OCF “Ipanema” – Clarify Mandatory Common Properties "rt" and "if" and Relationship (view) on the Wire – Core Technology WG CR 3184

7 Resource model

7.1 Introduction

The Resource model defines concepts and mechanisms that provide consistency and core interoperability between Devices in the OCF ecosystems. The Resource model concepts and mechanisms are then mapped to the transport protocols to enable communication between the Devices – each transport provides the communication protocol interoperability. The Resource model, therefore, allows for interoperability to be defined independent of the transports.

The primary concepts in the Resource model are: entity, Resources, Uniform Resource Identifiers (URI), Resource Types, Properties, Representations, OCF Interfaces, Collections and Links. In addition, the general mechanisms are CREATE, RETRIEVE, UPDATE, DELETE and NOTIFY. These concepts and mechanisms may be composed in various ways to define the rich semantics and interoperability needed for a diverse set of use cases that the Framework is applied to.

In the OCF Resource model Framework, an entity needs to be visible, interacted with or manipulated, it is represented by an abstraction called a Resource. A Resource encapsulates and represents the state of an entity. A Resource is identified, addressed and named using URIs.

Properties are "key=value" pairs and represent state of the Resource. A snapshot of these Properties is the Representation of the Resource. A specific view of the Representation and the mechanisms applicable in that view are specified as OCF Interfaces. Interactions with a Resource are done as Requests and Responses containing Representations.

A Resource instance is derived from a Resource Type. The uni-directional relationship between one Resource and another Resource is defined as a Link. A Resource that has Properties and Links is a Collection.

A set of Properties can be used to define a state of a Resource. This state may be retrieved or updated using appropriate Representations respectively in the response from and request to that Resource.

A Resource (and Resource Type) could represent and be used to expose a capability. Interactions with that Resource can be used to exercise or use that capability. Such capabilities can be used to define processes like discovery, management, advertisement etc. For example: *discovery of Resources on a Device* can be defined as the retrieval of a representation of a specific Resource where a Property or Properties have values that describe or reference the Resources on the Device.

The information for Request or Response with the Representation may be communicated on the wire by serializing using a transfer protocol or encapsulated in the payload of the transport protocol – the specific method is determined by the normative mapping of the Request or Response to the transport protocol. See 12 for transport protocols supported.

The OpenAPI 2.0 definitions (Annex A) used in this document are normative. This includes that all defined JSON payloads shall comply with the indicated OpenAPI 2.0 definitions. Annex A contains all of the OpenAPI 2.0 definitions for Resource Types defined in this document.

7.2 Resource

A Resource shall be defined by one or more Resource Type(s) – see Annex A for Resource Type. A request to CREATE a Resource shall specify one or more Resource Types that define that Resource.

A Resource is hosted in a Device. A Resource shall have a URI as defined in clause 6. The URI may be assigned by the Authority at the creation of the Resource or may be pre-defined by the definition of the Resource Type. An example Resource representation is depicted in Figure 4.

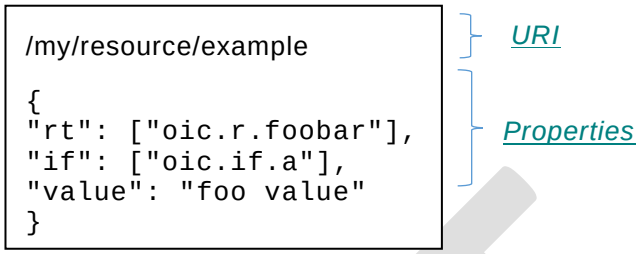

Figure 1 – Example Resource

Core Resources are the Resources defined in this document to enable functional interactions as defined in clause 10 (e.g., Discovery, Device management, etc). Among the Core Resources, "/oic/res", "/oic/p", and "/oic/d" shall be supported on all Devices. Devices may support other Core Resources depending on the functional interactions they support.

7.3 Property

7.3.1 Introduction

A Property describes an aspect that is exposed through a Resource including meta-information related to that Resource.

A Property shall have a name i.e. Property Name and a value i.e. Property Value. The Property is expressed as a key-value pair where key is the Property Name and value the Property Value like <Property Name> = <Property Value>. For example if the "temperature" Property has a Property Name "temp" and a Property Value "30F", then the Property is expressed as "temp=30F". The specific format of the Property depends on the encoding scheme. For example, in JSON, Property is represented as "key": value (e.g., "temp": 30).

In addition, the Property definition shall have a

– *Value Type* – the Value Type defines the values that a Property Value may take. The Value Type may be a simple data type (e.g. string, Boolean) as defined in 4.3 or may be a complex data type defined with a schema. The Value Type may define
  – Value Rules define the rules for the set of values that the Property Value may take. Such rules may define the range of values, the min-max, formulas, the set of enumerated values, patterns, conditional values, and even dependencies on values of other Properties. The rules may be used to validate the specific values in a Property Value and flag errors.
– *Mandatory* – specifies if the Property is mandatory or not for a given Resource Type.
– *Access modes* – specifies whether the Property may be read, written or both. Updates are equivalent to a write. "r" is used for read and "w" is used for write – both may be specified. Write does not automatically imply read.

The definition of a Property may include the following additional information – these items are informative:

– *Property Title* - a human-friendly name to designate the Property; usually not sent over the wire.

– *Description* – descriptive text defining the purpose and expected use of this Property.

In general, a Property is meaningful only within the Resource to which it is associated. However a base set of Properties that may be supported by all Resources, known as Common Properties, keep their semantics intact across Resources i.e. their "key=value" pair means the same in any Resource. Detailed tables for all Common Properties are defined in 7.3.2.

7.3.2 Common Properties

7.3.2.1 Introduction

The mandatory Common Properties defined in clause 7.3.2 shall be exposed and the optional Common Properties may be exposed in all Resources. The following Common Properties for all Resources are specified in 7.3.2.3 through 7.3.2.6 respectively and summarized as follows:

– *Resource Type* ("rt") – this mandatory Property is used to declare the Resource Type of that Resource. Since a Resource could be defined by more than one Resource Type the Property Value of the Resource Type Property may be used to declare more than one Resource Type (see clause 7.4.4). See 7.3.2.3 for details.

– *OCF Interface* ("if") – this mandatory Property declares the OCF Interfaces supported by the Resource. The Property Value of the OCF Interface Property may be multi-valued and lists all the OCF Interfaces supported. See 7.3.2.4 for details.

– *Name* ("n") – this optional Property declares human-readable name assigned to the Resource. See 7.3.2.5.

– *Resource Identity* ("id") – this optional Property Value shall be a unique (across the scope of the host Server) identifier for a specific instance of the Resource. The encoding of this identifier is Device and implementation dependent. See 7.3.2.6 for details.

An optional Common Property may be mandatory when explicitly specified in a particular Resource Type definition (e.g., the "n" Common Property for the "oic.wk.d" Resource Type).

The name of a Common Property is unique and is not used by other Properties. When defining a new Resource Type, its non-common Properties will not use the name of existing Common Properties (e.g., "rt", "if", "n", and "id").

The ability to UPDATE a Common Property (that supports write as an access mode) is restricted to the "oic.if.rw" (read-write) OCF Interface; thus a Common Property shall be updatable using the read-write OCF Interface if and only if the Property supports write access as defined by the Property definition and the associated schema for the read-write OCF Interface.

7.3.2.2 Property Name and Property Value definitions

The Property Name and Property Value as used in this document:

– *Property Name*– the key in "key=value" pair. Property Name is case sensitive and its data type is "string". Property names shall contain only letters A to Z, a to z, digits 0 to 9, hyphen, and dot, and shall not begin with a digit.

– *Property Value* – the value in "key=value" pair. Property Value is case sensitive when its data type is "string".

7.3.2.3 Resource Type

Resource Type Property is specified in 7.4.

7.3.2.4 OCF Interface

OCF Interface Property is specified in 7.6.

7.3.2.5 Name

A human friendly name for the Resource, i.e. a specific resource instance name (e.g., MyLivingRoomLight), The Name Property is as defined in Table 2

Table 1 – Name Property Definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Name	"n"	"string"	N/A	N/A	R, W	No	Human understandable name for the Resource.

This Property may be mandatory when specifically defined for a Resource Type (e.g., "oic.wk.d").

The Name Property is read-write unless otherwise restricted by the Resource Type (i.e. the Resource Type does not support UPDATE or does not support UPDATE using the read-write OCF Interface ("oic.if.rw")).

7.3.2.6 Resource Identity

The Resource Identity Property shall be a unique (across the scope of the host Server) instance identifier for a specific instance of the Resource. The encoding of this identifier is Device and implementation dependent as long as the uniqueness constraint is met, noting that an implementation may use a uuid as defined in 4.3. The Resource Identity Property is as defined in Table 3.

Table 2 – Resource Identity Property Definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Resource Identity	"id"	"string" or uuid	Implementation Dependent	N/A	R	No	Unique identifier of the Resource (over all Resources in the Device)

This Property may be mandatory when specifically defined for a Resource Type.

7.4 Resource Type

7.4.1 Introduction

Resource Type is a class or category of Resources and a Resource is an instance of one or more Resource Types.

The Resource Types of a Resource is declared using the Resource Type Common Property as described in 7.3.2.3 or in a Link using the Resource Type Parameter.

A Resource Type may either be pre-defined by OCF or in custom definitions by manufacturers, end users, or developers of Devices (vendor-defined Resource Types). Resource Types and their definition details may be communicated out of band (i.e. in documentation) or be defined explicitly using a meta-language which may be downloaded and used by APIs or applications. OCF has adopted OpenAPI 2.0 as the specification method for OCF's RESTful interfaces and Resource definitions.

Every Resource Type shall be identified with a Resource Type ID which shall be represented using the requirements and ABNF governing the Resource Type attribute in IETF RFC 6690 (clause 2 for ABNF and clause 3.1 for requirements) with the caveat that segments are separated by a "." (period). The entire string represents the Resource Type ID. When defining the ID each segment may represent any semantics that are appropriate to the Resource Type. For example, each segment could represent a namespace. Once the ID has been defined, the ID should be used opaquely and implementations should not infer any information from the individual segments. The string "oic", when used as the first segment in the definition of the Resource Type ID, is reserved for OCF-defined Resource Types. All OCF defined Resource Types are to be registered with the IANA Core Parameters registry as described also in IETF RFC 6690.

7.4.2 Resource Type Property

A Resource when instantiated or created shall have one or more Resource Types that are the template for that Resource. The Resource Types that the Resource conforms to shall be declared using the "rt" Common Property for the Resource as defined in Table 4. The Property Value for the "rt" Common Property shall be the list of Resource Type IDs for the Resource Types used as templates (i.e., "rt"=<list of Resource Type IDs>).

Table 3 – Resource Type Common Property definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Resource Type	"rt"	"array"	Array of strings, conveying Resource Type IDs	N/A	R	Yes	The Property name rt is as described in IETF RFC 6690

Resource Types may be explicitly discovered or implicitly shared between the user (i.e. Client) and the host (i.e. Server) of the Resource.

7.4.3 Resource Type definition

Resource Type is specified as follows:

– *Pre-defined URI* (optional) – a pre-defined URI may be specified for a specific Resource Type in an OCF specification. When a Resource Type has a pre-defined URI, all instances of that Resource Type shall use only the pre-defined URI. An instance of a different Resource Type shall not use the pre-defined URI.

– *Resource Type Title* (optional) – a human friendly name to designate the Resource Type**.**

– *Resource Type ID* – the value of "rt" Property which identifies the Resource Type, (e.g., "oic.wk.p").

– *Resource Interfaces* – list of the OCF Interfaces that may be supported by the Resource Type.

– *Properties* – definition of all the Properties that apply to the Resource Type. The Resource Type definition shall define whether a property is mandatory, conditional mandatory, or optional.

– *Related Resource Types* (optional) – the definition of other Resource Types that may be referenced as part of the Resource Type, applicable to Collections.

– *Mime Types* (optional) – mime types supported by the Resource including serializations (e.g., application/cbor, application/json, application/xml).

Table 5 and Table 6 provides an example description of an illustrative foobar Resource Type and its associated Properties.

Table 4 – Example foobar Resource Type

Pre-defined URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction	M/CR/O
none	"foobar"	"oic.r.foobar"	"oic.if.a"	Example "foobar" Resource	Actuation	O

Table 5 – Example foobar Properties

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Resource Type	"rt"	"array"	N/A	N/A	R	Yes	Resource Type
OCF Interface	"if"	"array"	N/A	N/A	R	Yes	OCF Interface
Foo value	value	"string"	N/A	N/A	R	Yes	Foo value

For example, an instance of the foobar Resource Type.

```
{
"rt": ["oic.r.foobar"],
"if": ["oic.if.a"],
"value": "foo value"
}
```

For example, a schema representation for the foobar Resource Type.

```
{
  "$schema": "http://json-schema.org/draft-04/schema",
  "type": "object",
  "properties": {
    "rt": {
      "type": "array",
      "items" : {
        "type" : "string",
        "maxLength": 64
      },
      "minItems" : 1,
      "readOnly": true,
      "description": "Resource Type of the Resource"
    },
    "if": {
      "type": "array",
      "items": {
        "type" : "string",
        "enum" : ["oic.if.baseline", "oic.if.ll", "oic.if.b", "oic.if.lb",
"oic.if.rw", "oic.if.r", "oic.if.a", "oic.if.s"]
    },
    "value": {"type": "string"}
```

```
  },
  "required": ["rt", "if", "value"]
}
```

7.4.4 Multi-value "rt" Resource

Multi-value "rt" Resource means a Resource with multiple Resource Types where none of the included Resource Types denote a well-known Resource Type (i.e. "oic.wk.<thing>"). Such a Resource is associated with multiple Resource Types and so has an "rt" Property Value of multiple Resource Type IDs (e.g. "rt": ["oic.r.switch.binary", "oic.r.light.brightness"]). The order of the Resource Type IDs in the "rt" Property Value is meaningless. For example, "rt": ["oic.r.switch.binary", "oic.r.light.brightness"] and "rt": ["oic.r.light.brightness", "oic.r.switch.binary"] have the same meaning.

Resource Types for multi-value "rt" Resources shall satisfy the following conditions:

– Property Name – Property Names for each Resource Type shall be unique (within the scope of the multi-value "rt" Resource) with the exception of Common Properties, otherwise there will be conflicting Property semantics. If two Resource Types have a Property with the same Property "Name, a multi-value "rt" Resource shall not be composed of these Resource Types.

A multi-value "rt" Resource satisfies all the requirements for each Resource Type and conforms to the OpenAPI 2.0 definitions for each component Resource Type. Thus the mandatory Properties of a multi-value "rt" Resource shall be the union of all the mandatory Properties of each Resource Type. For example, mandatory Properties of a Resource with "rt": ["oic.r.switch.binary", "oic.r.light.brightness"] are "value" and "brightness", where the former is mandatory for "oic.r.switch.binary" and the latter for "oic.r.light.brightness".

The multi-value "rt" Resource Interface set shall be the union of the sets of OCF Interfaces from the component Resource Types. The Resource Representation in response to a CRUDN action on an OCF Interface shall be the union of the schemas that are defined for that OCF Interface. The Default OCF Interface for a multi-value "rt" Resource shall be the baseline OCF Interface ("oic.if.baseline") as that is the only guaranteed common OCF Interface between the Resource Types.

For clarity if each Resource Type supports the same set of OCF Interfaces, then the resultant multi-value "rt" Resource has that same set of OCF Interfaces with a Default OCF Interface of baseline ("oic.if.baseline").

See 7.9.3 for the handling of query parameters as applied to a multi-value "rt" Resource.

7.5 Device Type

A Device Type is a class of Device. Each Device Type defined will include a list of minimum Resource Types that a Device shall implement for that Device Type. A Device may expose additional standard and vendor defined Resource Types beyond the minimum list. The Device Type is used in Resource discovery as specified in 11.2.3.

Like a Resource Type, a Device Type can be used in the Resource Type Common Property or in a Link using the Resource Type Parameter.

A Device Type may either be pre-defined by an ecosystem that builds on this document, or in custom definitions by manufacturers, end users, or developers of Devices (vendor-defined Device Types). Device Types and their definition details may be communicated out of band (like in documentation).

Every Device Type shall be identified with a Resource Type ID using the same syntax constraints as a Resource Type.

7.6 OCF Interface

7.6.1 Introduction

An OCF Interface provides first a view into the Resource and then defines the requests and responses permissible on that view of the Resource. So this view provided by an OCF Interface defines the context for requests and responses on a Resource. Therefore, the same request to a Resource when targeted to different OCF Interfaces may result in different responses. Depending on the view requested (i.e., OCF Interface), the Resource representation may not include all mandatory Properties (e.g., the "rt" and "if" Common Properties). If Common Properties are desired in the view requested, use the "oic.if.baseline" OCF Interface (see clause 7.6.3.2) which every Resource Type shall implement.

An OCF Interface may be defined by either this document (a Core OCF Interface), manufacturers, end users or developers of Devices (a vendor-defined OCF Interface).

The OCF Interface Property lists all the OCF Interfaces the Resource support. All Resources shall have at least one OCF Interface. The Default OCF Interface shall be defined by the Resource Type definition. The Default OCF Interface associated with all OCF-defined Resource Types shall be the supported OCF Interface listed first within the *applicable enumeration* in the definition of the Resource Type (see Annex A for the OCF-defined Resource Types defined in this document). The *applicable enumeration* is in the "parameters" enumeration referenced from the first "get" method in the first "path" in the OpenAPI 2.0 file ("post" method if no "get" exists) for the Resource Type. All Default OCF Interfaces specified in an OCF specification shall be mandatory.

In addition to any defined OCF Interface in this document, all Resources shall support the baseline OCF Interface ("oic.if.baseline") as defined in 7.6.3.2.

See 7.9.4 for the use of queries to enable selection of a specific OCF Interface in a request.

An OCF Interface may accept more than one media type. An OCF Interface may respond with more than one media type. The accepted media types may be different from the response media types. The media types are specified with the appropriate header parameters in the transfer protocol. (NOTE: This feature has to be used judiciously and is allowed to optimize representations on the wire) Each OCF Interface shall have at least one media type.

7.6.2 OCF Interface Property

The OCF Interfaces supported by a Resource shall be declared using the OCF Interface Common Property (Table 7), e.g., ""if": ["oic.if.ll", "oic.if.baseline"]". The Property Value of an OCF Interface Property shall be a lower case string with segments separated by a "." (dot). The string "oic", when used as the first segment in the OCF Interface Property Value, is reserved for OCF-defined OCF Interfaces. The OCF Interface Property Value may also be a reference to an authority similar to IANA that may be used to find the definition of an OCF Interface. A Resource Type shall support one or more of the OCF Interfaces defined in 7.6.3.

Table 6 – Resource Interface Property definition

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
OCF **Interface**	"if"	"array"	Array of strings, conveying OCF Interfaces	N/A	R	Yes	Property to declare the OCF Interfaces supported by a Resource.

7.6.3 OCF Interface methods

7.6.3.1 Overview

OCF Interface methods shall not violate the defined OpenAPI 2.0 definitions for the Resources as defined in Annex A.

The defined OCF Interfaces are listed in Table 8:

Table 7 – OCF standard OCF Interfaces

OCF Interface	Name	Applicable Operations	Description
baseline	"oic.if.baseline"	RETRIEVE, NOTIFY, UPDATE[1]	The baseline OCF Interface defines a view into all Properties of a Resource including the Meta Properties. This OCF Interface is used to operate on the full Representation of a Resource.
links list	"oic.if.ll"	RETRIEVE, NOTIFY	The links list OCF Interface provides a view into Links in a Collection (Resource). Since Links represent relationships to other Resources, the links list OCF Interfaces may be used to discover Resources with respect to a context. The discovery is done by retrieving Links to these Resources. For example: the Core Resource "/oic/res" uses this OCF Interface to allow discovery of Resource hosted on a Device.
batch	"oic.if.b"	RETRIEVE, NOTIFY, UPDATE	The batch OCF Interface is used to interact with a Collection of Resources at the same time. This also removes the need for the Client to first discover the Resources it is manipulating – the Server forwards the requests and aggregates the responses
read-only	"oic.if.r"	RETRIEVE NOTIFY	The read-only OCF Interface exposes the Properties of a Resource that may be read. This OCF Interface does not provide methods to update Properties, so can only be used to read Property Values.
read-write	"oic.if.rw"	RETRIEVE, NOTIFY, UPDATE	The read-write OCF Interface exposes only those Properties that may be read from a Resource during a RETRIEVE operation and only those Properties that may be written to a Resource during and UPDATE operation.
actuator	"oic.if.a"	RETRIEVE, NOTIFY, UPDATE	The actuator OCF Interface is used to read or write the Properties of an actuator Resource.
sensor	"oic.if.s"	RETRIEVE, NOTIFY	The sensor OCF Interface is used to read the Properties of a sensor Resource.
create	"oic.if.create"	CREATE	The create OCF Interface is used to create new Resources in a Collection. Both the Resource and the Link pointing to it are created in a single atomic operation.

[1] The use of UPDATE with the baseline OCF Interface is not recommended, see clause 7.6.3.2.3.

7.6.3.2 Baseline OCF Interface

7.6.3.2.1 Overview

The Representation that is visible using the baseline OCF Interface includes all the Properties of the Resource including the mandatory and implemented optional Common Properties. The baseline OCF Interface shall be defined for all Resource Types. All Resources shall support the baseline OCF Interface.

7.6.3.2.2 Use of RETRIEVE

The baseline OCF Interface is used when a Client wants to retrieve all Properties of a Resource; that is the Server shall respond with a Resource representation that includes all of the implemented Properties of the Resource. When the Server is unable to send back the whole Resource representation, it shall reply with an error message. The Server shall not return a partial Resource representation.

An example response to a RETRIEVE request using the baseline OCF Interface:

```
{
"rt": ["oic.r.temperature"],
"if": ["oic.if.a","oic.if.baseline"],
"temperature": 20,
"units": "C",
"range": [0,100]
}
```

7.6.3.2.3 Use of UPDATE

Support for the UPDATE operation using the baseline OCF Interface should not be provided by a Resource Type. Where a Resource Type needs to support the ability to be UPDATED this should only be supported using one of the other OCF Interfaces defined in Table 8 that supports the UPDATE operation.

If a Resource Type is required to support UPDATE using the baseline OCF Interface, then all Properties of a Resource with the exception of Common Properties may be modified using an UPDATE operation only if the Resource Type defines support for UPDATE using baseline in the applicable OpenAPI 2.0 schema for the Resource Type. If the OCF Interfaces exposed by a Resource in addition to the baseline OCF Interface do not support the UPDATE operation, then UPDATE using the baseline OCF Interface shall not be supported.