GWD-R
OCCI-WG

Ralf Nyrén
Florian Feldhaus, GWDG
February 25, 2011
Updated: January 18, 2014

Open Cloud Computing Interface - JSON Rendering

Status of this Document

This document provides information to the community regarding the specification of the Open Cloud Computing Interface. Distribution is unlimited.

Copyright Notice

Trademarks

OCCI is a trademark of the Open Grid Forum.

Abstract

This document, part of a document series, produced by the OCCI working group within the Open Grid Forum (OGF), provides a high-level definition of a Protocol and API. The document is based upon previously gathered requirements and focuses on the scope of important capabilities required to support modern service offerings.

Contents

1 Introduction

The Open Cloud Computing Interface (OCCI) is a RESTful Protocol and API for all kinds of management tasks. OCCI was originally initiated to create a remote management API for IaaS[1] model-based services, allowing for the development of interoperable tools for common tasks including deployment, autonomic scaling and monitoring. It has since evolved into a flexible API with a strong focus on interoperability while still offering a high degree of extensibility. The current release of the Open Cloud Computing Interface is suitable to serve many other models in addition to IaaS, including PaaS and SaaS.

In order to be modular and extensible the current OCCI specification is released as a suite of complimentary documents, which together form the complete specification. The documents are divided into three categories consisting of the OCCI Core, the OCCI Renderings and the OCCI Extensions.

- The OCCI Core specification consists of a single document defining the OCCI Core Model. The OCCI Core Model can be interacted with *renderings* (including associated behaviours) and expanded through *extensions*.
- The OCCI Rendering specifications consist of multiple documents each describing a particular rendering of the OCCI Core Model. Multiple renderings can interact with the same instance of the OCCI Core Model and will automatically support any additions to the model which follow the extension rules defined in OCCI Core.
- The OCCI Extension specifications consist of multiple documents each describing a particular extension of the OCCI Core Model. The extension documents describe additions to the OCCI Core Model defined within the OCCI specification suite. They do not require changes to the HTTP Rendering specifications as of this version of the specification.

The current specification consists of three documents. This specification describes version 1.1 of OCCI. Future releases of OCCI may include additional rendering and extension specifications. The documents of the current OCCI specification suite are:

OCCI Core describes the formal definition of the the OCCI Core Model [1].

OCCI HTTP Rendering defines how to interact with the OCCI Core Model using the RESTful OCCI API [2]. The document defines how the OCCI Core Model can be communicated and thus serialised using the HTTP protocol.

OCCI Infrastructure contains the definition of the OCCI Infrastructure extension for the IaaS domain [3]. The document defines additional resource types, their attributes and the actions that can be taken on each resource type.

2 Notational Conventions

All these parts and the information within are mandatory for implementors (unless otherwise specified). The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [4].

3 OCCI JSON Rendering

The OCCI JSON Rendering specifies a rendering of OCCI instance types in the JSON data interchange format as defined in [5].

The Rendering can be used to render OCCI instances independently of the transport mechanism being used. Thus messages can be delivered by e.g. the HTTP protocol as specified in [2] or by using text files with the .json file extension as defined in [5].

[1] Infrastructure as a Service

4 Namespace

The JSON Rendering provides a rendering (i.e. serialisation) of the OCCI Core model into an URL hierarchy by binding Kind and Mixin instances to unique URL paths. Such a URL path is called the *location* of the Kind or Mixin.

A provider is free to choose the *location* as long as it is unique within the service provider's URL namespace. It is recommended, that the locations are based on the term of the Kind or Mixin. To prevent namespace collisions between Kind and Mixin locations, it is recommended to prefix all Mixin locations with the location `/mixins/`.

A Kind instance whose associated type cannot be instantiated MUST NOT be bound to an URL path. This applies to the Kind instance for OCCI Entity.

5 JSON Format

The OCCI JSON Rendering consists of a JSON object holding information on the OCCI Core instances kind, mixin, action, link and resource. The rendering of each OCCI Core instance will be described in the following sections.

The following media-type MUST be used for the OCCI JSON Rendering:

`application/occi+json`

5.1 Resource Instance Format

The resource instance format consists of a JSON object as shown in the following example. Section 6.1 contains a detailed example. Table 1 defines the object members.

```
{
    "resources": [
        {
            "kind": "...",
            "mixins": [ "...", "..." ],
            "attributes": { },
            "actions": [ { } ],
            "links": [ { }, { } ]
        }
    ]
}
```

Table 1. Resource instances are rendered inside the top-level JSON object with name *resources* as an array of JSON objects with the following entries:

Object member	JSON type	Description	Mutability	Multiplicity
kind	string	Type identifier	immutable	1
mixins	array of strings	List of type identifiers of associated mixins	mutable	0..*
attributes	object	Instance attributes	mutable	0..*
actions	array of objects	Applicable actions as defined in 5	mutable	0..*
links	array of objects	Associated OCCI Links as defined in 2	mutable	0..*

5.2 Link Instance Format

The link instance format consists of a JSON object as shown in the following example. Section 6.2 contains a detailed example. Table 2 defines the object members.

```
{
    "links": [
        {
            "kind": "...",
            "rel": "...",
            "mixins": [ "...", "..." ],
            "attributes": { },
            "actions": [ { } ]
        }
    ]
}
```

Table 2. Link instances are rendered inside the top-level JSON object with name *links* as an array of JSON objects with the following entries:

Object member	JSON type	Description	Mutability	Multiplicity
kind	string	Type identifier	immutable	1
rel	string	Type identifier of the target resource	immutable	1
mixins	array of strings	List of type identifiers of associated mixins	mutable	0..*
attributes	object	Instance attributes	mutable	0..*
actions	array of objects	Applicable actions as defined in 5	mutable	0..*

5.3 Kind Format

An OCCI kind is used to describe a OCCI entity and cannot itself be instantiated. OCCI kinds provide a complete description of a specific OCCI entity sub-type.

The kind format consists of a JSON object as shown in the following example. Section 6.3 contains a detailed example. Table 3 defines the top-level object members.

Table 3. Kind instances are rendered inside the top-level JSON object with name *kinds* as an array of JSON objects with the following entries:

Object member	JSON type	Description	Mutability	Multiplicity
term	string	Unique identifier within the categorisation scheme	immutable	1
scheme	string	Categorisation scheme	immutable	1
title	string	Title of the kind	immutable	0..1
attributes	object	Attribute description, see 6	immutable	0..*
related	array of strings	List of type identifiers containing only the related “parent” Kind instance	immutable	0..1
actions	array of strings	List of action type identifiers	immutable	0..*
location	string	URI bound to the Kind instance. MUST be supplied for the kinds of all entities except the entity kind itself	immutable	0..1

```
{
    "kinds": [
        {
            "term": "...",
            "scheme": "...",
            "title": "...",
            "attributes": { },
            "actions": [ "...", "..." ],
            "related": [ "...", "..." ],
            "location": "..."
        }
    ]
}
```

5.4 Mixin Format

An OCCI mixin can be used to extend OCCI entities and cannot itself be instantiated. OCCI mixins provide a description of attributes and actions extending a specific OCCI entity sub-type.

The mixin format consists of a JSON object as shown in the following example. Section 6.4 contains a detailed example. Table 4 defines the top-level object members.

Table 4. Mixin instances are rendered inside the top-level JSON object with name *mixins* as an array of JSON objects with the following entries:

Object member	JSON type	Description	Mutability	Multiplicity
term	string	Unique identifier within the categorisation scheme	immutable	1
scheme	string	Categorisation scheme	immutable	1
title	string	Title of the mixin	immutable	0..1
attributes	object	Attribute description, see 6	immutable	0..*
related	array of strings	List of type identifiers of the related "parent" Mixin instances	immutable	0..*
actions	array of strings	List of action type identifiers	immutable	0..*
location	string	URI bound to the Mixin instance	immutable	1

```
{
    "mixins": [
        {
            "term": "...",
            "scheme": "...",
            "title": "...",
            "attributes": { },
            "actions": [ "...", "..." ],
            "related": [ "...", "..." ],
            "location": "..."
        }
    ]
}
```

5.5 Action Format

An OCCI action can be used to trigger specific actions on an OCCI entity and cannot itself be instantiated. Applicable actions SHOULD be linked to an OCCI resource.

The action format consists of a JSON object as shown in the following example. Table 5 defines the top-level object members.

Table 5. Action instances are rendered inside the top-level JSON object with name *actions* as an array of JSON objects with the following entries:

Object member	JSON type	Description	Mutability	Multiplicity
term	string	Unique type identifier within the categorisation scheme	immutable	1
scheme	string	Categorisation scheme	immutable	1
title	string	Title of the action	immutable	0..1
attributes	object	Attribute description, see 6	immutable	0..*
location	string	URI bound to the Action instance	immutable	

```
{
    "actions": [
        {
            "term": "...",
            "scheme": "...",
            "title": "...",
            "attributes": { },
            "location": "..."
        }
]
}
```

5.6 Attribute Description Format

Attribute descriptions of OCCI Categories are rendered as JSON objects. The dots of the attribute names define a hierarchy. This hierarchy is reflected by JSON objects within the higher layer JSON object or within the top level JSON object with name *attributes*. The last part of the attribute name hierarchy includes the properties-object pairs of the attribute as defined in table 6

Table 6. The attribute-properties object has the members defined in this table. All attribute properties are optional and the table shows which property default value an OCCI client MUST assume if a particular property is unspecified.

Object member	JSON type	Description	Default
mutable	boolean	Defines if the attribute is mutable by the client	false
required	boolean	defines if the attribute MUST be specified at resource instantiation	false
type	string	Enum {string, number, boolean}	string
pattern	string	Posix Extended Regular Expression as defined in [6]. For interoperability reasons, POSIX character classes (e.g. [:alpha:]) MUST NOT be used.	.*
default	string, number or boolean	Attribute default when not specified by client.	
description	string	Description of the attribute	

```
{
    "attributes": {
        "...": {
            "mutable": true,
            "required": false,
            "type": "string",
            "pattern": ".*",
            "default": null,
            "description": "..."
        }
    }
}
```

6 Detailed Examples

6.1 Resource Instance Format Example

```
{
    "resources": [
        {
            "kind": "http: //schemas.ogf.org/occi/infrastructure#compute",
            "mixins": [
                "http: //schemas.opennebula.org/occi/infrastructure#my_mixin",
                "http: //example.com/occi#my_mixin"
            ],
            "attributes": {
                "occi": {
                    "compute": {
                        "speed": 2,
                        "memory": 4,
                        "cores": 2
                    }
                },
                "com": {
                    "example": {
                        "occi": {
                            "my_mixin": {
                                "my_attribute": "my_value"
                            }
                        }
                    }
                }
            },
            "actions": [
                {
                    "title": "Start My Server",
                    "location":
"/compute/996ad860-2a9a-504f-8861-aeafd0b2ae29?action=start",
                    "category":
"http://schemas.ogf.org/occi/infrastructure/compute/action#start"
                }
            ],
             "id": "996ad860-2a9a-504f-8861-aeafd0b2ae29",
    "title": "Compute resource",
    "summary": "This is a compute resource",
    "links": [
        {
            "target":
"http://myservice.tld/storage/59e06cf8-f390-5093-af2e-3685be593a25",
            "kind":
"http://schemas.ogf.org/occi/infrastructure#storagelink",
            "attributes": {
                "occi": {
                    "storagelink": {
                        "deviceid": "ide:0:1"
                    }
                }
            },
            "id": "391ada15-580c-5baa-b16f-eeb35d9b1122",
            "title": "My disk"
        }
    ]
        }
    ]
}
```

6.2 Link Instance Format Example

```
{
    "links": [
        {
            "kind":
"http://schemas.ogf.org/occi/infrastructure#networkinterface",
            "mixins": [
                "http://schemas.ogf.org/occi/infrastructure/networkinterface#
ipnetworkinterface"
            ],
            "attributes": {
                "occi": {
                    "infrastructure": {
                        "networkinterface": {
                            "interface": "eth0",
                            "mac": "00:80:41:ae:fd:7e",
                            "address": "192.168.0.100",
                            "gateway": "192.168.0.1",
                            "allocation": "dynamic"
                        }
                    }
                }
            },
            "actions": [
                {
                    "title": "Disable networkinterface",
                    "href":
"/networkinterface/22fe83ae-a20f-54fc-b436-cec85c94c5e8?action=up",
                    "category": "http:
//schemas.ogf.org/occi/infrastructure/networkinterface/action#up"
                }
            ],
            "id": "22fe83ae-a20f-54fc-b436-cec85c94c5e8",
    "title": "My network interface",
            "target":
"http://myservice.tld/network/b7d55bf4-7057-5113-85c8-141871bf7635",
            "source":
"http://myservice.tld/compute/996ad860-2a9a-504f-8861-aeafd0b2ae29"
        }
    ]
}
```

6.3 Kind Format Example

```
{
    "kinds": [
        {
            "term": "compute",
            "scheme": "http://schemas.ogf.org/occi/infrastructure#",
            "title": "Compute Resource",
            "related": [
                "http://schemas.ogf.org/occi/core#resource"
            ],
            "attributes": {
                "occi": {
                    "compute": {
                        "hostname": {
                            "mutable": true,
                            "required": false,
                            "type": "string",
                            "pattern":
"(([a-zA-Z0-9]|[a-zA-Z0-9][a-zA-Z0-9\\-]*[a-zA-Z0-9])\\.)*",
                            "minimum": "1",
                            "maximum": "255"
                        },
                        "state": {
                            "mutable": false,
                            "required": false,
                            "type": "string",
                            "pattern": "inactive|active|suspended|failed",
                            "default": "inactive"
                        }
                    }
                }
            },
            "actions": [
                "http://schemas.ogf.org/occi/infrastructure/compute/action#start
",
                "http://schemas.ogf.org/occi/infrastructure/compute/action#stop"
,
                "http://schemas.ogf.org/occi/infrastructure/compute/action#
restart",
                "http://schemas.ogf.org/occi/infrastructure/compute/action#
suspend"
            ],
            "location": "/compute/"
        }
    ]
}
```

6.4 Mixin Format Example

```
{
    "mixins": [
        {
            "term": "medium",
            "scheme": "http://example.com/template/resource#",
            "title": "Medium VM",
            "related": [
                "http://schemas.ogf.org/occi/infrastructure#resource_tpl"
            ],
            "attributes": {
                "occi": {
                    "compute": {
                        "speed": {
                            "type": "number",
                            "default": 2.8
                        }
                    }
                }
            },
            "location": "/template/resource/medium/"
        }
    ]
}
```

6.5 Action Format Example

```
{
    "actions": [
        {
            "term": "stop",
            "scheme":
"http://schemas.ogf.org/occi/infrastructure/compute/action#",
            "title": "Stop Compute instance",
            "attributes": {
                "method": {
                    "mutable": true,
                    "required": false,
                    "type": "string",
                    "pattern": "graceful|acpioff|poweroff",
                    "default": "poweroff"
                }
            }
        }
    ]
}
```

7 Glossary

Term	Description
Action	An OCCI base type. Represent an invocable operation on a Entity sub-type instance or collection thereof.
Category	A type in the OCCI model. The parent type of Kind.
Client	An OCCI client.
Collection	A set of Entity sub-type instances all associated to a particular Kind or Mixin instance.
Entity	An OCCI base type. The parent type of Resource and Link.
Kind	A type in the OCCI model. A core component of the OCCI classification system.
Link	An OCCI base type. A Link instance associate one Resource instance with another.
mixin	An instance of the Mixin type associated with a **resource instance**. The “mixin” concept as used by OCCI *only* applies to instances, never to Entity types.
Mixin	A type in the OCCI model. A core component of the OCCI classification system.
OCCI	Open Cloud Computing Interface.
OCCI base type	One of Entity, Resource, Link or Action.
OGF	Open Grid Forum.
Resource	An OCCI base type. The parent type for all domain-specific resource types.
resource instance	An instance of a sub-type of Entity. The OCCI model defines two sub-types of Entity, the Resource type and the Link type. However, the term *resource instance* is defined to include any instance of a *sub-type* of Resource or Link as well.
Tag	A Mixin instance with no attributes or actions defined.
Template	A Mixin instance which if associated at resource instantiation time pre-populate certain attributes.
type	One of the types defined by the OCCI model. The OCCI model types are Category, Kind, Mixin, Action, Entity, Resource and Link.
concrete type/sub-type	A concrete type/sub-type is a type that can be instantiated.
URI	Uniform Resource Identifier.
URL	Uniform Resource Locator.
URN	Uniform Resource Name.

8 Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

9 Disclaimer

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

10 Full Copyright Notice

References

[1] R. Nyrén, A. Edmonds, A. Papaspyrou, and T. Metsch, “Open Cloud Computing Interface – Core,” GFD-P-R.183, April 2011. [Online]. Available: http://ogf.org/documents/GFD.183.pdf

[2] T. Metsch and A. Edmonds, “Open Cloud Computing Interface – HTTP Rendering,” GFD-P-R.185, April 2011. [Online]. Available: http://ogf.org/documents/GFD.185.pdf

[3] ——, “Open Cloud Computing Interface – Infrastructure,” GFD-P-R.184, April 2011. [Online]. Available: http://ogf.org/documents/GFD.184.pdf

[4] S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels,” RFC 2119 (Best Current Practice), Internet Engineering Task Force, Mar. 1997. [Online]. Available: http://www.ietf.org/rfc/rfc2119.txt

[5] D. Crockford, “The application/json Media Type for JavaScript Object Notation (JSON),” RFC 4627 (Informational), Internet Engineering Task Force, Jul. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4627.txt

[6] “Information technology – portable operating system interface (posix) base specifications, issue 7,” 2009.