_CONTENT

```
public static final PARTIAL_CONTENT
```

PARTIAL_CONTENT

FULL_CONTENT

```
public static final FULL_CONTENT
```

FULL_CONTENT

Method Detail

values

```
public static ResponsePrimitiveDTO.ContentStatus[] values()
```

valueOf

```
public static ResponsePrimitiveDTO.ContentStatus valueOf(String name)
```

Class ResponseTypeInfoDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.ResponseTypeInfoDTO
```

```
public class ResponseTypeInfoDTO
extends org.osgi.dto.DTO
```

Expressing ResponseTypeInfo

Nested Class Summary		Page
final static class	ResponseTypeInfoDTO.ResponseType enum ResponseType	84

Field Summary		Page
	notificationURI Notification URI	83
	responseTypeValue Response Type Value	83

Constructor Summary	Page
ResponseTypeInfoDTO()	83

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

responseTypeValue

```
public responseTypeValue
```

Response Type Value

notificationURI

```
public notificationURI
```

Notification URI

Constructor Detail

ResponseTypeInfoDTO

```
public ResponseTypeInfoDTO()
```

Class ResponseTypeInfoDTO.ResponseType

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.ResponseTypeInfoDTO.ResponseType
```

All Implemented Interfaces:

Comparable, Serializable

Enclosing class:

ResponseTypeInfoDTO

```
final public static class ResponseTypeInfoDTO.ResponseType
extends Enum
```

enum ResponseType

Field Summary		Page
static	**blockingRequest** blockingRequest	85
static	**flexBlocking** flexBlocking	85
static	**nonBlockingRequestAsynch** nonBlockingRequestAsynch	84
static	**nonBlockingRequestSynch** nonBlockingRequestSynch	84

Method Summary		Page
int	**getValue**() get assigned value	85
static ResponseTypeInfoDTO.ResponseType	**valueOf**(String name)	85
static ResponseTypeInfoDTO.ResponseType[]	**values**()	85

Field Detail

nonBlockingRequestSynch

```
public static final nonBlockingRequestSynch
```

nonBlockingRequestSynch

nonBlockingRequestAsynch

```
public static final nonBlockingRequestAsynch
```

nonBlockingRequestAsynch

blockingRequest

```
public static final blockingRequest
```

blockingRequest

flexBlocking

```
public static final flexBlocking
```

flexBlocking

Method Detail

values

```
public static ResponseTypeInfoDTO.ResponseType[] values()
```

valueOf

```
public static ResponseTypeInfoDTO.ResponseType valueOf(String name)
```

getValue

```
public int getValue()
```

get assigned value

Returns:

assigned integer value.

Class SecurityInfoDTO

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ org.osgi.dto.DTO
    └ org.osgi.service.onem2m.dto.SecurityInfoDTO
```

```
public class SecurityInfoDTO
extends org.osgi.dto.DTO
```

DTO expressing Security Info.

Nested Class Summary		Page
final static class	SecurityInfoDTO.SecurityInfoType Enum SecurityInfoType	88

Field Summary		Page
	dasRequest Das Request	86
	dasResponse Das Response	87
	escertkeMessage Escertke Message	87
	esprimObject Esprim Object	87
	esprimRandObject Esprim Rand Objecgt	87
	securityInfoType Security Info Type	86

Constructor Summary	Page
SecurityInfoDTO()	87

Methods inherited from class org.osgi.dto.DTO
toString

Field Detail

securityInfoType

```
public securityInfoType
```

Security Info Type

dasRequest

```
public dasRequest
```

Das Request

dasResponse

```
public dasResponse
```

Das Response

esprimRandObject

```
public esprimRandObject
```

Esprim Rand Objecgt

esprimObject

```
public esprimObject
```

Esprim Object

escertkeMessage

```
public escertkeMessage
```

Escertke Message

Constructor Detail

SecurityInfoDTO

```
public SecurityInfoDTO()
```

Class SecurityInfoDTO.SecurityInfoType

org.osgi.service.onem2m.dto

```
java.lang.Object
  └ java.lang.Enum
    └ org.osgi.service.onem2m.dto.SecurityInfoDTO.SecurityInfoType
```

All Implemented Interfaces:

Comparable, Serializable

Enclosing class:

SecurityInfoDTO

```
final public static class SecurityInfoDTO.SecurityInfoType
extends Enum
```

Enum SecurityInfoType

Field Summary		Page
static	**DynamicAuthorizationRelationshipMappingRequest** DynamicAuthorizationRelationshipMappingRequest	89
static	**DynamicAuthorizationRelationshipMappingResponse** DynamicAuthorizationRelationshipMappingResponse	89
static	**DynamicAuthorizationRequest** DynamicAuthorizationRequest	89
static	**DynamicAuthorizationResponse** DynamicAuthorizationResponse	89
static	**ESCertKEMessage** ESCertKEMessage	89
static	**ESPrimObject** ESPrimObject	89
static	**ReceiverESPrimRandObjectRequest** ReceiverESPrimRandObjectRequest	89
static	**ReceiverESPrimRandObjectResponse** ReceiverESPrimRandObjectResponse	89

Method Summary		Page
int	**getValue**() Get assigned value.	90
static SecurityInfoDTO.SecurityInfoType	**valueOf**(String name)	89
static SecurityInfoDTO.SecurityInfoType[]	**values**()	89

Field Detail

DynamicAuthorizationRequest

```
public static final DynamicAuthorizationRequest
```

DynamicAuthorizationRequest

~~DynamicAuthorizationResponse~~

```
public static final DynamicAuthorizationResponse
```

DynamicAuthorizationResponse

~~ReceiverESPrimRandObjectRequest~~

```
public static final ReceiverESPrimRandObjectRequest
```

ReceiverESPrimRandObjectRequest

~~ReceiverESPrimRandObjectResponse~~

```
public static final ReceiverESPrimRandObjectResponse
```

ReceiverESPrimRandObjectResponse

~~ESPrimObject~~

```
public static final ESPrimObject
```

ESPrimObject

~~ESCertKEMessage~~

```
public static final ESCertKEMessage
```

ESCertKEMessage

~~DynamicAuthorizationRelationshipMappingRequest~~

```
public static final DynamicAuthorizationRelationshipMappingRequest
```

DynamicAuthorizationRelationshipMappingRequest

~~DynamicAuthorizationRelationshipMappingResponse~~

```
public static final DynamicAuthorizationRelationshipMappingResponse
```

DynamicAuthorizationRelationshipMappingResponse

Method Detail

values

```
public static SecurityInfoDTO.SecurityInfoType[] values()
```

~~valueOf~~

```
public static SecurityInfoDTO.SecurityInfoType valueOf(String name)
```

getValue

```
public int getValue()
```

Get assigned value.

Returns:
assigned value

8 Considered Alternatives

~~For posterity, record the design alternatives that were considered but rejected along with the reason for rejection. This is especially important for external/earlier solutions that were deemed not applicable.~~

8.1 Representation of DTO

8.1.1 JAXB generated Class

As alternative solution, utilization of generated Java classes by JAXB has been considered, since oneM2M provides well defined XSD for defining data format. With the following aspects, this approach is not applied.

Many classes: Currently 65 XSD files are defined in oneM2M specification and JAXB tool (xjc) generates more than 140 Java classes. Using many classes as interface could make specification more complicated than its nature.

No Uniqueness: Generated classes by xjc are not unique, because it is possible to customize generation processes.

Changeability: Depending on the version of oneM2M, XSD files differ. It is preferable to choose version independent API, as much as possible. oneM2M ensures any data can be converted to JSON and CBOR, so proposed approach can be used with out modification, even if XSD file would be changed.

8.1.2 Generic DTO

Generic~~Genric~~ DTO, which has Map<String, Obj> in the top, has been discussed in Gent meeting. But it seems bad usage of defining DTO.

8.1.3 Specific DTO

SpecificDTO definitions have been generated from XSD generated classes. The number of DTO exceeds 170 and Java doc pages are getting 300 pages. It is apparently too much to express data formats. So middle approach of generic DTO and specific DTO has been chosen.

8.2 Resource Types Expression

In DTO, enum was eagerly used for clear candidates of possible values. But resource types seems more fragile because new resource types could be easily added. So Integer was chosen for resource types.

8.3 Use of Annotation defined by JAXB in DTO

Currently annotations defined by JAXB was used in DTO. It was pointed out as confusing because it might give impression that it only support XML serialization. But it was kept in the definitions by following reasons.

1. Removing the annotations are easier than inserting.

2. It is informative to specify optionality.

New OSGI annotation specifying optionality could be possible, but it might take time because it should be published as Core specification and R7 just has released.

8.4 Service Property for Service Layer Interface

In previous version, property named as PoAforAE is supposed to attach to Service Layer Interface. The way application need to search dedicated service for the AE, however, is not easy for developers nor OSGi way. The property has been removed and matching will be done by using Service Factory and mapping of bundle location and appropriate PoA is known by the service implementation.

8.5 Service Property for Notification Listener Interface

In previous version, property named as PoA and AE-ID are supposed to attach to Notification Listener Interface.

To attache AE-ID, application need to check result of <AE> resource creation and know modified AE-ID by CSE, and attach it as a property. This is complicated procedure and it is not a good design choice requiring all applications to implement this manner. Regarding PoA, the design that AE does not need to care of PoA was chosen.

Implementation of Service Layer Interface should manage both information by monitoring method request for creating <AE> resource and modifying the response. [This part need to be confirmed by reference implementation.]

9 Security Considerations

~~*Description of all known vulnerabilities this may either introduce or address as well as scenarios of how the weaknesses could be circumvented.*~~

9.1 ServiceLayer~~ProtocolBinding~~ Service with secure protocol configuration

In case that ~~ProtocolBinding~~ServiceLayer Service uses secure protocols, it is expected to handle pre-shared key or certificate and other parameters. Those configuration could be very diverse. This is out of scope of this RFC and it is responsibility of bundle developer that provides ProtocolBindingService.

9.2 Binding of AE Core and Protocol Binding

Protocol Binding has identity information, such as a key or certificate, which represents AE, so that AE core MUST be bound to the right protocol service binding. It is implementation's responsibility of protocol binding, it SHOULD utilize Service Factory to determine calling entity.

10 Document Support

10.1 References

[1]. Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, RFC2119, March 1997.

[2]. oneM2M TS-0001 Functional Architecture, http://onem2m.org/images/files/deliverables/Release2/TS-0001-%20Functional_Architecture-V2_10_0.pdf

[3]. oneM2M TS-0004 Service Layer Core Protocol, http://onem2m.org/images/files/deliverables/Release2/TS-0004_Service_Layer_Core_Protocol_V2_7_1.zip

[4]. oneM2M TS-0001 Functional Architecture Draft v3.11.0, http://www.onem2m.org/technical/published-drafts

[5]. oneM2M TS-0004 Service Layer Core Protocol Draft v3.7.0, http://www.onem2m.org/technical/published-drafts

[6]. XSD files for oneM2M, https://git.onem2m.org/PRO/XSD.git

[7]. Software Requirements & Specifications. Michael Jackson. ISBN 0-201-87712-0 (NOTE:Is this needed?)

Add references simply by adding new items. You can then cross-refer to them by chosing <Insert><Cross Reference><Numbered Item> and then selecting the paragraph. STATIC REFERENCES (I.E. BODGED) ARE NOT ACCEPTABLE, SOMEONE WILL HAVE TO UPDATE THEM LATER, SO DO IT PROPERLY NOW.

10.2 Author's Address

Name	Hiroyuki Maeomichi
Company	NTT
Address	Midorimachi 3-9-11, Musashino, Tokyo, Japan
Voice	+81 422 59 4072
e-mail	maeomichi.hiroyuki@lab.ntt.co.jp

10.3 Acronyms and Abbreviations

CSE: Common Services Entity

AE: Application Entity

CBOR: Concise Binary Object Representation

10.4 End of Document