RFC 69 - Metatyping for the Control Units and Diagnostics

Confidential, Draft

32 Pages

Abstract

This RFC presents an extension of RFC 62 - Metatype Update and defines metatype information for Control Units and Diagnostics.

0 Document Information

0.1 Table of Contents

0 Document Information 2
0.1 Table of Contents 2
0.2 Terminology and Document Conventions 3
0.3 Revision History 3

1 Introduction 5

2 Application Domain 5

3 Problem Description 6

4 Requirements 6

5 Technical Solution 7
5.1 Solution description 7
5.2 API Specification 8
5.2.1 Package org.osgi.service.metatype2 8
5.3 MetaType Providing 15
5.3.1 Providing MetaType with a MetaTypeProvider 15
5.3.2 Providing MetaType with XML Definition 15
5.3.3 MetaType XML Format XSD 16
5.3.4 Example MetaType Definition 18
5.4 The MetaDataService 20
5.4.1 MetaDataService API 20
5.4.2 MetaDataListener API 23

6 Considered Alternatives 25
6.1 Extended Attributes and Object Class Definitions 25
6.2 Representing Control Unit and Diagnostics Objects 25
6.3 Javadoc 26
6.3.1 org.osgi.service.metatype2 Interface ExtendedObjectClassDefinition 26
6.3.2 org.osgi.service.metatype2 Interface ExtendedAttributeDefinition 28
6.3.3 Constant Field Values 31

7 Security Considerations 31

8 Document Support 32
8.1 References 32

All Page Within This Box

8.2 Author's Address 32
8.3 Acronyms and Abbreviations 32
8.4 End of Document 32

0.2 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 [1].

```
Source code is shown in this typeface.
```

0.3 Revision History

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

Revision	Date	Comments
Initial	Nov 2 2004	Pavlin Dobrev, ProSyst Software AG, p.dobrev@prosyst.com
2nd	Dec 20 2004	Olivier Pavé, Siemens AG, olivier.pave@siemens.com
3rd	Jan 11 2005	Olivier Pavé. Remove DICTIONARY type in alternative solution and rename OBJECT_CLASS_DEFINITION type to OBJECT_CLASS Move alternative solution to main section and move technical solution to alternative solution.
4th	Jan 13 2005	Jordan Simeonov, ProSyst, j.simeonov@prosyst.com Add a DTD and example of XML format for metatype defintion.
5th	Jan 26 2005	Rumen Monev, ProSyst, r_monev@prosyst.com DTD and example replaced with XSD & conforming example
6th	Feb 07 2005	Rumen Monev, ProSyst, r_monev@prosyst.com Minor modifications in the XML schema according to the Olivier Pavé's remarks.
7th	Apr 01, 2005	Olivier Pavé, Siemens AG, olivier.pave@siemens.com Improvement of the specification and fix of typos.
8th	Apr 07, 2005	Nickola Jetchev, Prosyst, n_jetchev@prosyst.bg MetaDataService and MetaDataListener defined.

All Page Within This Box

Revision	Date	Comments

9th	Apr 07, 2005	Nickola Jetchev, Prosyst, n_jetchev@prosyst.bg
		
		Method `getInputArgumentDefinition()` of the `ActionDefinition` interface renamed to `getInputArgumentDefinitions`() and now returns an array of extended attribute definitions.
		
		Method
		
		`ObjectClassDefinition getObjectClassDefinition()`
		
		of the `ExtendedAttributeDefinition` interface replaced with
		
		`ExtendedAttributeDefinition[] getAttributeDefinitions()`
		
		method.
		
		Section 6.2 updated to reflect the last changes.

10th	Apr 18 , 2005	Nickola Jetchev, Prosyst, n_jetchev@prosyst.bg Method `getObjectClassDefinition`() of the `MetaDataService` interface now returns `ObjectClassDefinition` instead of `ExtendedObjectClassDefinition`. MetaType XSD updated according to the last changes. Method `getIcon(int size)` added to `ExtendedObjectClassDefinition`. `MetaDataService` JavaDoc extended to define the service behavior in response to some errors.
11th	Apr 22, 2005	Nickola Jetchev, Prosyst, n_jetchev@prosyst.bg Localization of the MetaType provided trough XML resource described. Minor changes of the XML Schema and the example XML definition – action and attribute definitions moved before object class definition to easy parsing, root element renamed to `ExtendedMetaData`, name and description optional attributes added to `ObjectClassDefinition` element. Removed the “name” attribute from the “Meta-Type” header definition. Added object class ID parameter to the `getObjectClassDefinition` method of `MetaDataService`.
12th	May 18, 2005	Olivier Pavé, Siemens AG, olivier.pave@siemens.com Update of the class diagram.

All Page Within This Box

Revision	Date	Comments
13th	May 19, 2005	Nickola Jetchev, Prosyst, n_jetchev@prosyst.bg Changes to the XML Schema - ExADTypes enumeration now extends the metatype:Scalar enumeration and ActionTypes extends ExADTypes.

1 Introduction

The main idea of the proposed metatype extension is to provide metatype suitable for Control Units and Diagnostics. It requires defining an extension in the area of metatyping actions and state variables.

This document uses the term “control unit” as a generic interface for “device”, “driver” and “service” in order to avoid the repeated and excessive use of these terms.

The metatype extension defined in this document is not dedicated only to control unit and can be used in other contexts.

2 Application Domain

This document proposes definition of metatyping information that is suitable for representation of Control Unit and Diagnostics information.

As mentioned in the introduction section, the metatype extension can be used in other domains.

All Page Within This Box

3 Problem Description

Applications need to be able to provide users with a user-friendly representation of every control unit. Because of that, the applications need to be able to get a human readable description of the control unit and its actions from the control unit interface.

The application should be able to query what variables and actions are available in the control unit. It should be able to read the data from a variable without dependencies to protocol- or device-specific object types (e.g. returning an Object should not be allowed, the types used for variables and action arguments should be restricted to primitive types and perhaps vectors and arrays of primitive types). By restricting the data types only to primitive types will make them easily displayed in any HMI, stored persistently, created from any input means, and transported over different protocols without any problems.

4 Requirements

According to the requirements defined by RFP 47 - Control Unit [3]:

• An application should be able to query the control unit for a human readable description for its variables and actions;

• Every control unit should be uniquely identifiable by an ID (String/long/etc.);

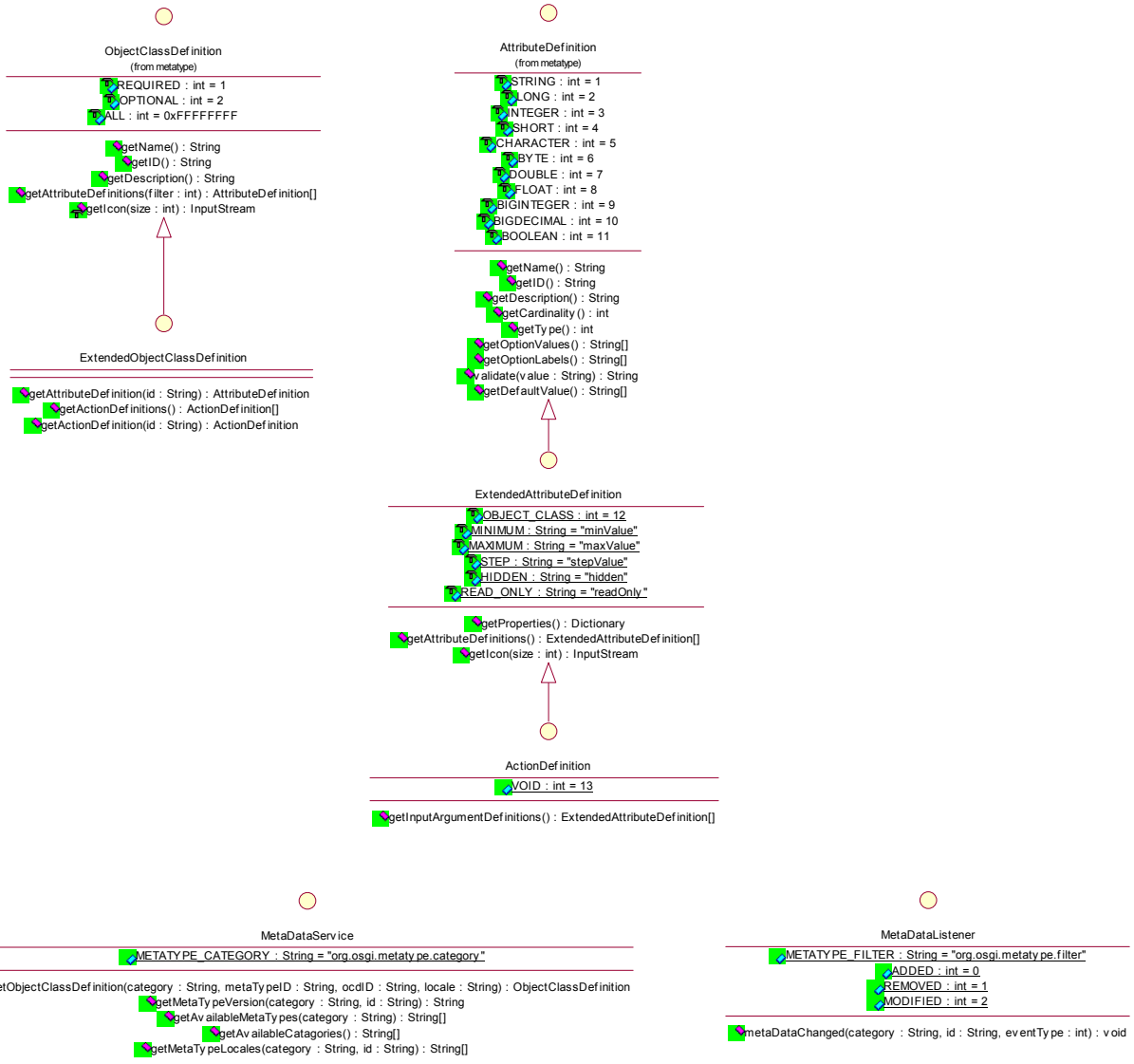
• The information should support localization;

• The state variables and action arguments should offer metatyping information about themselves, which facilitates the building of helpful visual interface – e.g. data type, min/max value, enumerated possible values, default value.

5 Technical Solution

5.1 Solution description

The solution proposed is to define an action as an extension of an attribute definition because it is more less the same thing with additional information: the list of input arguments.



The solution is based on three interfaces:

All Page Within This Box

- `ExtendedObjectClassDefintion` that extends `ObjectClassDefinition` to add the access to all attribute definitions, to one action definition or to all action definitions;
- `ExtendedAttributeDefinition` that extends `AttributeDefintion` to add additional properties to an attribute and new types;
- `ActionDefintion` that extends `ExtendedAttributeDefintion` to add an access to input argument definitions of the action. Note that the ouput results of an action is defined by the attribute definition type.

5.2 API Specification

5.2.1 Package org.osgi.service.metatype2

Interface Summary	
ActionDefinition	An interface to describe an action.
ExtendedAttributeDefinition	An interface to describe a complex attribute.
ExtendedObjectClassDefinition	Defines an object class that may contain attributes and actions.

5.2.2 Interface ExtendedObjectClassDefinition

All Superinterfaces:
org.osgi.service.metatype.ObjectClassDefinition

public interface **ExtendedObjectClassDefinition**
extends org.osgi.service.metatype.ObjectClassDefinition

Defines an object class that may contain attributes and actions.

Version:
$Revision: 1.1 $

Field Summary

Fields inherited from interface org.osgi.service.metatype.ObjectClassDefinition
`ALL, OPTIONAL, REQUIRED`

Method Summary	
`ActionDefinition`	**`getActionDefinition`**`(java.lang.String id)` Return the definition of the action with the specified ID.
`ActionDefinition[]`	**`getActionDefinitions`**`()` Return the definitions of all actions defined in this class.

All Page Within This Box

AttributeDefinition	**getAttributeDefinition**(java.lang.String id) Return the definition of the attribute with the specified ID.

Methods inherited from interface org.osgi.service.metatype.ObjectClassDefinition
getAttributeDefinitions, getDescription, getIcon, getID, getName

Method Detail

getAttributeDefinition

```
public org.osgi.service.metatype.AttributeDefinition
getAttributeDefinition(java.lang.String id)
```

Return the definition of the attribute with the specified ID.

Parameters:

`id` - The ID of the requested attribute

Returns:

The definition of the attribute or `null` if no such attribute exists

---

getActionDefinitions

```
public ActionDefinition[] getActionDefinitions()
```

Return the definitions of all actions defined in this class.

Returns:

An array of action definitions or `null` if no actions are found

---

getActionDefinition

```
public ActionDefinition getActionDefinition(java.lang.String id)
```

Return the definition of the action with the specified ID.

Parameters:

`id` - The ID of the requested action

Returns:

The definition of the action or `null` if no such action exists

All Page Within This Box

5.2.3 Interface ExtendedAttributeDefinition

All Superinterfaces:
org.osgi.service.metatype.AttributeDefinition
All Known Subinterfaces:
ActionDefinition

---

public interface **ExtendedAttributeDefinition**
extends org.osgi.service.metatype.AttributeDefinition

An interface to describe a complex attribute.

An `ExtendedAttributeDefinition` object defines the description of an attribute whose data type can be complex.

Version:
$Revision: 1.1 $

Field Summary

`Static java.lang.String`	**`HIDDEN`** Key used to indicate if the attribute is hidden or not.
`Static java.lang.String`	**`MAXIMUM`** Key used to store and/or retrieve the maximum value of the attribute from its additional properties.
`Static java.lang.String`	**`MINIMUM`** Key used to store and/or retrieve the minimum value of the attribute from its additional properties.
`static int`	**`OBJECT_CLASS`** The `OBJECT_CLASS`(12) type.
`Static java.lang.String`	**`READ_ONLY`** Key used to indicate if the attribute is read only or not.
`Static java.lang.String`	**`STEP`** Key used to store and/or retrieve the step value of the attribute from its additional properties.

Fields inherited from interface org.osgi.service.metatype.AttributeDefinition
`BIGDECIMAL, BIGINTEGER, BOOLEAN, BYTE, CHARACTER, DOUBLE, FLOAT, INTEGER, LONG, SHORT, STRING`

Method Summary

ExtendedAttributeDefinition`[]`	**`getAttributeDefinitions`**`()` Return the attribute definitions attached to this attribute if the type is OBJECT_CLASS.
`java.io.InputStream`	**`getIcon`**`(int size)` Return an `InputStream` object that can

All Page Within This Box

	be used to create an icon from.
java.util.Dictionary	**getProperties**() Return the additional properties of this attribute.

Methods inherited from interface org.osgi.service.metatype.AttributeDefinition
getCardinality, getDefaultValue, getDescription, getID, getName, getOptionLabels, getOptionValues, getType, validate

Field Detail

OBJECT_CLASS

public static final int **OBJECT_CLASS**

The OBJECT_CLASS(12) type. Attributes of this type should be stored as an object, Vector with objects or an array of these objects depending on getCardinality().

See Also:
Constant Field Values

MINIMUM

public static final java.lang.String **MINIMUM**

Key used to store and/or retrieve the minimum value of the attribute from its additional properties.

See Also:
Constant Field Values

MAXIMUM

public static final java.lang.String **MAXIMUM**

Key used to store and/or retrieve the maximum value of the attribute from its additional properties.

See Also:
Constant Field Values

STEP

public static final java.lang.String **STEP**

Key used to store and/or retrieve the step value of the attribute from its additional properties.

See Also:
Constant Field Values

HIDDEN

public static final java.lang.String **HIDDEN**

Key used to indicate if the attribute is hidden or not.

See Also:
Constant Field Values

READ_ONLY

All Page Within This Box

`public static final java.lang.String` **`READ_ONLY`**

Key used to indicate if the attribute is read only or not.

See Also:

Constant Field Values

Method Detail

getProperties

`public java.util.Dictionary` **`getProperties`**`()`

Return the additional properties of this attribute.

Returns:

Return a dictionary that contains the additional properties or null if there is no additional properties.

---

getAttributeDefinitions

`public ExtendedAttributeDefinition[]` **`getAttributeDefinitions`**`()`

Return the attribute definitions attached to this attribute if the type (as returned by `getType()`) is OBJECT_CLASS.

Returns:

Returns the attribute definitions of the class if this attribute has the type OBJECT_CLASS, null otherwise.

---

getIcon

```
public java.io.InputStream getIcon(int size)
                           throws java.io.IOException
```

Return an `InputStream` object that can be used to create an icon from.

Indicate the size and return an `InputStream` object containing an icon. The returned icon maybe larger or smaller than the indicated size.

The icon may depend on the localization.

Parameters:

`size` - Requested size of an icon, e.g. a 16x16 pixels icon then size = 16

Returns:

An InputStream representing an icon or `null`

Throws:

`java.io.IOException` - if an I/O error occurs

All Page Within This Box

5.2.4 Interface ActionDefinition

All Superinterfaces:

org.osgi.service.metatype.AttributeDefinition, ExtendedAttributeDefinition

---

public interface **ActionDefinition**
extends ExtendedAttributeDefinition

An interface to describe an action.

An `ActionDefinition` object defines a description of an action. An action is very similar to an attribute, it only adds the definition of the input arguments if any.

Version:

$Revision: 1.1 $

Field Summary

Static int	**VOID** The `VOID(13)` type.

Fields inherited from interface org.osgi.service.metatype2.ExtendedAttributeDefinition

HIDDEN, MAXIMUM, MINIMUM, OBJECT CLASS, READ ONLY, STEP

Fields inherited from interface org.osgi.service.metatype.AttributeDefinition

BIGDECIMAL, BIGINTEGER, BOOLEAN, BYTE, CHARACTER, DOUBLE, FLOAT, INTEGER, LONG, SHORT, STRING

Method Summary

ExtendedAttributeDefinition[]	**getInputArgumentDefinitions**() Returns the definitions of the input arguments of this action.

Methods inherited from interface org.osgi.service.metatype2.ExtendedAttributeDefinition

getObjectClassDefinition, getProperties

Methods inherited from interface org.osgi.service.metatype.AttributeDefinition

getCardinality, getDefaultValue, getDescription, getID, getName, getOptionLabels, getOptionValues, getType, validate

Field Detail

All Page Within This Box

VOID

```
public static final int VOID
```

The `VOID(13)` type. The action may have VOID type meaning that it does not have output results.

See Also:

Constant Field Values

Method Detail

getInputArgumentDefinitions

```
public ExtendedAttributeDefinition[] getInputArgumentDefinitions()
```

Returns the definitions of the input arguments of this action.

Returns:

The definitions of the input arguments or null if there is no input arguments.

5.3 MetaType Providing

There are two ways to provide metatype definition in the OSGi framework:

- By registering a `MetaTypeProvider` service in the framework
- By providing a xml resource in a bundle's jar file and describing it in the bundle's manifest

To facilitate users of metatype definitions a `MetaDataService` is introduced. This service can be used to obtain metadata in a uniform way, regardless of the method it is provided in the framework.

The `MetaDataService` identifies metatype definitions provided in the framework by their category and ID. The category is optional and defines the type of meta-data that is provided (for example "ControlUnit", "Config", etc.). The ID is obligatory and should be unique in the scope of the framework.

Along with the `MetaDataService` a `MetaDataListener` interface is introduced. Listeners, implementing this interface, are registered using the White Board model [4] and are notified by the MetaDataService when a metatype definition has appeared, disappeared or was modified.

5.3.1 Providing MetaType with a MetaTypeProvider

One way to provide MetaType information is by registering in the framework a `MetaTypeProvider`, with optional service property `MetaDataService.METATYPE_CATEGORY` equal to the category of the provided meta-data. The service PID of the `MetaTypeProvider` will be considered to be its unique ID in the scope of the OSGi framework.

5.3.2 Providing MetaType with XML Definition

A bundle may provide the metatype definition for its resources (control units) by providing an xml resource in its jar file and describing it in the bundle's manifest using the `Meta-Type` header.

A metatype header must follow the syntax:

```
 Meta-Type: = xml=<path_to_xml>; id=<MetaType ID> [;category=<meta type category>;
][; version=<version>]
```

where `<path_to_xml>` is the path of xml file inside the jar archive, `<MetaType ID>` is the ID, which will be used to identify the metatype definition and must be unique in the scope of the framework. Optionally category (for example "ControlUnit", "Config", etc.), display name and version can be specified.

5.3.2.1 Localization

The `MetaDataService` will support localization of the name, icon, description and label attributes using the same mechanisms described in RFC 74 [5]. The ***localization*** attribute of the ***ExtendedMetaData*** element can be used to specify a different localization file base name. If this attribute is not specified, then the defaults as specified in RFC 74 are used.

The `MetaDataService` must examine the bundle and its fragments to locate all localization resources for the localization base name. From that list, the `MetaDataService` can derive the list of locales which are available for the meta type information. This list can then be returned by `MetaDataService.getMetaTypeLocales` method.

All Page Within This Box

5.3.2.2 MetaType XML Format XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.osgi.org/xmlns/exmetatype/v1.0.0"
xmlns:exmetatype="http://www.osgi.org/xmlns/exmetatype/v1.0.0"
xmlns:metainfo="http://www.osgi.org/xmlns/metatype/v1.0.0"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="unqualified"
attributeFormDefault="unqualified">

    <!-- MetaData root element -->
    <element name="ExtendedMetaData" type="exmetatype:ExMetaDataType"/>
    <complexType name="ExMetaDataType">
        <sequence>
          <!-- attribute definitions -->
          <element name="Attribute" type="exmetatype:ExADType" minOccurs="0"
maxOccurs="unbounded"/>
          <!-- action definitions -->
          <element name="Action" type="exmetatype:ActionType" minOccurs="0"
maxOccurs="unbounded"/>
           <!-- object class definitions -->
          <element name="ObjectClassDefinition" type="exmetatype:ExOCDType"
maxOccurs="unbounded"/>
       </sequence>

       <!-- MetaType localization -->
       <attribute name="localization" type="string" use="optional"/>
    </complexType>

    <!-- ObjectClassDefinition type -->
    <complexType name="ExOCDType">
       <sequence>
          <!-- references to attributes-->
          <element name="AttributeRef" type="exmetatype:RefType" minOccurs="0"
maxOccurs="unbounded"/>
          <!-- references to actions-->
          <element name="ActionRef" type="exmetatype:RefType" minOccurs="0"
maxOccurs="unbounded"/>
          <!-- MetaType icon -->
          <element name="Icon" type="metainfo:Icon" minOccurs="0"/>
       </sequence>

       <!-- the ID of the object class definition -->
       <attribute name="id" type="string" use="required"/>
       <!-- localized name for the object class definition -->
       <attribute name="name" type="string" use="optional"/>
       <!-- localized description for the object class definition -->
       <attribute name="description" type="string" use="optional"/>
    </complexType>

    <!-- Attribute definition type -->
    <complexType name="ExADType">
       <complexContent>
          <extension base="exmetatype:ExADBaseType">
             <!-- the type of the attribute definition -->
            <attribute name="type" type="exmetatype:ExADTypes" use="required"/>
          </extension>
```

```
        </complexContent>
    </complexType>

    <!-- Action type -->
    <complexType name="ActionType">
        <complexContent>
            <extension base="exmetatype:ExADBaseType">
                <sequence>
                    <!-- Defines input argument of the action. If exists should refer
to a previously defined Attribute definition. -->
                    <element name="Argument" type="exmetatype:RefType" minOccurs="0"
maxOccurs="unbounded"/>
                </sequence>
                <!-- (return) type of the action -->
                <attribute name="type" type="exmetatype:ActionTypes" use="required"/>
            </extension>
        </complexContent>
    </complexType>

    <!-- Base for ExtendedAttributeDefinition -->
    <complexType name="ExADBaseType">
        <sequence>
            <!-- if exist (only when "type" attribute of the derived/ExADType &
ActionType/ types has the value "ObjectClass"),  should refer to the "id" field of
a previously defined Attribute definition -->
            <element name="Attribute" type="exmetatype:RefType" minOccurs="0"
maxOccurs="unbounded"/>
            <!-- attribute or action icon -->
            <element name="Icon" type="metainfo:Icon" minOccurs="0"/>
            <!-- Option values and labels -->
            <element name="Option" type="metainfo:Option" minOccurs="0"
maxOccurs="unbounded"/>
            <!-- extended attribute definition properties -->
            <element name="Property" type="exmetatype:PropertyType" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
        <!-- ID by which this element should be refered. Should be used only during
the parse process. -->
        <attribute name="id" type="string" use="required"/>
        <!-- cardinality -->
        <attribute name="cardinality" type="integer" use="optional" default="0"/>
    </complexType>

    <!-- Allowed attribute definition types -->
    <simpleType name="ExADTypes">
        <union memberTypes="metainfo:Scalar">
          <simpleType>
            <restriction base="string">
              <enumeration value="ObjectClass"/>
            </restriction>
          </simpleType>
        </union>
    </simpleType>
```

All Page Within This Box

```
  <!-- Allowed action (return) types -->
  <simpleType name="ActionTypes">
    <union memberTypes="exmetatype:ExADTypes">
      <simpleType>
        <restriction base="string">
          <enumeration value="Void"/>
        </restriction>
      </simpleType>
    </union>
  </simpleType>

  <!-- Reference type. Should supply id and optionally name and description by
itself.
       Refid should refer to another, already defined, object -->
  <complexType name="RefType">
      <!-- the id of the object -->
      <attribute name="id" type="string" use="required"/>
      <!-- the id of the referenced object -->
      <attribute name="refid" type="string" use="required"/>
      <!-- the localized name of the object -->
      <attribute name="name" type="string" use="optional"/>
      <!-- the localized description of the object -->
      <attribute name="description" type="string" use="optional"/>
  </complexType>

  <!-- Property definition type -->
  <complexType name="PropertyType">
      <!-- property key -->
      <attribute name="key" type="string" use="required"/>
      <!-- property value -->
      <attribute name="value" type="string" use="required"/>
  </complexType>

</schema>
```

5.3.2.3 Example MetaType Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<ExtendedMetaData xmlns="http://www.osgi.org/xmlns/exmetatype/v1.0.0"
xmlns:mi="http://www.osgi.org/xmlns/metatype/v1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.osgi.org/xmlns/exmetatype/v1.0.0 metadata.xsd
http://www.osgi.org/xmlns/metatype/v1.0.0 metatype.xsd">

    <!-- BEGIN STATE VAR & PARAM DEFS -->
    <Attribute id="msgDigestProviderIDDef" type="String">
        <Property key="readOnly" value="true"/>
    </Attribute>
    <Attribute id="timeDef" type="Long"/>
    <Attribute id="instanceIDArgDef" type="String"/>
    <Attribute id="byteArrayDef" type="Byte" cardinality="2147483647"/>
    <Attribute id="algorithmDef" type="String"/>
    <!-- END STATE VAR & PARAM DEFS -->

    <!-- BEGIN ACTION DEF -->
    <Action id="getExpireTimeDef" type="Long"/>
```

All Page Within This Box

```
    <Action id="setExpireTimeDef" type="Void">
        <Argument id="expireTime" refid="timeDef"/>
    </Action>
    <Action id="createInstanceDef" type="String">
        <Argument id="algorithm" refid="algorithmDef" name="%algorithmName"
description="%algorithmDescription"/>
    </Action>
    <Action id="releaseInstanceDef" type="Void">
        <Argument id="instanceIDArg" refid="instanceIDArgDef"/>
    </Action>
    <Action id="resetInstanceDef" type="Void">
        <Argument id="instanceIDArg" refid="instanceIDArgDef"/>
    </Action>
    <Action id="checkDigestEqualityDef" type="Boolean">
        <Argument id="bytesArg1" refid="byteArrayDef" name="%bytesArgName1"
description="%bytesArgDescr1"/>
        <Argument id="bytesArg2" refid="byteArrayDef" name="%bytesArgName2"
description="%bytesArgDescr2"/>
    </Action>
    <Action id="updateInstanceDef" type="Void">
        <Argument id="instanceIDArg" refid="instanceIDArgDef"/>
        <Argument id="bytesArg" refid="byteArrayDef" name="%bytesArgName3"
description="%bytesArgDescr3"/>
    </Action>
    <Action id="callDigestOnInstanceDef" type="Byte" cardinality="2147483647">
        <Argument id="generatedMessageDigest" refid="byteArrayDef"
name="%genDigestName" description="%genDigestDescr"/>
    </Action>
    <!-- END ACTION DEF -->

    <!-- BEGIN CU CLASS DEF -->
    <ObjectClassDefinition id="messageDigestService" name="%serviceName"
description="%serviceDescription">
        <!-- State Var Declaration -->
        <AttributeRef id="Provider" refid="msgDigestProviderIDDef" name="%provider"
description="%prvDescription"/>
        <AttributeRef id="expireTime" refid="timeDef" name="%timeName"
description="%timeDescription"/>
        <!-- Action Declaration -->
        <ActionRef id="setExpireTime" refid="setExpireTimeDef"
name="%setExpireTimeName" description="%setExpireTimeDescr"/>
        <ActionRef id="getExpireTime" refid="getExpireTimeDef"
name="%detExpireTimeName" description="%getExpireTimeDescr"/>
        <ActionRef id="createInstance" refid="createInstanceDef"
name="%createInstance" description="%createInstanceDescr"/>
        <ActionRef id="releaseInstance" refid="releaseInstanceDef"
name="%releaseInstance" description="%releaseInstanceDescr"/>
        <ActionRef id="resetInstance" refid="resetInstanceDef"
name="%resetInstanceName" description="%resetInstanceDescr"/>
        <ActionRef id="checkDigestEquality" refid="checkDigestEqualityDef"
name="%checkDigestEqName" description="%checkDigestEqDescr"/>
        <ActionRef id="updateInstance" refid="updateInstanceDef"
name="%updateInstanceName" description="%updateInstanceDescr"/>
        <ActionRef id="callDigestOnInstance" refid="callDigestOnInstanceDef"
name="%callDigestOnInstanceName" description="%callDigestOnInstanceDescr"/>
```

```
        <mi:Icon resource="%mdicon16" size="16"/>
    </ObjectClassDefinition>
    <!-- END CU CLASS DEF -->

</ExtendedMetaData>
```

5.4 The MetaDataService

This service can be used to obtain metatype information provided in the framework - from `MetaTypeProvider` instances or xml resources - in a uniform way. Through it one can obtain a metatype definition by providing a category and an ID, regardless of the method in which the metadata is made available in the framework.

The `MetaDataService` also has the responsibility to track the registered in the framework `MetaDataListener` instances and notify them about appearing, disappearing and modifying of metatype definitions.

5.4.1 MetaDataService API

public interface **MetaDataService**
This service can be used to obtain MetaType information provided in the framework - from `MetaTypeProviders` or xml resources - in an uniform way.
The `MetaDataService` identifies the `MetaTypeProviders` by their category and ID. The category is optional and defines the type of meta-data that is provided (for example "ControlUnit", "Config", etc.). The ID is obligatory and should be unique in the scope of the framework.
One way to provide MetaType information is by registering in the framework a `MetaTypeProvider`, with optional service property `METATYPE_CATEGORY` equal to the category of the provided meta-data. The service PID of the `MetaTypeProvider` will be considered to be its ID. If the `MetaTypeProvider` service has no PID specified its service ID would be taken instead.

Another way is by providing a xml resource in a bundle's jar file and describing it in the bundle's manifest using the `Meta-Type` header.
Note: The xml format is defined in RFC 69 - "Metatyping for the Control Units and Diagnostics".

A metatype header must follow the syntax :

```
 Meta-Type: = xml=<resource>; id=<MetaType ID> [;category=<meta type category>; ][;
version=<version>]
```

For example :

```
  Meta-Type:
   xml=org/osgi/impl/cu/fw/fwcu.xml; category=ControlUnit; id=FRAMEWORK;
version=1.0.0,
   xml=org/osgi/impl/cu/bundle/bundlecu.xml; category=ControlUnit; id=BUNDLE;
version=1.0.0,
   xml=configuration.xml; category=Config; id=cu.config; version=1.0.0
```

If a non-valid MetaType XML is provided the `MetaDataService` will log an error message and ignore the definition.
The MetaType definition will be ignored even if its XML is partially valid.

All Page Within This Box

If a MetaType definition appears, which has the same category/ID pair as an already available MetaType definition, the new one will be ignored and the `MetaDataService` will log an error message.

Through this service one can obtain the MetaType definition by providing a category and an ID, regardless of the method in which the MetaType is made available in the framework.

Field Summary

`Static java.lang.String`	**METATYPE_CATEGORY** Service property identifying a MetaType's category.

Method Summary

`java.lang.String[]`	**getAvailableCatagories**`()` Return all categories for which there is currently available MetaType information provided in the framework.
`java.lang.String[]`	**getAvailableMetaTypes**`(java.lang.String category)` Returns the IDs of all MetaTypes, for the given category, which are currently available in the framework.
`java.lang.String[]`	**getMetaTypeLocales**`(java.lang.String category, java.lang.String id)` Returns the available locales which a given MetaType provides.
`java.lang.String`	**getMetaTypeVersion**`(java.lang.String category, java.lang.String id)` Returns the version of the MetaType with the given id, currently available in the framework for the specified category.
ObjectClassDefinition	**getObjectClassDefinition**`(java.lang.String category, java.lang.String metaTypeID, java.lang.String ocdID, java.lang.String locale)` Returns an object class definition with the specified category and ID localized to the specified locale.

Field Detail

METATYPE_CATEGORY

```
public static final java.lang.String METATYPE_CATEGORY
```

Service property identifying a MetaType's category.

See Also:

Constant Field Values

Method Detail

getObjectClassDefinition

```
public ObjectClassDefinition getObjectClassDefinition(java.lang.String category,
                                                      java.lang.String metaTypeID,
                                                      java.lang.String ocdID,
                                                      java.lang.String locale)
```

All Page Within This Box

```
                                    throws java.lang.IllegalArgumentException
```

Returns an object class definition with the specified category and ID localized to the specified locale.

Parameters:

`category` - The category of the meta-data or null if it has no category.

`metaTypeID` - The ID of the meta-data.

`ocdID` - The ID of the requested object class.

`locale` - The locale of the definition or null for default locale.

Returns:

`ExtendedObjectClassDefinition` or `null` if there is no MetaType provided with the given category and ID.

Throws:

`java.lang.IllegalArgumentException` - If the locale argument is not valid.

`java.lang.NullPointerException` - If the meta-data or object class id is `null`.

---

getAvailableMetaTypes

```
public java.lang.String[] getAvailableMetaTypes(java.lang.String category)
```

Returns the IDs of all MetaTypes, for the given category, which are currently available in the framework.

Parameters:

`category` - The category or `null` for IDs of MetaTypes with no category.

Returns:

Array of meta-type IDs or `null` if there is no MetaType information provided in the framework for the given category.

---

getAvailableCatagories

```
public java.lang.String[] getAvailableCatagories()
```

Return all categories for which there is currently available MetaType information provided in the framework.

Returns:

Array of meta-data categories or `null` if no MetaType in the framework has specified a category.

---

getMetaTypeLocales

```
public java.lang.String[] getMetaTypeLocales(java.lang.String category,
                                             java.lang.String id)
```

Returns the available locales which a given MetaType provides.

Parameters:

`category` - The MetaType category or `null` for no category.

`id` - The MetaType ID for which available locales are requested.

Returns:

Array of locales or `null` if there is no locale specific localization available for the MetaType.

Throws:

`java.lang.NullPointerException` - If the id is `null`.

---

getMetaTypeVersion

```
public java.lang.String getMetaTypeVersion(java.lang.String category,
                                           java.lang.String id)
```

Returns the version of the MetaType with the given id, currently available in the framework for the specified category.

All Page Within This Box

Parameters:
`category` - The MetaType category or `null` for no category.
`id` - The MetaType ID which version is requested.
Returns:
MetaType version, `null` if there is no MetaType provided with the given category and ID or the available MetaType has no version.
Throws:
`java.lang.NullPointerException` - If the id is `null`.

5.4.2 MetaDataListener API

public interface **MetaDataListener**
`MetaDataListeners` are registered as OSGi Services. The `MetaDataService` is responsible for tracking these services and notifying them when a MetaType has been added, removed or modified.

A `MetaDataListener` can narrow the MetaTypes for which events will be received by including in its service registration properties a filter under the key `METATYPE_FILTER`. The value of this property should be a `String` representing LDAP filtering expression. The properties, which may be used in the LDAP filter are `MetaDataService.METATYPE_CATEGORY` and `org.osgi.framework.Constants#SERVICE_PID` (for the MetaType ID).
The listener will be notified only for changes in MetaTypes which category and ID satisfy this filter.
If such property is omitted the listener will receive events for all MetaTypes.

Field Summary

`static int`	**ADDED** Event type which signals that a new MetaType is available.
`static java.lang.String`	**METATYPE_FILTER** `MetaDataListeners` may specify a LDAP filter under this key in their service registration properties to limit the MetaTypes for which to receive events.
`static int`	**MODIFIED** Event type which signals that the corresponding MetaType was modified.
`static int`	**REMOVED** Event type which signals that the corresponding MetaType is no move available.

Method Summary

`void`	**metaDataChanged**`(java.lang.String category, java.lang.String id, int eventType)` Receive a MetaType event.

Field Detail

METATYPE_FILTER

`public static final java.lang.String` **`METATYPE_FILTER`**

All Page Within This Box

`MetaDataListeners` may specify a LDAP filter under this key in their service registration properties to limit the MetaTypes for which to receive events. The value of this property must be a `String` representing a valid LDAP filter.
See Also:
Constant Field Values

ADDED
`public static final int` **`ADDED`**

Event type which signals that a new MetaType is available.
See Also:
Constant Field Values

REMOVED
`public static final int` **`REMOVED`**

Event type which signals that the corresponding MetaType is no move available.
See Also:
Constant Field Values

MODIFIED
`public static final int` **`MODIFIED`**

Event type which signals that the corresponding MetaType was modified.
See Also:
Constant Field Values

Method Detail

metaDataChanged
```
public void metaDataChanged(java.lang.String category,
                            java.lang.String id,
                            int eventType)
```

Receive a MetaType event.
Parameters:
`category` - The category of the MetaType for which event is received or `null` if it has no category.
`id` - The ID of the MetaType for which event is received.
`eventType` - the event type. Possible values are `ADDED`, `REMOVED`, `MODIFIED`.

All Page Within This Box

6 Considered Alternatives

6.1 Extended Attributes and Object Class Definitions

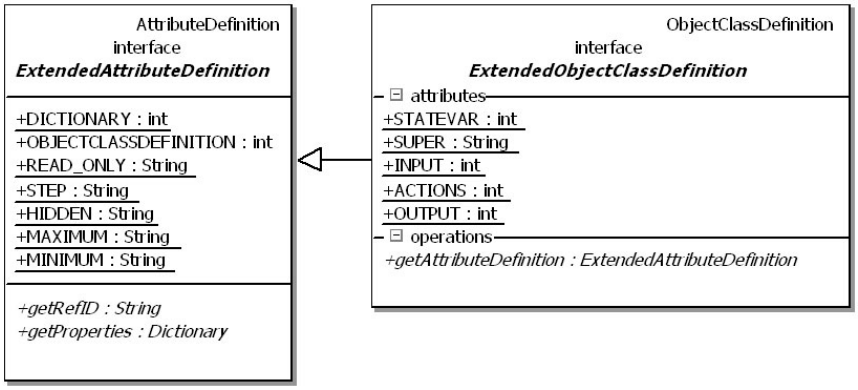


Figure 1. Class Diagram

6.2 Representing Control Unit and Diagnostics Objects

A `ControlUnit` is described if there is registered `org.osgi.service.metatype.MetaTypeProvider` service, with a service PID same as the control unit type.

Another way is with a xml resource in a bundle's jar file, described in the bundle's manifest. The localized, to a specified locale, `ExtendedObjectClassDefinition` can be obtained through the `MetaDataService`. The metatype ID, in this case, should be treated as a TYPE of the Control Unit.

In order to get metatype information it is required to do the following:

- To retrieve a localized version of the metatyping information, the user application queries the `MetaTypeProvider` service for the specified type or receives it through `MetaDataService`.
- To query available state variables, the user application calls `getAttributeDefinitions(STATEVAR)` of the `ExtendedObjectClassDefinition`, and the implementation of the `ObjectClassDefinition` must give the user a full set of `AttributeDefinition` objects, describing unit state variables.
- To list the available actions, the user obtains a localized `ExtendedObjectClassDefinition` from the `MetaTypeProvider` service and uses its `getActionDefinitions()` method to obtain meta data information about all available actions.
- To query action attributes, from the `ActionDefinition` object for the action, the user uses `get getInputArgumentDefinitions()` method to obtain `ExtendedAttributeDefinition` for the input arguments or `getAttributeDefinitions()` method for the output arguments respectively.
- To obtain extended information about State Variables the developer may check if the returned Object implements `ExtendedAttributeDefinition` interface and eventually perform a type cast.

All Page Within This Box

6.3 Javadoc

6.3.1 org.osgi.service.metatype2 Interface ExtendedObjectClassDefinition

All Superinterfaces:

org.osgi.service.metatype.AttributeDefinition, ExtendedAttributeDefinition, org.osgi.service.metatype.ObjectClassDefinition

public interface **ExtendedObjectClassDefinition**
extends org.osgi.service.metatype.ObjectClassDefinition, ExtendedAttributeDefinition

Description for the data type information of an objectclass.

Field Summary	
static int	**ACTIONS** Argument for `getAttributeDefinitions(int)`.
static int	**INPUT** Argument for `getAttributeDefinitions(int)`.
static int	**OUTPUT** Argument for `getAttributeDefinitions(int)`.
static int	**STATEVAR** Argument for `getAttributeDefinitions(int)`.
static java.lang.String	**SUPER** Constant for attribute definition ID.

Fields inherited from interface org.osgi.service.metatype.ObjectClassDefinition
ALL, OPTIONAL, REQUIRED

Fields inherited from interface org.osgi.service.metatype2.
DICTIONARY, HIDDEN, MAXIMUM, MINIMUM, OBJECTCLASSDEFINITION, READ ONLY, STEP

Fields inherited from interface org.osgi.service.metatype.AttributeDefinition
BIGDECIMAL, BIGINTEGER, BOOLEAN, BYTE, CHARACTER, DOUBLE, FLOAT, INTEGER, LONG, SHORT, STRING

Method Summary	
ExtendedAttributeDefinition	**getAttributeDefinition**(java.lang.String id) Return the attribute definition for the specified attribute ID.

Methods inherited from interface org.osgi.service.metatype.ObjectClassDefinition
getAttributeDefinitions, getDescription, getIcon, getID, getName

All Page Within This Box

Methods inherited from interface org.osgi.service.metatype2.
getProperties, getRefID

Methods inherited from interface org.osgi.service.metatype.AttributeDefinition
getCardinality, getDefaultValue, getDescription, getID, getName, getOptionLabels, getOptionValues, getType, validate

Field Detail

INPUT

`public static final int` **`INPUT`**

Argument for `getAttributeDefinitions(int). INPUT` indicates that only definitions that may be used as input data are returned. The value is 3.

See Also:
Constant Field Values

OUTPUT

`public static final int` **`OUTPUT`**

Argument for `getAttributeDefinitions(int). OUTPUT` indicates that only definitions that may be used as output data are returned. The value is 4.

See Also:
Constant Field Values

STATEVAR

`public static final int` **`STATEVAR`**

Argument for `getAttributeDefinitions(int). STATEVAR` indicates that only definitions that represent State Variables are returned. The value is 5.

See Also:
Constant Field Values

ACTIONS

`public static final int` **`ACTIONS`**

Argument for `getAttributeDefinitions(int). ACTIONS` indicates that only definitions that represent Actions are returned. The value is 6.

See Also:
Constant Field Values

SUPER

All Page Within This Box

`public static final java.lang.String` **`SUPER`**

Constant for attribute definition ID. If the object class definition has such an attribute, then it is an extension to another object class definition. The type of the attribute is `STRING`, cardinality is 0 and its default value (returned by `AttributeDefinition.getDefaultValue()`) indicated the PID of the extended object class definition.

See Also:

Constant Field Values

Method Detail

getAttributeDefinition

`public ExtendedAttributeDefinition` **`getAttributeDefinition`**`(java.lang.String id) throws java.lang.IllegalArgumentException`

Returns the attribute definition for the specified attribute ID.

Parameters:

`id` - The ID of the requested attribute.

Returns:

An ExtendedAttributeDefinition object for the requested ID.

Throws:

`java.lang.IllegalArgumentException` - If no attribute with the specified ID exists for this object class.

skip-navbar_bottom

6.3.2 org.osgi.service.metatype2

Interface ExtendedAttributeDefinition

All Superinterfaces:

org.osgi.service.metatype.AttributeDefinition

All Known Subinterfaces:

ExtendedObjectClassDefinition

public interface **ExtendedAttributeDefinition**
extends org.osgi.service.metatype.AttributeDefinition

An interface to describe an attribute.

An `ExtendedAttributeDefinition` object defines a description of the data type of a property/attribute.

Field Summary

`static int`	**`DICTIONARY`** The DICTIONARY (13) type.
`static java.lang.String`	**`HIDDEN`** This constant defines a key for attribute extension properties.
`static java.lang.String`	**`MAXIMUM`** This constant defines a key for attribute extension properties.
`static java.lang.String`	**`MINIMUM`** This constant defines a key for attribute extension properties.
`static int`	**`OBJECTCLASSDEFINITION`**

All Page Within This Box

	The OBJECTCLASSDEFINITION (12) type.
static java.lang.String	**READ_ONLY** This constant defines a key for attribute extension properties.
static java.lang.String	**STEP** This constant defines a key for attribute extension properties.

Fields inherited from interface org.osgi.service.metatype.AttributeDefinition
BIGDECIMAL, BIGINTEGER, BOOLEAN, BYTE, CHARACTER, DOUBLE, FLOAT, INTEGER, LONG, SHORT, STRING

Method Summary

java.util.Dictionary	**getProperties**() Returns the properties of this attribute.
java.lang.String	**getRefID**() Returns the ID of an object class if the type of this attribute is OBJECTCLASSDEFINITION.

Methods inherited from interface org.osgi.service.metatype.AttributeDefinition
getCardinality, getDefaultValue, getDescription, getID, getName, getOptionLabels, getOptionValues, getType, validate

Field Detail

OBJECTCLASSDEFINITION

public static final int **OBJECTCLASSDEFINITION**

The OBJECTCLASSDEFINITION (12) type. Attributes of this type are ObjectClassDefinition. This attribute type is not supported by configurations in ConfigurationAdmin.

See Also:
Constant Field Values

DICTIONARY

public static final int **DICTIONARY**

The DICTIONARY (13) type. Attributes of this type should be stored as Dictionary objects, where keys are of type String and values are of the only types definied by the standard metatype API. The method getCardinality() value MUST always return 0.

See Also:
Constant Field Values

MINIMUM

All Page Within This Box

`public static final java.lang.String` **`MINIMUM`**

This constant defines a key for attribute extension properties. The property key is "minValue" and the property value should be a string representation of the minimum value acceptable for the attribute.

See Also:

Constant Field Values

MAXIMUM

`public static final java.lang.String` **`MAXIMUM`**

This constant defines a key for attribute extension properties. The property key is "maxValue" and the property value should be a string representation of the maximum value acceptable for the attribute.

See Also:

Constant Field Values

STEP

`public static final java.lang.String` **`STEP`**

This constant defines a key for attribute extension properties. The property key is "stepValue" and the property value should be a string representation of the value that is acceptable for increment/decrement operations over the attribute.

See Also:

Constant Field Values

HIDDEN

`public static final java.lang.String` **`HIDDEN`**

This constant defines a key for attribute extension properties. The property key is "hidden" and the property value should be a string. If the property value is equal, ignoring case, to the string "true", then any visual attribute editors must not display the attribute value. This property is useful for passwords.

See Also:

Constant Field Values

READ_ONLY

`public static final java.lang.String` **`READ_ONLY`**

This constant defines a key for attribute extension properties. The property key is "readOnly" and the property value should be a string. If the property value is equal, ignoring case, to the string "true", then any visual attribute editors must not allow means to directly alter the attribute value.

See Also:

Constant Field Values

Method Detail

getProperties

`public java.util.Dictionary` **`getProperties`**`()`

Returns the properties of this attribute. The properties have case sensitive keys and cannot be modified.

Returns:

The properties for this attribute.

All Page Within This Box

getRefID

`public java.lang.String` **`getRefID`**`()`

Returns the ID of an object class if the type of this attribute is OBJECTCLASSDEFINITION.

Returns:

ID of the referenced object class or `null` if the type of this attribute is not OBJECTCLASSDEFINITION.

6.3.3 Constant Field Values

Contents

- org.osgi.*

org.osgi.*

org.osgi.service.metatype2.

public static final int	DICTIONARY	13
public static final java.lang.String	HIDDEN	"hidden"
public static final java.lang.String	MAXIMUM	"maxValue"
public static final java.lang.String	MINIMUM	"minValue"
public static final int	OBJECTCLASSDEFINITION	12
public static final java.lang.String	READ_ONLY	"readOnly"
public static final java.lang.String	STEP	"stepValue"

org.osgi.service.metatype2.

public static final int	ACTIONS	6
public static final int	INPUT	3
public static final int	OUTPUT	4
public static final int	STATEVAR	5
public static final java.lang.String	SUPER	"super"

7 Security Considerations

No special security requirement. The services that receive meta information must have appropriate bundle permission in order to have access to it.

All Page Within This Box

8 Document Support

8.1 References

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

[2]. Software Requirements & Specifications. Michael Jackson. ISBN 0-201-87712-0

[3]. RFP 47 – Control Unit, Prosyst, August 2003

[4]. Technical Whitepaper: Listeners Considered Harmful: The "Whiteboard" Pattern

[5]. RFC 74 - Manifest Localization

8.2 Author's Address

Name	Pavlin Dobrev
Company	ProSyst Software AG
Address	D-50858 Cologne, Germany . Dürener Strasse 405
Voice	+49 (0)221 6604 0
e-mail	p.dobrev@prosyst.com

Name	Pavé Olivier
Company	Siemens VDO Automotive
Address	Batiment Alpha 80, route des lucioles – BP 305 06906 Sophia-Antipolis Cedex, France
Voice	+33 (0)4-9238-1129
e-mail	olivier.pave@siemens.com

8.3 Acronyms and Abbreviations

8.4 End of Document

All Page Within This Box