XXXXX
OCCI-WG

Augusto Ciuffoletti, Università di Pisa
Thijs Metsch, Platform Computing
Andy Edmonds, Intel
September 22, 2014
Updated: September 24, 2014

Open Cloud Computing Interface - Notification Extension

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 through *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.

TODO: replace with 1.2, note backwards compatibility. define new set of docs for 1.2 below...

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 [1].

3 Motivations

It is often the case that an entity changes during its lifetime: for instance a Compute resource experiences a transient at the beginning of its lifetime during bootup [2].

We want to give the provider the tools to allow the user to define entities such that their changes are observable. The specifications for such tools are the premises for an interoperable notification framework.

We introduce an OCCI Extension that allows the user to differentiate an OCCI resource that produces notifications, and how such notifications are visible to other OCCI Resources.

4 OCCI notification

The way to define a property of an OCCI entity is to associate a mixin to it. So the straightforward way to assert that an entity is one whose changes are observable is by associating a mixin that is related with this property. We call this mixin **Notify**.

To define where a notification is directed the user defines an OCCI Link of a definite subkind, that is called **Notification**.

[1] Infrastructure as a Service

4.1 The Notify mixin

Table 1. The immutable model attributes of the **Notify** mixin. The base URL **http://schemas.ogf.org/occi** has been replaced with <**schema**> in this table for a better reading experience.

Term	Scheme	Title	Attributes	Actions	Depends	Applies
Notify	<schema>/notification#	**Notify** Mixin	{}	{}	{}	<schema>/core#Resource

The provider that supports the OCCI Notification extension MUST implement the **Notify**mixin for each provided entity kind.

There is no capabillity associated with the **Notify**mixin: it is a *tag*.

4.2 The Notification link

Table 2. The immutable model attributes of the **Notification**category. The base URL **http://schemas.ogf.org/occi** has been replaced with <**schema**> in this table for a better readability experience.

Term	Scheme	Title	Attributes	Actions	Parent
Notification	<schema>/notification#	**Notification** Link	{}	{}	<schema>/core#Link

The target of a **Notification** is a generic Resource, while the source MUST be associated with the **Notify** mixin.

The instantiation of an **Notification** whose source is not associated with a **Notify** mixin fails and an error is raised.

If the **Notification** mixin associated with a Resource is removed, all outgoing **Notification** links are silently removed.

There is no capability associated with the **Notification**.

According with the core model [3], the resource that is the source of **Notification**s is able to discover all such links through the *links* property: this is not true for the target of the link.

According with the core model [3], the removal of the source of an **Notification**link determines the removal of the link itself: the same is not true for the target.

5 Application notes and an example

From the user perspective, the application of the **Notify** corresponds to enabling the access to the dynamic content of a resource. This is significant for a Resource that is changing in time.

The **Notify** alone does ot specify which changing aspect is in fact notified, and how. This specification may be made explicit with a further mixin, or may be left implicit as in the following example.

From the provider perspective the association of an **Notify** is reflected in the implementation of the functionalities needed to observe and render the change. This operation is guided by the Resources that are targets of the **Notification**.

The way in which notifications are used falls ouside the scope of this document: as a general rule, they are used for monitoring and management. Such aspects can be defined by the user with mixins associated with the **Notification**, or in the Resource targeted by the **Notification**.

The following example illustrates a use case where the explicit description of the capabilities of the involved entities is not required.

A provider offers a Resource of type *3-out-of-k*, that keeps in the *active* state 3 of the Compute resources from which it receives notifications.

The user that wants to take advantage of this service instantes a *3-out-of-k* Resource $3ook$, and associated an **Notify** to each of the k Compute resources $C_{1..k}$. For each of them a **Notification** link N_i is instantiated that originates from C_i and targets $3ook$.

The schema is portable across any platform that offers the OCCI-infrastructure and OCCI-notification, and that provides a *3-out-of-k* Resource type.

Distinct providers may interoperate, for instance if one provides the *3-out-of-k* Resource and another the Compute resources, provided that an agreement exists between the two that allows cross provider information transfer.

6 Security issues

The OCCI Notification specification is an extension to the OCCI Core and Model specification [3]; thus the same security considerations as for the OCCI Core and Model specification apply here.

7 Glossary

Term	Description
Action	An OCCI base type. Represents an invocable operation on a Entity sub-type instance or collection thereof.
Attribute	A type in the OCCI Core Model. Describes the name and properties of attributes found in Entity types.
Category	A type in the OCCI Core Model and the basis of the OCCI type identification mechanism. The parent type of Kind.
capabilities	In the context of Entity sub-types **capabilities** refer to the OCCI Attributes and OCCI Actions exposed by an **entity instance**.
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.
entity instance	An instance of a sub-type of Entity but not an instance of the Entity type itself. The OCCI model defines two sub-types of Entity, the Resource type and the Link type. However, the term *entity instance* is defined to include any instance of a sub-type of Resource or Link as well.
Kind	A type in the OCCI Core Model. A core component of the OCCI classification system.
Link	An OCCI base type. A Link instance associates one Resource instance with another.
Mixin	A type in the OCCI Core Model. A core component of the OCCI classification system.
mix-in	An instance of the Mixin type associated with an *entity instance*. The "mix-in" concept as used by OCCI *only* applies to instances, never to Entity types.
model attribute	An internal attribute of a the Core Model which is *not* client discoverable.
OCCI	Open Cloud Computing Interface.
OCCI base type	One of Entity, Resource, Link or Action.
OCCI Action	see Action.
OCCI Attribute	A client discoverable attribute identified by an instance of the Attribute type. Examples are occi.core.title and occi.core.summary.
OCCI Category	see Category.
OCCI Entity	see Entity.
OCCI Kind	see Kind.
OCCI Link	see Link.
OCCI Mixin	see Mixin.
OGF	Open Grid Forum.
Resource	An OCCI base type. The parent type for all domain-specific Resource sub-types.
resource instance	See *entity instance*. This term is considered obsolete.
tag	A Mixin instance with no attributes or actions defined.
template	A Mixin instance which if associated at instance creation-time pre-populate certain attributes.
type	One of the types defined by the OCCI Core Model. The Core Model types are Category, Attribute, 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] 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

[2] T. Metsch and A. Edmonds, "Open Cloud Computing Interface – Infrastructure," GFD-P-R.184, April 2011. [Online]. Available: http://ogf.org/documents/GFD.184.pdf

[3] 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