

1 Draft
2 OCCI-WG
3
4

Thijs Metsch, Intel
Andy Edmonds, ICCLab, ZHAW
Boris Parák, CESNET
Updated: April 13, 2015

5 **Open Cloud Computing Interface - Infrastructure**

6 Status of this Document

7 This document is a draft including proposed errata updates to the OCCI Infrastructure [1] specification.

8 The errata updates are summarized in section A.

9 Eventually this document will obsolete GFD-P-R.143. This document is fully backward compatible to [1].

10 Copyright Notice

11 Copyright © Open Grid Forum (2009-2015). All Rights Reserved.

12 Trademarks

13 OCCI is a trademark of the Open Grid Forum.

14 Abstract

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

18 Contents

19	1 Introduction	3
20	2 Notational Conventions	3
21	3 Infrastructure	4
22	3.1 Compute	5
23	3.2 Network	5
24	3.2.1 IPNetworking Mixin	6
25	3.3 Storage	7
26	3.4 Linking Infrastructure Resources	8
27	3.4.1 Linking to Network	8
28	3.4.2 Linking to Storage	10
29	3.4.3 Linking to CDMI Managed Storage	10
30	3.5 Infrastructure Templates	11
31	3.5.1 OS Template	11
32	3.5.2 Resource Template	12
33	4 Security Considerations	13
34	5 Glossary	14
35	6 Contributors	14
36	7 Intellectual Property Statement	16
37	8 Disclaimer	16
38	9 Full Copyright Notice	16
39	A Errata	17

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 IaaS¹ 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 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. The OCCI Core Model can be interacted through *renderings* (including associated behaviours) and expanded 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 interoperable 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 [2].

¹Infrastructure 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 utilising 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 and represents a L2 networking resource. This is complimented 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 complimented by an IPNetwork-Interface Mixin.

StorageLink connects a Compute instance to a Storage instance.

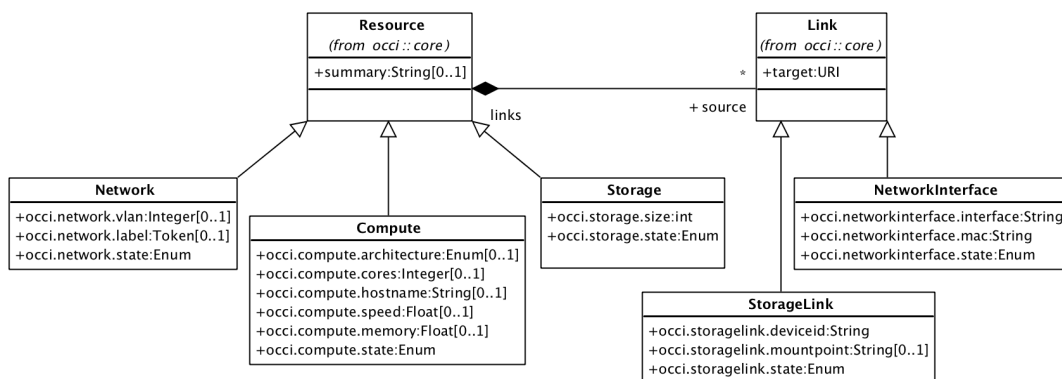


Figure 1. Overview Diagram of OCCI Infrastructure Types.

These infrastructure types inherit the OCCI Core Model Resource base type and all their attributes. The HTTP Rendering document [3] defines how to serialise 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 Resource	<schema>/core#resource
storage	<schema>/infrastructure#	Storage Resource	<schema>/core#resource
storagelink	<schema>/infrastructure#	StorageLink Link	<schema>/core#link
network	<schema>/infrastructure#	Network Resource	<schema>/core#resource
networkinterface	<schema>/infrastructure#	NetworkInterface Link	<schema>/core#link

96 Table 2 describes the Kind instances defined for each of the infrastructure Resource or Link sub-types. For
 97 information on extending these types, please refer to the OCCI Core Model document [4].

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

102 3.1 Compute

103 The Compute type represents a generic information processing resource, e.g. a virtual machine or container.
 104 Compute inherits the Resource base type defined in OCCI Core Model [4]. Compute is assigned the Kind
 105 instance <http://schemas.ogf.org/occi/infrastructure#compute>. A Compute instance MUST use and expose
 106 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 ⁹ (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.

107 Table 3 describes the OCCI Attributes² defined by Compute through its Kind instance. These attributes MAY
 108 or MUST be exposed by an instance of the Compute type depending on the “Multiplicity” column in the
 109 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#> categorisation 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}

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

113 3.2 Network

114 The Network type represents a L2 networking entity (e.g. a virtual switch). It can be extended using the
 115 mixin mechanism (or sub-typed) to support L3/L4 capabilities such as TCP/IP etc. For the purposes of this

²See the “attributes” attribute defined by the Category type and inherited by Kind [4].

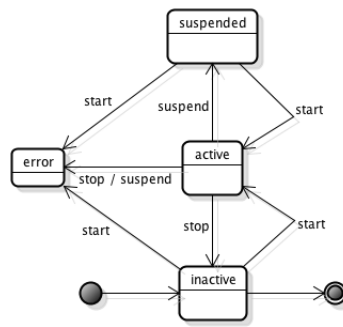


Figure 2. State Diagram for a Compute instance.

116 specification we define an OCCI mixin so that IP networking can be supported where required. Network inherits
 117 the Resource base type defined in OCCI Core Model [4].

118 The Network type is assigned the *http://schemas.ogf.org/occi/infrastructure#network* Kind. A Network
 119 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.

120 Table 5 describes the OCCI Attributes³ defined by Network through its Kind instance. These attributes MAY
 121 or MUST be exposed by an instance of the Network type depending on the “Multiplicity” column in the
 122 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	–

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

3.2.1 IPNetworking Mixin

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

128 The IPNetworking mixin is assigned⁴ the “scheme” of *http://schemas.ogf.org/occi/infrastructure/network#*
 129 and the “term” value *ipnetwork*. An IPNetworking mixin MUST support these values.

130 Table 7 define the attributes introduced by the IPNetworking mixin.

³See the “attributes” attribute defined by the Category type and inherited by Kind [4].

⁴Both assignments use data members from the inherited Category type [4].

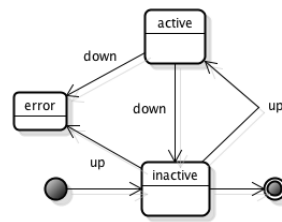


Figure 3. State Diagram for a Network instance.

- 131 The IPNetworking mixin MUST be related to the Network kind by setting the *applies* attribute to:
- 132 <http://schemas.ogf.org/occi/infrastructure#network>.
- 133 A Network instance associated with the IPNetworking mixin Mixin instance MUST implement these attributes.

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

Attribute	Type	Multiplicity	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.

- 134 In Figure 4 a UML object diagram depicts how Network would be associated with an IPNetwork Mixin when
- 135 both are instantiated.

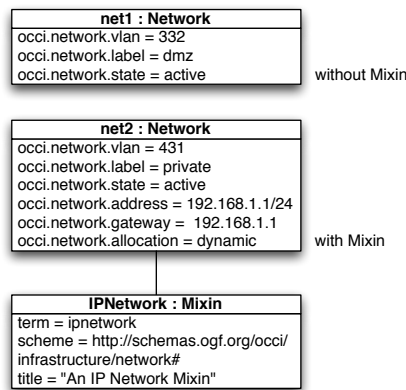


Figure 4. Object Diagram of a Network Instance and its Associated IPNetwork Mixin.

136 3.3 Storage

- 137 The Storage type represent resources that record information to a data storage device. Storage inherits the
- 138 Resource base type defined in the OCCI Core Model [4]. The Storage type is assigned the Kind instance
- 139 <http://schemas.ogf.org/occi/infrastructure#storage>. A Storage instance MUST use and expose this Kind.

140 Table 8 describes the OCCI Attributes⁵ defined by Storage through its Kind instance. These attributes MAY

⁵See the "attributes" attribute defined by the Category type and inherited by Kind [4].

Table 8. Attributes defined for the Storage type.

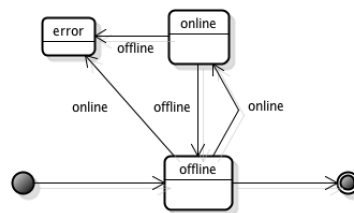
Attribute	Type	Multiplicity	Mutability	Description
occi.storage.size	Float, 10 ⁹ (GiB)	1	Mutable	Storage size in gigabytes of the instance.
occi.storage.state	Enum {online, offline, error}	1	Immutable	Current status of the instance.
occi.storage.state.message	String	0..1	Immutable	Human-readable explanation of the current instance state.

141 or MUST be exposed by an instance of the Storage type depending on the “Multiplicity” column in the
 142 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/#> categorisation scheme. “Action Term” below refers to Action.term.

Action Term	Target State	Attributes
online	online	–
offline	offline	–

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

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

146 OCCI can be used in conjunction with the SNIA cloud storage standard, Cloud Data Management Interface
 147 (CDMI) [5] to provide enhanced management of the cloud computing storage and data. For storage managed
 148 through CDMI, see the section on StorageLink

149 3.4 Linking Infrastructure Resources

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

155 3.4.1 Linking to Network

156 The NetworkInterface type represents an L2 client device (e.g. network adapter). It can be extended using the
 157 mix-in mechanism or sub-typed to support L3/L4 capabilities such as TCP/IP etc. NetworkInterface inherits
 158 the Link base type defined in the OCCI Core Model [4].

159 The NetworkInterface type is assigned the Kind instance <http://schemas.ogf.org/occi/infrastructure#networkinterface>.
 160 A NetworkInterface instance MUST use and expose this Kind. The Kind instance assigned to the Network-

161 Interface type MUST be related to the <http://schemas.ogf.org/occi/core#link> Kind by setting the *parent*
 162 attribute.

Table 10. Attributes defined for the NetworkInterface type.

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

163 Table 10 describes the OCCI Attributes⁶ defined by NetworkInterface through its Kind instance. These attributes
 164 MAY or MUST be exposed by an instance of the NetworkInterface type depending on the “Multiplicity” column
 165 in the aforementioned table. Figure 6 illustrates the state diagram for a NetworkInterface instance.

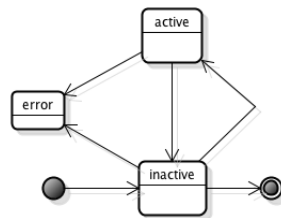


Figure 6. State Diagram for a NetworkInterface instance.

166 **3.4.1.1 IPNetworkInterface Mixin** In order to support L3/L4 capabilities (e.g. IP, TCP etc.) with the
 167 NetworkInterface type, an OCCI Mixin instance is herewith defined.

168 The IPNetworkInterface mixin is assigned⁷ the “scheme” of <http://schemas.ogf.org/occi/infrastructure/networkinterface#>
 169 and the “term” value *ipnetworkinterface*. An IPNetworkInterface mixin MUST support these attributes.

170 The IPNetworkInterface mixin MUST be related to the NetworkInterface kind by setting the *applies* attribute
 171 to:

172 <http://schemas.ogf.org/occi/infrastructure#networkinterface>.

173 Table 11 define the attributes introduced by the IPNetworkInterface mixin. A NetworkInterface instance
 174 associated with the IPNetworkInterface mixin Mixin instance MUST expose these attributes.

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

Attribute	Type	Multi- plicity	Mutability	Description
occi.networkinterface.address	IPv4 or IPv6 Address	1	Mutable	Internet Protocol(IP) network address (e.g. 192.168.0.1/24, fc00::/7) of the link
occi.networkinterface.gateway	IPv4 or IPv6 Address	0..1	Mutable	Internet Protocol(IP) network address (e.g. 192.168.0.1/24, fc00::/7)
occi.networkinterface.allocation	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.

⁶See the “attributes” attribute defined by the Category type and inherited by Kind [4].

⁷Both assignments use data members from the inherited Category type [4].

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

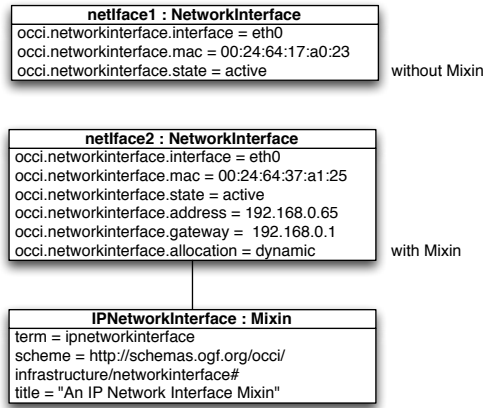


Figure 7. Object Diagram of a NetworkInterface Instance and its Associated IPNetworkInterface Mixin.

177 **3.4.2 Linking to Storage**

178 The StorageLink type represents a link from a Resource to a target Storage instance. This enables a Storage
 179 instance be attached to a Compute instance, with all the prerequisite low- level operations handled by the
 180 OCCI implementation. Storage inherits the Link base type defined in the OCCI Core Model [4].

181 The StorageLink type is assigned the Kind instance *http://schemas.ogf.org/occi/infrastructure#storagelink*. A
 182 StorageLink instance MUST use and expose this Kind. The Kind instance assigned to the StorageLink type
 183 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
occi.storagelink.deviceid	String	1	Mutable	Device identifier as defined by the OCCI service provider.
occi.storagelink.mountpoint	String	0..1	Mutable	Point to where the storage is mounted in the guest OS.
occi.storagelink.state	Enum {active, inactive, error}	1	Immutable	Current status of the instance.
occi.storagelink.state.message	String	0..1	Immutable	Human-readable explanation of the current instance state.

184 Table 12 describes the OCCI Attributes⁸ defined by StorageLink through its Kind instance. These attributes
 185 MAY or MUST be exposed by an instance of the StorageLink type depending on the “Multiplicity” column in
 186 the aforementioned table. Figure 8 illustrates the state diagram for a StorageLink instance.

187 **3.4.3 Linking to CDMI Managed Storage**

188 As previously stated, OCCI can be used in conjunction with the SNIA cloud storage standard, Cloud Data
 189 Management Interface (CDMI) [5] to provide enhanced management of the cloud computing storage and data.
 190 In order to integrate the two, the use of StorageLink should be used. This will link OCCI managed Resources
 191 to CDMI resources. The 'occi.storagelink.deviceid' attribute of StorageLink, defined above, should be set to
 192 the CDMI Object ID of an exported CDMI Container.

⁸See the “attributes” attribute defined by the Category type and inherited by Kind [4].

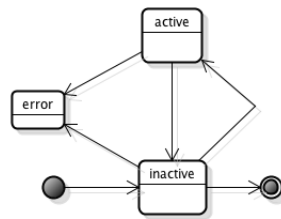


Figure 8. State Diagram for a StorageLink instance.

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 2 supported infrastructure template types in OCCI Infrastructure.

3.5.1 OS Template

OS (Operating System) Templates allow clients 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). Of the two supported template types, this 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 [4].

A 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.

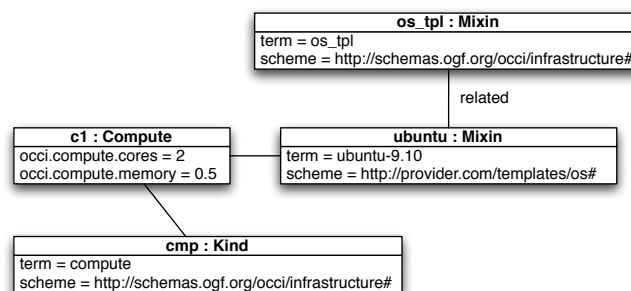


Figure 9. Object Diagram of a Compute Instance and its Associated OS Template Mixin.

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 themselves and so implementation-specific.

217 3.5.2 Resource Template

218 The Resource Template Mixin builds upon the concept of OS Templates. A Resource Template is a provider-
219 defined Mixin instance that refers to a preset Resource configuration.

220 The preset Resource configuration is not visible through the OCCI Discovery mechanism. The Mixin.attributes
221 (inherited from Category) is empty for a Resource Template Mixin. The side-effect of initialising Resource
222 attributes with pre-defined values is handled by the implementation.

223 The OCCI implementation associates a set of Resource attributes (via Category's 'attributes') with a particular
224 term identifier.

225 An implementation-defined Resource Template Mixin MUST be related to the OCCI Resource Template Mixin
226 in order to give absolute type information. This is done by setting the *depends* attribute. The OCCI Resource
227 Template is defined by the Mixin instance http://schemas.ogf.org/occi/infrastructure#resource_tpl and MUST
228 be supported SHOULD Resource Templates be offered by the OCCI implementation.

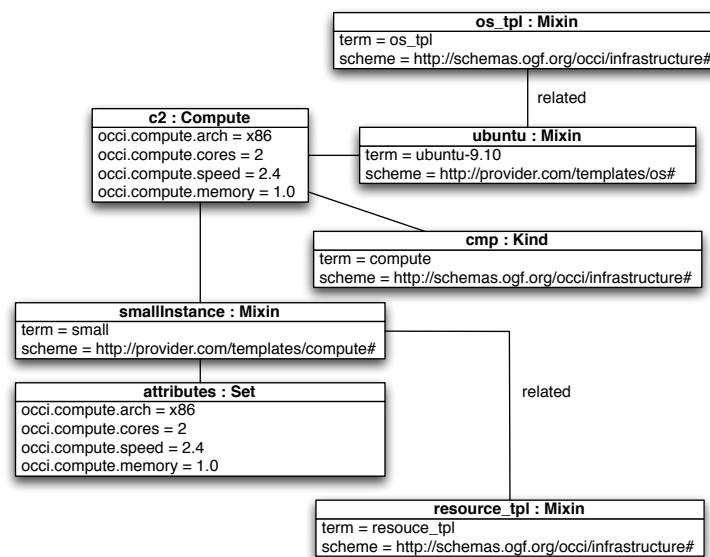


Figure 10. Object Diagram of a Compute Instance and its Associated OS Template Mixin and Resource Template Mixin.

229 A typical example of such a Mixin's use is shown in figure 10) using a UML object diagram. In this example, the
230 provider offers Compute Resources based on different sizes (i.e. small, medium, large). Each "size" of Compute
231 (i.e. the term) corresponds to a predetermined set of OCCI Resource-specific attributes. In the example below a
232 'small' Compute instance is created. Specifying "small" as the term corresponds to an implementation-specific
233 Compute Resource-specific attribute set⁹ that is shown by the object instance named "attributes" in figure 10.

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

236 **3.5.2.1 Credentials Mixin** When creating a Compute Resource a client normally supplies security creden-
237 tials in the form of a public SSH key. This SSH key is injected into the Compute Resource by the provider on
238 the client's behalf. This feature is provided by the Credentials Mixin.

239 If a provider that offers VMs with access secured by SSH then that OCCI implementation SHOULD support
240 this. Otherwise no user supplied public SSH key can be injected into the Compute Resource.

241 The OCCI credentials mixin has the term '*ssh_key*' and the schema '<http://schemas.ogf.org/occi/credentials#>'.

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

244 <http://schemas.ogf.org/occi/infrastructure#compute>.

⁹This attribute set is implementation-specific and is *not* related to Mixin.attributes inherited from the Category type [4].

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

Attribute	Type	Multiplicity	Mutability	Description
occi.credentials.ssh.publickey	String	1	Mutable	The contents of the public key file to be injected into the Compute Resource

245 **3.5.2.2 Contextualisation Mixin** In order to ease automation, OCCI supports the means to execute a
 246 program once Resource has been instantiated. This feature is provided by the contextualisation mixin. On
 247 receipt of the contextualisation data the OCCI implementation MUST distinguish the type of data being
 248 presented and then supply that content to the Compute Resource being instantiated. That content is then
 249 executed by the Compute Resource as the last step in the Compute's boot-order.

250 OCCI implementations SHOULD support this otherwise no contextualisation of a resource instance can be done.
 251 The OCCI contextualisation mixin has the term *user.data* and the schema <http://schemas.ogf.org/occi/compute#>

252 Contextualisation mixin MUST only apply to the Compute Kind and therefore the mixin should have its *applies*
 253 attribute set to:

254 <http://schemas.ogf.org/occi/infrastructure#compute>.

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

Attribute	Type	Multiplicity	Mutability	Description
occi.compute.userdata	String	1	Mutable	Contextualisation data (e.g. script, executable) that the client supplies once and only once. It cannot be updated.

255 4 Security Considerations

256 The OCCI Infrastructure specification is an extension to the OCCI Core and Model specification [4]; thus the
 257 same security considerations as for the OCCI Core and Model specification apply here.

258 5 Glossary

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

261 6 Contributors

²⁶² 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
²⁶³ Gary Mazzaferro	Independent	garymazzaferro at gmail.com
Thijs Metsch	Intel	thijs.metsch at intel.com
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	NetApp	florian.feldhaus at gmail.com

²⁶⁴ Next to these individual contributions we value the contributions from the OCCI working group.

265 **7 Intellectual Property Statement**

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

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

276 **8 Disclaimer**

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

280 **9 Full Copyright Notice**

281 Copyright © Open Grid Forum (2009-2015). All Rights Reserved.

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

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

292 **References**

293 [1] T. Metsch and A. Edmonds, “Open Cloud Computing Interface – Infrastructure,” GFD-P-R.184, April
294 2011. [Online]. Available: <http://ogf.org/documents/GFD.184.pdf>

295 [2] S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels,” RFC 2119 (Best Current Practice),
296 Internet Engineering Task Force, Mar. 1997. [Online]. Available: <http://www.ietf.org/rfc/rfc2119.txt>

297 [3] T. Metsch and A. Edmonds, “Open Cloud Computing Interface – HTTP Rendering,” GFD-P-R.185, April
298 2011. [Online]. Available: <http://ogf.org/documents/GFD.185.pdf>

299 [4] R. Nyrén, A. Edmonds, A. Papaspyrou, and T. Metsch, “Open Cloud Computing Interface – Core,”
300 GFD-P-R.183, April 2011. [Online]. Available: <http://ogf.org/documents/GFD.183.pdf>

301 [5] D. Slik, M. Siefer, E. Hibbard, C. Schwarzer, A. Yoder, L. N. Bairavasundaram, S. Baker, M. Carlson,
302 H. Nguyen, and R. Ramos, “Cloud data management interface (cdmi) v1.0,” <http://www.snia.org/>,
303 Apr. 2010. [Online]. Available: [http://www.snia.org/tech_activities/standards/curr_standards/cdmi/
304 CDMI_SNIA_Architecture_v1.0.pdf](http://www.snia.org/tech_activities/standards/curr_standards/cdmi/CDMI_SNIA_Architecture_v1.0.pdf)

305 **A Errata**

- 306 • New credentials mixin - allows credentials to be supplied to the creation of a compute resource
- 307 • New contextualisation mixin - allows a script to be supplied with the creation request of a compute
308 resource
- 309 • Added error state to all resource state models
- 310 • Added occi.compute.share attribute to Compute. This allows for basic support of container virtualisation
311 technologies.
- 312 • Added state.message to all infrastructure resources (Compute, Storage, Network, NetworkInterface,
313 StorageLink)
- 314 • Added references to the core model parent, applies and depends for infrastructure mixins, kinds
- 315 • Updated figures to reflect new Core model
- 316 • Updated the storage state model - removes resize. removal of error action from tables. resize done
317 through a resource update
- 318 • Removed backup, snapshot, resize and degraded actions from state tables