

1 Draft  
2 OCCI-WG  
3  
4

Thijs Metsch, Intel  
Andy Edmonds, ICCLab, ZHAW  
Boris Parák, CESNET  
Updated: February 23, 2016

## 5 **Open Cloud Computing Interface – Infrastructure**

### 6 Status of this Document

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

### 9 Copyright Notice

10 Copyright © Open Grid Forum (2009-2016). All Rights Reserved.

### 11 Trademarks

12 OCCI is a trademark of the Open Grid Forum.

### 13 Abstract

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

**17 Contents**

18	<b>1 Introduction</b>	<b>3</b>
19	<b>2 Notational Conventions</b>	<b>3</b>
20	<b>3 Infrastructure</b>	<b>4</b>
21	3.1 Compute . . . . .	5
22	3.2 Network . . . . .	6
23	3.2.1 IPNetwork Mixin . . . . .	7
24	3.3 Storage . . . . .	8
25	3.4 Linking Infrastructure Resources . . . . .	8
26	3.4.1 Linking to Network . . . . .	9
27	3.4.2 Linking to Storage . . . . .	10
28	3.4.3 Linking to CDMI Managed Storage . . . . .	11
29	3.5 Infrastructure Templates . . . . .	11
30	3.5.1 OS Template . . . . .	11
31	3.5.2 Resource Template . . . . .	12
32	<b>4 Security Considerations</b>	<b>14</b>
33	<b>5 Glossary</b>	<b>14</b>
34	<b>6 Contributors</b>	<b>14</b>
35	<b>7 Intellectual Property Statement</b>	<b>16</b>
36	<b>8 Disclaimer</b>	<b>16</b>
37	<b>9 Full Copyright Notice</b>	<b>16</b>
38	<b>A Change Log</b>	<b>17</b>

## 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<sup>1</sup> 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 complementary documents, which together form the complete specification. The documents are divided into four categories consisting of the OCCI Core, the OCCI Protocols, the OCCI Renderings and the OCCI Extensions.

- The OCCI Core specification consists of a single document defining the OCCI Core Model. OCCI interaction occurs through *renderings* (including associated behaviors) and is expandable through *extensions*.
- The OCCI Protocol specifications consist of multiple documents, each describing how the model can be interacted with over a particular protocol (e.g. HTTP, AMQP, etc.). Multiple protocols can interact with the same instance of the OCCI Core Model.
- 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.

The current specification consists of seven documents. This specification describes version 1.2 of OCCI and is backward compatible with 1.1. Future releases of OCCI may include additional protocol, rendering and extension specifications. The specifications to be implemented (MUST, SHOULD, MAY) are detailed in the table below.

**Table 1.** What OCCI specifications must be implemented for the specific version.

Document	OCCI 1.1	OCCI 1.2
Core Model	MUST	MUST
Infrastructure Model	SHOULD	SHOULD
Platform Model	MAY	MAY
SLA Model	MAY	MAY
HTTP Protocol	MUST	MUST
Text Rendering	MUST	MUST
JSON Rendering	MAY	MUST

OCCI makes an ideal inter-operable boundary interface between the web and the internal resource management system of infrastructure providers.

## 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].

<sup>1</sup>Infrastructure as a Service

### 3 Infrastructure

The OCCI Infrastructure document details how an OCCI implementation can model and implement an Infrastructure as a Service API offering by utilizing the OCCI Core Model. This API allows for the creation and management of typical resources associated with an IaaS service, for example, creating a **Compute** instance and **Storage** instance and then linking them with **StorageLink**. The main infrastructure types defined within OCCI Infrastructure are:

**Compute** Information processing resources.

**Network** Interconnection resource that represents an L2 networking resource. This is complemented by the **IPNetwork Mixin**.

**Storage** Information recording resources.

Supporting these **Resource** types are the following **Link** sub-types:

**NetworkInterface** connects a **Compute** instance to a **Network** instance. This is complemented by an **IPNetworkInterface Mixin**.

**StorageLink** connects a **Compute** instance to a **Storage** instance.

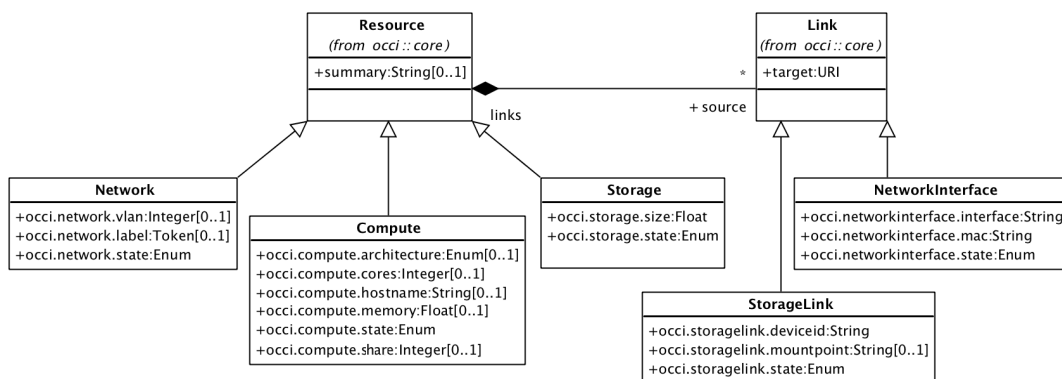


Figure 1. Overview Diagram of OCCI Infrastructure Types.

These infrastructure types inherit the OCCI Core Model **Resource** base type and all its attributes. The HTTP Protocol [2] and Text Rendering [?] documents define how to serialize and interact with these types using RESTful communication. Implementers are free to choose what **Resource** and **Link** sub-types to implement. Those that are supported by an implementation will be discoverable through the OCCI Query Interface.

As REQUIRED by the OCCI Core Model specification, every type instantiated that is a sub-type of **Resource** or **Link** MUST be assigned a **Kind** that identifies the instantiated type. Each such **Kind** instance MUST be related to the **Resource** or **Link** base type's **Kind** by setting the *parent* attribute. That assigned **Kind** instance MUST always remain immutable to any client.

Table 2. The **Kind** instances defined for the infrastructure sub-types of **Resource**, **Link** and related **Mixins**. The base URL <http://schemas.ogf.org/occi> has been replaced with <schema> in this table for a better readability experience.

Term	Scheme	Title	Parent Kind
compute	<schema>/infrastructure#	Compute <b>Resource</b>	<schema>/core#resource
storage	<schema>/infrastructure#	Storage <b>Resource</b>	<schema>/core#resource
storagelink	<schema>/infrastructure#	StorageLink <b>Link</b>	<schema>/core#link
network	<schema>/infrastructure#	Network <b>Resource</b>	<schema>/core#resource
networkinterface	<schema>/infrastructure#	NetworkInterface <b>Link</b>	<schema>/core#link

95 Table 2 describes the **Kind** instances defined for each of the infrastructure **Resource** or **Link** sub-types. For  
 96 information on extending these types, please refer to the OCCI Core Model document [3].

97 The following sections on **Compute**, **Storage** and **Network** types detail the **Attributes**, **Actions** and states  
 98 defined for each of them, including type-specific mixins where appropriate. Following those, the definition of  
 99 infrastructure-related **Link** sub-types are given and finally OS and Resource Templates are defined. Figure 1  
 100 gives an overview of the key types involved in this infrastructure specification.

### 101 3.1 Compute

102 The **Compute** type represents a generic information processing resource, e.g., a virtual machine or container.  
 103 **Compute** inherits the **Resource** base type defined in OCCI Core Model [3]. **Compute** is assigned the **Kind**  
 104 instance <http://schemas.ogf.org/occi/infrastructure#compute>. A **Compute** instance MUST use and expose  
 105 this **Kind**.

**Table 3.** **Attributes** defined for the **Compute** type.

Attribute	Type	Multiplicity	Mutability	Description
occi.compute.architecture	Enum {x86, x64}	0..1	Mutable	CPU Architecture of the instance.
occi.compute.cores	Integer	0..1	Mutable	Number of virtual CPU cores assigned to the instance.
occi.compute.hostname	String	0..1	Mutable	Fully Qualified DNS hostname for the instance.
occi.compute.share	Integer	0..1	Mutable	Relative number of CPU shares for the instance.
occi.compute.memory	Float, 10 <sup>9</sup> (GiB)	0..1	Mutable	Maximum RAM in gigabytes allocated to the instance.
occi.compute.state	Enum {active, inactive, suspended, error}	1	Immutable	Current state of the instance.
occi.compute.state.message	String	0..1	Immutable	Human-readable explanation of the current instance state.

106 Table 3 describes the OCCI **Attributes**<sup>2</sup> defined by **Compute** through its **Kind** instance. These attributes MAY  
 107 or MUST be exposed by an instance of the **Compute** type depending on the “Multiplicity” column in the  
 108 aforementioned table.

**Table 4.** **Actions** applicable to instances of the **Compute** type. The **Actions** are defined by the **Kind**  
 instance <http://schemas.ogf.org/occi/infrastructure#compute>. Every **Action** instance in the table uses the  
<http://schemas.ogf.org/occi/infrastructure/compute/action#> categorization scheme. “Action Term” below refers to  
**Action.term**.

Action Term	Target state	Attributes
start	active	–
stop	inactive	method={graceful, acpioff, poweroff}
restart	active (via stop and start chain)	method={graceful, warm, cold}
suspend	suspended	method={hibernate, suspend}
save	active (via stop and start chain)	method={hot, deferred}, name= <i>String</i>

109 Table 4 describes the **Actions** defined for **Compute** by its **Kind** instance. These **Actions** MUST be exposed  
 110 by an instance of the **Compute** type of an OCCI implementation. Figure 2 illustrates the state diagram for a  
 111 **Compute** instance.

112 Action “save” is expected to create an OS Template (see Section 3.5.1) referencing an independent copy  
 113 of the current state of the **Compute** instance. The provider MAY choose to respect the “name” given by  
 114 the client or override it according to its internal policies. A successful execution of this action MUST lead to  
 115 a response containing the rendering of the newly created OS Template as defined by the chosen rendering

<sup>2</sup>See the “attributes” attribute defined by the **Category** type and inherited by **Kind** [3].

116 and transport protocol. The provider MAY choose to include a reference to the original **Compute** instance in  
 117 **Mixin.Attributes** of the newly created OS Template.

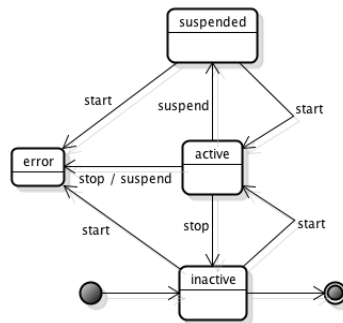


Figure 2. State Diagram for a **Compute** instance.

### 118 3.2 Network

119 The **Network** type represents an L2 networking entity (e.g., a virtual switch). It can be extended using the  
 120 mixin mechanism (or sub-typed) to support L3/L4 capabilities such as TCP/IP etc. For the purposes of this  
 121 specification we define an OCCI mixin so that IP networking can be supported where required. **Network** inherits  
 122 the **Resource** base type defined in OCCI Core Model [3].

123 The **Network** type is assigned the `http://schemas.ogf.org/occi/infrastructure#network` **Kind**. A **Network**  
 124 instance MUST use and expose this **Kind**.

Table 5. Attributes defined for the **Network** type.

Attribute	Type	Multiplicity	Mutability	Description
occi.network.vlan	Integer: 0-4095	0..1	Mutable	802.1q VLAN Identifier (e.g., 343).
occi.network.label	Token	0..1	Mutable	Tag based VLANs (e.g., external-dmz).
occi.network.state	Enum {active, inactive, error}	1	Immutable	Current state of the instance.
occi.network.state.message	String	0..1	Immutable	Human-readable explanation of the current instance state.

125 Table 5 describes the OCCI **Attributes**<sup>3</sup> defined by **Network** through its **Kind** instance. These attributes MAY  
 126 or MUST be exposed by an instance of the **Network** type depending on the “Multiplicity” column in the  
 127 aforementioned table.

Table 6. Actions applicable to instances of the **Network** type. The **Actions** are defined by the **Kind** instance `http://schemas.ogf.org/occi/infrastructure#network`. Every **Action** instance in the table uses the `http://schemas.ogf.org/occi/infrastructure/network/action#` categorisation scheme. “Action Term” below refers to **Action.term**.

Action Term	Target State	Attributes
up	active	–
down	inactive	–

128 Table 6 describes the **Actions** defined for **Network** by its **Kind** instance. These **Actions** MUST be exposed  
 129 by an instance of the **Network** type of an OCCI implementation. Figure 3 illustrates the state diagram for a  
 130 **Network** instance.

<sup>3</sup>See the “attributes” attribute defined by the **Category** type and inherited by **Kind** [3].

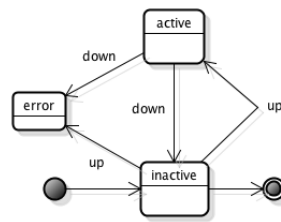


Figure 3. State Diagram for a **Network** instance.

131 **3.2.1 IPNetwork Mixin**

132 In order to support L3/L4 capabilities (e.g., IP, TCP, etc.) an OCCI mixin is herewith defined.

133 The **IPNetwork** mixin is assigned<sup>4</sup> the “scheme” of <http://schemas.ogf.org/occi/infrastructure/network#> and  
 134 the “term” value *ipnetwork*. An **IPNetwork** mixin MUST support these values.

135 Table 7 defines the attributes introduced by the **IPNetwork** mixin.

136 The **IPNetwork** mixin MUST be related to the **Network** kind by setting the *applies* attribute to:

137 <http://schemas.ogf.org/occi/infrastructure#network>.

138 A **Network** instance associated with the **IPNetwork** mixin’s **Mixin** instance MUST implement these attributes.

**Table 7.** Attributes defined by the **IPNetwork** mixin. A **Network** instance associated with this **Mixin** instance MUST expose these attributes.

Attribute	Type	Multi- plicity	Mutability	Description
occi.network.address	IPv4 or IPv6 Address range, CIDR notation	0..1	Mutable	Internet Protocol (IP) network address (e.g., 192.168.0.1/24, fc00::/7)
occi.network.gateway	IPv4 or IPv6 Address	0..1	Mutable	Internet Protocol (IP) network address (e.g., 192.168.0.1, fc00::)
occi.network.allocation	Enum {dynamic, static}	0..1	Mutable	Address allocation mechanism: <i>dynamic</i> e.g., uses the dynamic host configuration protocol, <i>static</i> e.g., uses user supplied static network configurations.

139 In Figure 4 a UML object diagram depicts how **Network** would be associated with an **IPNetwork Mixin** when  
 140 both are instantiated.

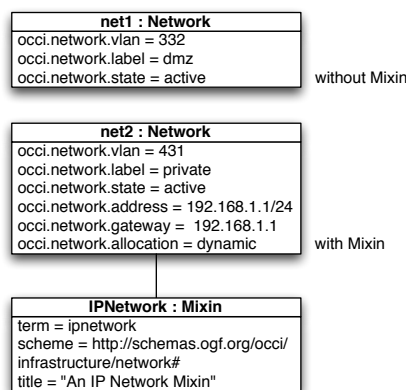


Figure 4. Object Diagram of a **Network** Instance and its associated **IPNetwork Mixin**.

<sup>4</sup>Both assignments use data members from the inherited **Category** type [3].

### 141 3.3 Storage

142 The **Storage** type represents resources that record information to a data storage device. **Storage** inherits the  
 143 **Resource** base type defined in the OCCI Core Model [3]. The **Storage** type is assigned the **Kind** instance  
 144 <http://schemas.ogf.org/occi/infrastructure#storage>. A **Storage** instance MUST use and expose this **Kind**.

**Table 8.** Attributes defined for the **Storage** type.

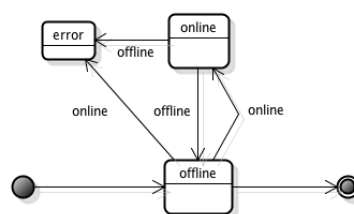
Attribute	Type	Multi- plicity	Mutability	Description
occi.storage.size	Float, 10 <sup>9</sup> (GiB)	1	Mutable	Storage size of the instance in gigabytes.
occi.storage.state	Enum {online, off- line, error}	1	Immutable	Current status of the instance.
occi.storage.state.message	String	0..1	Immutable	Human-readable explanation of the cur- rent instance state.

145 Table 8 describes the OCCI **Attributes**<sup>5</sup> defined by **Storage** through its **Kind** instance. These attributes MAY  
 146 or MUST be exposed by an instance of the **Storage** type depending on the “Multiplicity” column in the  
 147 aforementioned table.

**Table 9.** **Actions** applicable to instances of the **Storage** type. The **Actions** are defined by the **Kind**  
 instance <http://schemas.ogf.org/occi/infrastructure#storage>. Every **Action** instance in the table uses the  
<http://schemas.ogf.org/occi/infrastructure/storage/action#> categorization scheme. “Action Term” below refers to **Ac-**  
**tion.term**.

Action Term	Target State	Attributes
online	online	–
offline	offline	–

148 Table 9 describes the **Actions** defined for **Storage** by its **Kind** instance. These **Actions** MUST be exposed by an  
 149 instance of the **Storage** type of an OCCI implementation. Figure 5 illustrates the state diagram for a **Storage**  
 150 instance.



**Figure 5.** State Diagram for a **Storage** instance.

151 OCCI can be used in conjunction with the SNIA cloud storage standard, Cloud Data Management Interface  
 152 (CDMI) [4], to provide enhanced management of the cloud computing storage and data. For storage managed  
 153 through CDMI, see Section 3.4.3.

### 154 3.4 Linking Infrastructure Resources

155 In order to create entities like virtual data centers or virtual clusters, it is necessary to allow the linkage of  
 156 the previously defined infrastructure **Resource** sub-types. This is accomplished by extending (sub-typing) the  
 157 OCCI Core Model **Link** base type. This is done as the **Link** base type cannot fully represent specific types of  
 158 infrastructure links (e.g., links to storage or networks). These infrastructure links require additional attributes  
 159 (e.g., network interface name), which can only be supported by sub-typing the **Link** base type.

<sup>5</sup>See the “attributes” attribute defined by the **Category** type and inherited by **Kind** [3].



### 160 3.4.1 Linking to Network

161 The `NetworkInterface` type represents an L2 client device (e.g., network adapter). It can be extended using the  
 162 mix-in mechanism or sub-typed to support L3/L4 capabilities such as TCP/IP, etc. `NetworkInterface` inherits  
 163 the `Link` base type defined in the OCCI Core Model [3].

164 The `NetworkInterface` type is assigned the `Kind` instance `http://schemas.ogf.org/occi/infrastructure#networkinterface`.  
 165 A `NetworkInterface` instance MUST use and expose this `Kind`. The `Kind` instance assigned to the `Network-`  
 166 `Interface` type MUST be related to the `http://schemas.ogf.org/occi/core#link` `Kind` by setting the parent  
 167 attribute.

Table 10. Attributes defined for the `NetworkInterface` type.

Attribute	Type	Multiplicity	Mutability	Description
<code>occi.networkinterface.interface</code>	String	1	Immutable	Identifier that relates the link to the link's device interface.
<code>occi.networkinterface.mac</code>	String	1	Mutable	MAC address associated with the link's device interface.
<code>occi.networkinterface.state</code>	Enum {active, inactive, error}	1	Immutable	Current status of the instance.
<code>occi.networkinterface.state.message</code>	String	0..1	Immutable	Human-readable explanation of the current instance state.

168 Table 10 describes the OCCI `Attributes`<sup>6</sup> defined by `NetworkInterface` through its `Kind` instance. These attributes  
 169 MAY or MUST be exposed by an instance of the `NetworkInterface` type depending on the “Multiplicity” column  
 170 in the aforementioned table. Figure 6 illustrates the state diagram for a `NetworkInterface` instance.

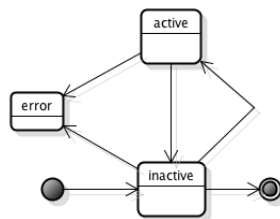


Figure 6. State Diagram for a `NetworkInterface` instance.

171 **3.4.1.1 IPNetworkInterface Mixin** In order to support L3/L4 capabilities (e.g., IP, TCP etc.) with the  
 172 `NetworkInterface` type, an OCCI `Mixin` instance is herewith defined.

173 The `IPNetworkInterface` mixin is assigned<sup>7</sup> the “scheme” of `http://schemas.ogf.org/occi/infrastructure/`  
 174 `networkinterface#` and the “term” value `ipnetworkinterface`. An `IPNetworkInterface` mixin MUST support  
 175 these attributes.

176 The `IPNetworkInterface` mixin MUST be related to the `NetworkInterface` kind by setting the `applies` attribute  
 177 to:

178 `http://schemas.ogf.org/occi/infrastructure#networkinterface`.

179 Table 11 defines the attributes introduced by the `IPNetworkInterface` mixin. A `NetworkInterface` instance  
 180 associated with the `IPNetworkInterface` mixin's `Mixin` instance MUST expose these attributes.

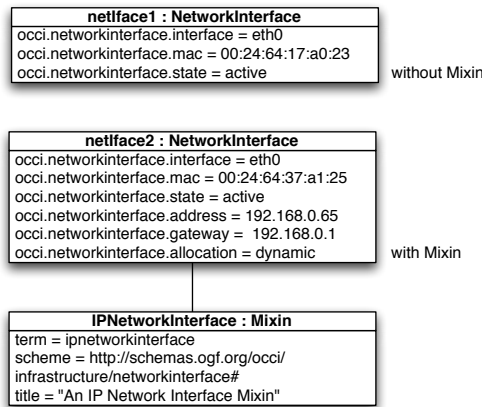
181 In Figure 7 a UML object diagram depicts how `NetworkInterface` would be associated with an `IPNetworkInterface`  
 182 `Mixin` when both are instantiated.

<sup>6</sup>See the “attributes” attribute defined by the `Category` type and inherited by `Kind` [3].

<sup>7</sup>Both assignments use data members from the inherited `Category` type [3].

**Table 11.** Attributes defined by the `IPNetworkInterface` mixin. A `NetworkInterface` instance associated with this `Mixin` instance MUST expose these attributes.

Attribute	Type	Multiplicity	Mutability	Description
<code>occi.networkinterface.address</code>	IPv4 or IPv6 Address	1	Mutable	Internet Protocol(IP) network address (e.g., 192.168.0.1/24, fc00::/7) of the link
<code>occi.networkinterface.gateway</code>	IPv4 or IPv6 Address	0..1	Mutable	Internet Protocol(IP) network address (e.g., 192.168.0.1/24, fc00::/7)
<code>occi.networkinterface.allocation</code>	Enum {dynamic, static}	1	Mutable	Address mechanism: <i>dynamic</i> e.g., uses the dynamic host configuration protocol, <i>static</i> e.g., uses user supplied static network configurations.



**Figure 7.** Object Diagram of a `NetworkInterface` Instance and its Associated `IPNetworkInterface` `Mixin`.

### 3.4.2 Linking to Storage

The `StorageLink` type represents a link from a `Resource` to a target `Storage` instance. This allows a `Storage` instance be attached to a `Compute` instance, with all the prerequisite low-level operations handled by the OCCI implementation. This mechanism SHOULD NOT be used to choose an operating system for the given `Compute` instance, see Section 3.5.1. `StorageLink` inherits the `Link` base type defined in the OCCI Core Model [3].

The `StorageLink` type is assigned the `Kind` instance `http://schemas.ogf.org/occi/infrastructure#storagelink`. A `StorageLink` instance MUST use and expose this `Kind`. The `Kind` instance assigned to the `StorageLink` type MUST be related to the `http://schemas.ogf.org/occi/core#link` `Kind` by setting the parent attribute.

**Table 12.** Attributes defined for the `StorageLink` type.

Attribute	Type	Multiplicity	Mutability	Description
<code>occi.storagelink.deviceid</code>	String	1	Mutable	Device identifier as defined by the OCCI service provider.
<code>occi.storagelink.mountpoint</code>	String	0..1	Mutable	Point to where the storage is mounted in the guest OS.
<code>occi.storagelink.state</code>	Enum {active, inactive, error}	1	Immutable	Current status of the instance.
<code>occi.storagelink.state.message</code>	String	0..1	Immutable	Human-readable explanation of the current instance state.

Table 12 describes the OCCI `Attributes`<sup>8</sup> defined by `StorageLink` through its `Kind` instance. These attributes MAY or MUST be exposed by an instance of the `StorageLink` type depending on the “Multiplicity” column in the aforementioned table. Figure 8 illustrates the state diagram for a `StorageLink` instance.

<sup>8</sup>See the “attributes” attribute defined by the `Category` type and inherited by `Kind` [3].

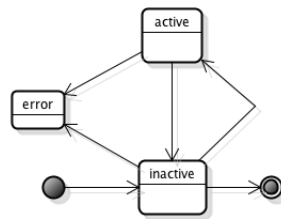


Figure 8. State Diagram for a `StorageLink` instance.

### 3.4.3 Linking to CDMI Managed Storage

As previously stated, OCCI can be used in conjunction with the SNIA cloud storage standard, Cloud Data Management Interface (CDMI) [4], to provide enhanced management of the cloud computing storage and data. In order to integrate the two, the `StorageLink` should be used. This will link OCCI managed Resources to CDMI resources. The “`occi.storagelink.deviceid`” attribute of `StorageLink`, defined above, should be set to the CDMI Object ID of an exported CDMI Container.

## 3.5 Infrastructure Templates

Infrastructure Templates allow clients of an OCCI implementation to quickly and conveniently apply pre-defined configurations to OCCI Infrastructure defined types. They are implemented using `Mixin` instances. There are two supported infrastructure template types in OCCI Infrastructure.

### 3.5.1 OS Template

OS (Operating System) Templates allow clients to specify what operating system must be installed on a requested `Compute` resource. OCCI implementations SHOULD support this, otherwise what they provision will be merely offer `Resources` without any available execution environment (e.g., operating system). They MAY, however, choose to define a default OS Template that will be used if not explicitly specified. Of the two supported template types, OS Template is the most basic and necessary template that a provider SHOULD offer.

Its construction is a `Mixin` instance consisting of a provider specific “scheme” and a descriptive “title” detailing the OS. The “term” value of the template `Mixin` is a provider-specific identifier that corresponds to a particular image configuration. Where an implementation requires additional attributes associated with the OS Template, it can do so using “attributes” value inherited from the `Category` type.

Default values for OCCI `Attributes` defined by the `Kind` or the OS Template `Mixin` MAY be provided using the `Attribute.default` attribute property [3].

An implementation-defined OS Template `Mixin` MUST be related to the OCCI OS Template `Mixin` in order to give absolute type information by setting the `depends` attribute.

The OCCI OS Template is defined by the `http://schemas.ogf.org/occi/infrastructure#os.tpl` `Mixin` and MUST be supported should OS Templates be offered by the OCCI implementation.

Associating a new OS Template with an existing `Resource` instance MAY be supported depending on the limitations of the implementation and MUST result in an immediate removal of the old OS Template and association of the new OS Template. The change MUST affect the execution environment of the given `Resource` instance, in a provider-specific way. If this functionality is not supported, an appropriate error MUST be returned to the client, using mechanisms defined by the chosen rendering and transport protocol.

A typical example of using such a `Mixin` is shown in figure 9 using a UML object diagram. In the example illustrated in figure 9 a provider has defined an OS template which offers the ability to run Ubuntu Linux, version 9.10, upon a client’s provisioned compute resource.

How a provider manages their set of OS templates will be determined by the provider and will be implementation-specific.

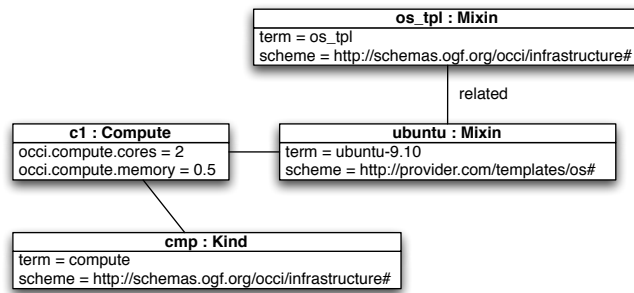


Figure 9. Object Diagram of a **Compute** Instance and its Associated OS Template **Mixin**.

231 **3.5.2 Resource Template**

232 The Resource Template **Mixin** builds upon the concept of OS Templates. A Resource Template is a provider-  
 233 defined **Mixin** instance that refers to a pre-set **Resource** configuration. If a Resource Template **Mixin** is not  
 234 provided, the provider is free to choose a default pre-set **Resource** configuration. If a **Resource** instance carries  
 235 its own size-related attributes, an assigned Resource Template **Mixin** will override them where applicable.

236 The pre-set **Resource** configuration is not fully visible through the OCCI Discovery mechanism, depending on  
 237 the chosen OCCI rendering and necessary provider-specific implementation details. The **Mixin.attributes**  
 238 (inherited from **Category**) for a Resource Template **Mixin** SHOULD contain relevant attributes and default  
 239 attribute values. Provider-specific side-effects are handled by the implementation and MUST NOT be exposed.

240 The OCCI implementation associates a set of Resource attributes (via **Category**'s "attributes") with a particular  
 241 term identifier.

242 An implementation-defined Resource Template **Mixin** MUST be related to the OCCI Resource Template **Mixin**  
 243 in order to give absolute type information. This is done by setting the *depends* attribute. The OCCI Resource  
 244 Template is defined by the **Mixin** instance [http://schemas.ogf.org/occi/infrastructure#resource\\_tpl](http://schemas.ogf.org/occi/infrastructure#resource_tpl) and MUST  
 245 be supported SHOULD Resource Templates be offered by the OCCI implementation.

246 If a Resource Template is already associated with the given **Resource** instance, associating a new Resource  
 247 Template (using mechanisms defined by the chosen rendering and transport protocol) MUST result in an  
 248 immediate removal of the old Resource Template and association of the new Resource Template. The change  
 249 must affect the given **Resource** instance, in a provider-specific way (e.g., resizing the instance).

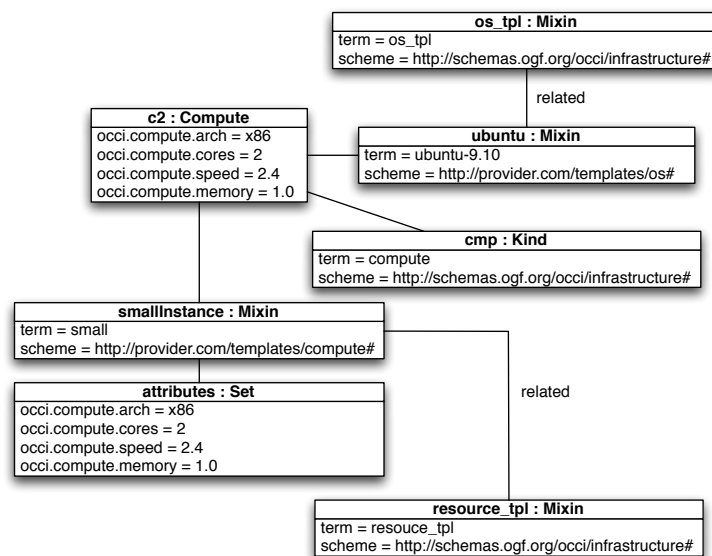


Figure 10. Object Diagram of a **Compute** Instance and its associated OS Template **Mixin** and Resource Template **Mixin**.

250 A typical example of such a **Mixin**'s use is shown in figure 10 using a UML object diagram. In this example,  
 251 the provider offers **Compute Resources** based on different sizes (i.e., small, medium, large). Each "size" of  
 252 **Compute** (i.e., the term) corresponds to a predetermined set of OCCI **Resource**-specific attributes. In the  
 253 example below a "small" **Compute** instance is created. Specifying "small" as the term corresponds to an  
 254 implementation-specific **Compute Resource**-specific attribute set that is shown by the object instance named  
 255 "attributes" in figure 10. When this **Mixin** is associated with a **Compute** instance, the **Compute** instance will  
 256 take on provided attributes and default attribute values.

257 From the administrative point of view, how an OCCI service provider manages their set of Resource Templates  
 258 will be determined by the provider and so is implementation-specific.

259 **3.5.2.1 Credentials Mixin** When creating a **Compute Resource** a client normally supplies security creden-  
 260 tials in the form of a public SSH key. This SSH key is injected into the **Compute Resource** by the provider on  
 261 the client's behalf. This feature is provided by the **Credentials Mixin**.

262 If a provider offers VMs with access secured by SSH then their OCCI implementation SHOULD support this.  
 263 Otherwise no user-supplied public SSH key can be injected into the **Compute Resource**.

264 The OCCI credentials mixin has the term `ssh_key` and the schema `http://schemas.ogf.org/occi/infrastructure/credentials#`.

266 The credentials mixin MUST only apply to the **Compute Kind** and therefore the mixin should have its `applies`  
 267 attribute set to:

268 `http://schemas.ogf.org/occi/infrastructure#compute`.

**Table 13.** Attributes defined by the **Credentials** mixin. A **Compute** instance associated with this **Mixin** instance MUST expose these attributes.

Attribute	Type	Multi- plicity	Mutability	Description
<code>occi.credentials.ssh.publickey</code>	String	1	Mutable	The contents of the public key file to be injected into the <b>Compute Resource</b>

269 **3.5.2.2 Contextualization Mixin** In order to ease automation, OCCI supports the means to execute a  
 270 program once a **Compute Resource** has been instantiated. This feature is provided by the contextualization  
 271 mixin. On receipt of the contextualization data the OCCI implementation MUST distinguish the type of data  
 272 being presented and then supply that content to the **Compute Resource** being instantiated. That content is  
 273 then executed by the **Compute Resource** as the last step in the **Compute**'s boot-order.

274 OCCI implementations SHOULD support this otherwise no contextualization of a resource instance can be done.  
 275 The OCCI contextualization mixin has the term `user_data` and the schema `http://schemas.ogf.org/occi/infrastructure/compute#`.

277 Contextualization mixin MUST only apply to the **Compute Kind** and therefore the mixin should have its  
 278 `applies` attribute set to:

279 `http://schemas.ogf.org/occi/infrastructure#compute`.

**Table 14.** Attributes defined by the **Contextualization** mixin. A **Compute** instance associated with this **Mixin** instance MUST expose these attributes.

Attribute	Type	Multi- plicity	Mutability	Description
<code>occi.compute.userdata</code>	String	1	Mutable	Contextualization data (e.g., script, executable) that the client supplies once and only once. It cannot be updated.

## 280 4 Security Considerations

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

## 283 5 Glossary

Term	Description
Action	An OCCI base type. Represents an invocable operation on an <b>Entity</b> sub-type instance or collection thereof.
Attribute	A type in the OCCI Core Model. Describes the name and properties of attributes found in <b>Entity</b> types.
Category	A type in the OCCI Core Model and the basis of the OCCI type identification mechanism. The parent type of <b>Kind</b> .
capabilities	In the context of <b>Entity</b> sub-types <b>capabilities</b> refer to the <b>Attributes</b> and <b>Actions</b> exposed by an <b>entity instance</b> .
Collection	A set of <b>Entity</b> sub-type instances all associated to a particular <b>Kind</b> or <b>Mixin</b> instance.
Entity	An OCCI base type. The parent type of <b>Resource</b> and <b>Link</b> .
entity instance	An instance of a sub-type of <b>Entity</b> but not an instance of the <b>Entity</b> type itself. The OCCI model defines two sub-types of <b>Entity</b> : the <b>Resource</b> type and the <b>Link</b> type. However, the term <i>entity instance</i> is defined to include any instance of a sub-type of <b>Resource</b> or <b>Link</b> as well.
Kind	A type in the OCCI Core Model. A core component of the OCCI classification system.
284 Link	An OCCI base type. A <b>Link</b> instance associates one <b>Resource</b> 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 <b>Mixin</b> type associated with an <i>entity instance</i> . The “mix-in” concept as used by OCCI <i>only</i> applies to instances, never to <b>Entity</b> types.
OCCI	Open Cloud Computing Interface.
OGF	Open Grid Forum.
Resource	An OCCI base type. The parent type for all domain-specific <b>Resource</b> sub-types.
resource instance	See <i>entity instance</i> . This term is considered obsolete.
tag	A <b>Mixin</b> instance with no attributes or actions defined. Used for taxonomic organisation of entity instances.
template	A <b>Mixin</b> 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 <b>Category</b> , <b>Attribute</b> , <b>Kind</b> , <b>Mixin</b> , <b>Action</b> , <b>Entity</b> , <b>Resource</b> and <b>Link</b> .
concrete type/sub-type	A concrete type/sub-type is a type that can be instantiated.
URI	Uniform Resource Identifier.
URL	Uniform Resource Locator.
285 URN	Uniform Resource Name.

## 286 6 Contributors

287 We would like to thank the following people who contributed to this document:

Name	Affiliation	Contact
Michael Behrens	R2AD	behrens.cloud at r2ad.com
Mark Carlson	Toshiba	mark at carlson.net
Augusto Ciuffoletti	University of Pisa	augusto.ciuffoletti at gmail.com
Andy Edmonds	ICCLab, ZHAW	edmo at zhaw.ch
Sam Johnston	Google	samj at samj.net
Gary Mazzaferro	Independent	garymazzaferro at gmail.com
Thijs Metsch	Intel	thijs.metsch at intel.com
288 Ralf Nyren	Independent	ralf at nyren.net
Alexander Papaspyrou	Adesso	alexander at papaspyrou.name
Boris Parák	CESNET	parak at cesnet.cz
Alexis Richardson	Weaveworks	alexis.richardson at gmail.com
Shlomo Swidler	Orchestratus	shlomo.swidler at orchestratus.com
Florian Feldhaus	Independent	florian.feldhaus at gmail.com
Zdeněk Šustr	CESNET	zdenek.sustr at cesnet.cz
Jean Parpaillon	Inria	jean.parpaillon at inria.fr
Philippe Merle	Inria	philippe.merle@inria.fr

289 Next to these individual contributions we value the contributions from the OCCI working group.

## 290 **7 Intellectual Property Statement**

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

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

## 301 **8 Disclaimer**

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

## 305 **9 Full Copyright Notice**

306 Copyright © Open Grid Forum (2009-2016). All Rights Reserved.

307 This document and translations of it may be copied and furnished to others, and derivative works that comment  
308 on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in  
309 whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph  
310 are included on all such copies and derivative works. However, this document itself may not be modified in  
311 any way, such as by removing the copyright notice or references to the OGF or other organizations, except  
312 as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights  
313 defined in the OGF Document process must be followed, or as required to translate it into languages other  
314 than English.

315 The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or  
316 assignees.

## 317 **References**

- 318 [1] S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels,” RFC 2119 (Best Current Practice),  
319 Internet Engineering Task Force, Mar. 1997. [Online]. Available: <http://www.ietf.org/rfc/rfc2119.txt>
- 320 [2] R. Nyren, T. Metsch, and A. Edmonds, “Open Cloud Computing Interface – HTTP Protocol,” Draft,  
321 November 2015. [Online]. Available: TBD
- 322 [3] R. Nyrén, A. Edmonds, A. Papaspyrou, and T. Metsch, “Open Cloud Computing Interface – Core,” Draft,  
323 November 2015. [Online]. Available: TBD
- 324 [4] D. Slik, M. Siefer, E. Hibbard, C. Schwarzer, A. Yoder, L. N. Bairavasundaram, S. Baker, M. Carlson,  
325 H. Nguyen, and R. Ramos, “Cloud data management interface (cdmi) v1.0,” <http://www.snia.org/>,  
326 Apr. 2010. [Online]. Available: [http://www.snia.org/tech\\_activities/standards/curr\\_standards/cdmi/](http://www.snia.org/tech_activities/standards/curr_standards/cdmi/CDMI_SNIA_Architecture_v1.0.pdf)  
327 [CDMI\\_SNIA\\_Architecture\\_v1.0.pdf](http://www.snia.org/tech_activities/standards/curr_standards/cdmi/CDMI_SNIA_Architecture_v1.0.pdf)



## 328 A Change Log

329 The corrections introduced by the February 23, 2016 update are summarized below. This section describes the  
330 possible impact of the corrections on existing implementations and associated dependent specifications.

- 331 ● Outlined expected behavior when replacing `Mixins`, specifically `Resource Template` and `OS Template`
- 332 ● New “save” action for `Compute`
- 333 ● New credentials mixin – allows credentials to be supplied to the creation of a compute resource
- 334 ● New contextualization mixin – allows a script to be supplied with the creation request of a compute  
335 resource
- 336 ● Added error state to all resource state models
- 337 ● Added `occi.compute.share` attribute to `Compute`. This allows for basic support of container virtual-  
338 ization technologies.
- 339 ● Removed `occi.compute.speed` attribute to `Compute`.
- 340 ● Added `state.message` to all infrastructure resources (`Compute`, `Storage`, `Network`, `NetworkInterface`,  
341 `StorageLink`)
- 342 ● Added references to the core model `parent`, `applies` and `depends` for infrastructure `Mixins` and `Kinds`.
- 343 ● Updated figures to reflect new Core model
- 344 ● Updated the storage state model – removes `resize`. Removal of error action from tables. Resize done  
345 through a resource update
- 346 ● Removed `backup`, `snapshot`, `resize` and `degraded` actions from state tables.