Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP

An Interface for Managing Cloud Infrastructure

Information for Work-in-Progress version:

IMPORTANT: This specification is not a standard. It does not necessarily reflect the views of the DMTF or all of its members. Because this document is a Work in Progress, this specification may still change, perhaps profoundly. This document is available for public review and comment until the stated expiration date.

It expires on: 2012-05-15

Provide any comments through the DMTF Feedback Portal: http://www.dmtf.org/standards/feedback

Version: 0.0.46a
Status: Work In Progress - not a DMTF Standard
Publication Date: 2011-11-15
Document Number: DSP0263
Copyright © 2011 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. Members and non-members may reproduce DMTF specifications and documents, provided that correct attribution is given. As DMTF specifications may be revised from time to time, the particular version and release date should always be noted.

Implementation of certain elements of this standard or proposed standard may be subject to third party patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose, or identify any or all such third party patent right, owners or claimants, nor for any incomplete or inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize, disclose, or identify any such third party patent rights, or for such party’s reliance on the standard or incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any party implementing such standard, whether such implementation is foreseeable or not, nor to any patent owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified after publication, and shall be indemnified and held harmless by any party implementing the standard from any and all claims of infringement by a patent owner for such implementations.

For information about patents held by third-parties which have notified the DMTF that, in their opinion, such patent may relate to or impact implementations of DMTF standards, visit http://www.dmtf.org/about/policies/disclosures.php.
Abstract

This document is a deliverable from the DMTF Cloud Management Working Group. It defines a logical model for the management of resources within the Infrastructure as a Service domain. This model was developed to address the use cases outlined in the “Scoping Framework for Cloud Management Models and Protocol Requirements” document.

Acknowledgments

TBD
CONTENTS

1 Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP .................. 1
2 CONTENTS ............................................................................................................................... 4
3 FIGURES ................................................................................................................................. 7
4 1 Scope .................................................................................................................................. 8
   1.1 Document Structure ......................................................................................................... 8
   1.2 Typographical Conventions ............................................................................................... 8
5 2 References ............................................................................................................................. 8
6 3 Terms and Definitions ............................................................................................................ 10
7   3.1 Authentication .................................................................................................................. 10
8   3.2 Authorization .................................................................................................................... 10
9   3.3 Cloud Service Consumer .................................................................................................. 10
10   3.4 Cloud Service Provider .................................................................................................. 10
11   3.5 Configuration .................................................................................................................. 11
12   3.6 Message Confidentiality ................................................................................................. 11
13   3.7 Message Integrity ............................................................................................................. 11
14   3.8 Template ......................................................................................................................... 11
15 4 REST/HTTP Protocol ........................................................................................................... 11
16   4.1 Protocol Definition ........................................................................................................ 11
17        4.1.1 Protocol Security ................................................................................................... 12
18        4.1.2 XML Namespaces .................................................................................................. 12
19        4.1.3 URI Space .............................................................................................................. 12
20        4.1.4 Media Types .......................................................................................................... 12
21        4.1.5 Request Headers .................................................................................................... 12
22        4.1.6 Request Parameters ............................................................................................... 13
23        4.1.7 Response Headers ................................................................................................ 14
24        4.1.8 HTTP Status Codes .............................................................................................. 14
25        4.1.9 Serialization of References ..................................................................................... 16
26        4.1.10 Serialization of Arrays ......................................................................................... 16
27   4.2 Protocol Resource Operations ...................................................................................... 16
28        4.2.1 Operational Principles ........................................................................................ 17
29        4.2.2 Common CRUD (Create Read Update and Delete) Operations ............................. 19
30 5 Model .................................................................................................................................... 23
31        5.1 Identifiers ................................................................................................................. 24
32        5.2 Attribute "Properties" ............................................................................................... 24
33        5.3 Data Types and Values ............................................................................................ 24
34        5.4 Relationship Semantics ............................................................................................ 25
35        5.5 Alternative Model Formats ....................................................................................... 25
5.12.7 VSP (Virtual Switch Port) Template .......................................................... 87
5.12.8 VSP (Virtual Switch Port) Template Collection ........................................... 88
5.12.9 VSP (Virtual Switch Port) Configuration ....................................................... 89
5.12.10 VSP (Virtual Switch Port) Configuration Collection .................................... 90
5.12.11 VSP (Virtual Switch Port) ................................................................. 91
5.12.12 VSP (Virtual Switch Port) Collection ......................................................... 93
5.12.13 Routing Group ........................................................................................ 95
5.12.14 Routing Group Collection ........................................................................ 96
5.13 Monitoring Entities and Relationships ............................................................. 98
5.13.1 Job ............................................................................................................... 99
5.13.2 Job Collection ............................................................................................ 101
5.13.3 Meter Template ......................................................................................... 102
5.13.4 Meter Template Collection ....................................................................... 104
5.13.5 Meter ......................................................................................................... 105
5.13.6 Meter Collection ........................................................................................ 108
5.13.7 Event Log .................................................................................................. 110
5.13.8 Event Log Collection ................................................................................ 112
5.13.9 Event ......................................................................................................... 113
5.13.10 Event Collection ....................................................................................... 114
6 Scenarios ............................................................................................................. 115
6.1 Initial Scenario ............................................................................................... 115
6.1.1 Create and deploy a Machine using a Provider created Machine Template .... 116
6.1.2 Create a Machine by passing a Machine Template by value ....................... 116
6.1.3 Create a Machine using a Consumer created Machine Template ................ 117
6.1.4 Create a Machine Template by specifying individual components ............ 117
6.1.5 Create a Machine Template from a template file ...................................... 117
6.1.6 Control Machine State (CMWG065) ......................................................... 117
6.2 Machine Image Scenarios .............................................................................. 118
6.2.1 Create new Machine Image from an image file ........................................... 118
6.2.2 Create new Machine Image from Machine instance .................................. 118
6.3 System Scenario ............................................................................................. 118
6.3.1 List System Templates (CMWG010) ......................................................... 118
6.3.2 Create System Template ............................................................................ 118
6.3.3 Create and Deploy a System to a Site Using a System Template (CMWG017/CMWG035) ................................................................. 118
6.3.4 Get Monitoring Information (CMWG066) ................................................ 119
6.3.5 Control System State (CMWG065) ............................................................ 119
6.3.6 Remove System from a Site (CMWG051) .................................................. 119
7 Security .............................................................................................................. 119
7.1 API Level Security ........................................................................................................... 119
  7.1.1 Authentication ......................................................................................................... 119
  7.1.2 Message Integrity .................................................................................................... 119
  7.1.3 Message Confidentiality ........................................................................................ 120
  7.1.4 Authorization .......................................................................................................... 120
  7.1.5 Multi-Tenancy ......................................................................................................... 120
7.2 Resource Level Credentials ......................................................................................... 120

FIGURES

Figure 1 - System Entities .................................................................................................. 34
Figure 2 - Machine Entities .............................................................................................. 43
Figure 3 - Volume Entities ................................................................................................. 66
Figure 4 - Network Entities .............................................................................................. 78
Figure 5 - Monitoring Entities ........................................................................................ 99
1 Scope

Quoting the “Architecture for Managing Clouds White Paper”:

Any programmatic API has an underlying resource model, whether implicit or explicit. In the IT
management domain, the practice has long been to make resource models explicit and clearly
separated from the protocols used to manipulate model elements.

This document describes an abstract, service offering model by defining a set of logical entities that are
shared between consumers and service providers.

1.1 Document Structure

This document defines a model and an HTTP/REST-based protocol.

The core REST pattern is defined first and, after each entity is defined, any REST-specific information for
that entity will be specified.

1.2 Typographical Conventions

This specification uses the following syntax to define the serialization of resources:

- Values in *italics* indicate data types instead of literal values.
- Characters are appended to items to indicate cardinality:
  - "?" (0 or 1)
  - "*" (0 or more)
  - "+" (1 or more)
- Ellipses (i.e., "...") indicate points of extensibility. Note that the lack of an ellipses does not mean
  no extensibility point exists, rather it is just not explicitly called out - usually for the sake of
  brevity.

2 References

The following referenced documents are indispensable for the application of this document. For dated
references, only the edition cited applies. For undated references, the latest edition of the referenced
document (including any amendments) applies:

  units – Part 13: Information science and technology*, April 2008,
- **IETF RFC 2045**, N. Freed et al, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of
- **IETF RFC 2616**, R. Fielding et al, *Hypertext Transfer Protocol -- HTTP/1.1*, June 1999,
Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP


Note: the CIMI-RNG document is not yet available
3 Terms and Definitions

In this document, some terms have a specific meaning beyond the normal English meaning. Those terms are defined in this clause.

The terms "shall" ("required"), "shall not," "should" ("recommended"), "should not" ("not recommended"), "may," "need not" ("not required"), "can" and "cannot" in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Annex H. The terms in parenthesis are alternatives for the preceding term, for use in exceptional cases when the preceding term cannot be used for linguistic reasons. Note that ISO/IEC Directives, Part 2, Annex H specifies additional alternatives. Occurrences of such additional alternatives shall be interpreted in their normal English meaning.

The terms "clause," "subclause," "paragraph," and "annex" in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 5.

The terms "normative" and "informative" in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 3. In this document, clauses, subclauses, or annexes labeled "(informative)" do not contain normative content. Notes and examples are always informative elements.

3.1 Authentication

The process of verifying a claim, made by a subject, that it should be allowed to act on behalf of a given principal (person, service, etc.). Typical authentication mechanisms involved the use of username/password combination or public/private key pairs.

3.2 Authorization

(also known as Access Control) The process of verifying that an authenticated principal (person, service, etc.) has permission to perform certain operations (e.g. read, update) on specific resources.

3.3 Cloud Service Consumer

A category of actors that includes the Consumer Business Manager (who approves business and financial expenditures for consumed services, accounts for used service instances, establishes business relationships; sets up accounts, budget, and terms; etc.), the Consumer Service Administrator (who requests service instances and changes to service instances, purchase services within the business relationship; create Service Users (including policies), allocate resources, such as compute and storage, generate reports (usage), etc.), and Service Users (who uses service instances provided by a Cloud Service Provider). The term "Consumer" is used when the indicated action or activity could involve one or more of the above actors. In cases where the distinction between the actors in this category is relevant, the more detailed term will be used.

3.4 Cloud Service Provider

A category of actors that includes the Service Operations Manager (who manages the technical infrastructure required for providing cloud services, monitors and measures performance and utilization against SLAs, provides reports from monitoring and measurement, etc.), Service Business Manager (who offers all types of services developed by cloud service developers, accounts for services potentially offered by service providers themselves and services offered on behalf of cloud service developers, establishes a portfolio of business relationships, and sets up accounts and terms for Consumers, etc.), and Service Transition Manager (who enables a customer to use the cloud service, including "onboarding", integration, and process adoption, defines and creates service offerings based on Templates and Configurations that can be used by Consumers and are populated into the catalog, etc.). The term "Provider" is used when the indicated action or activity could involve one or more of the above actors. In cases were the distinction between the actors in the category is relevant, the more detailed term will be used.
3.5 Configuration

A Configuration is a set of metadata, the values of which serve as the parameters of a discrete conformation of a specific type of virtual resource. For example, a Machine Configuration may define a Machine with the equivalent of a 2.66 GHz processor, 4 GB of memory, and 320 GB of local disk storage.

3.6 Message Confidentiality

A quality of a message which prevents anyone but the intended receiver(s) from viewing its contents.

3.7 Message Integrity

A quality of a message which allows a receiver of that message to determine if the contents of the message have been altered since its creation.

3.8 Template

A Template is the entity that represents the set of metadata and instructions used to instantiate resources (e.g. a Machine Template is used to create Machines). Templates may aggregate other metadata entities such as other Templates, Configurations and Images. For example, a Machine Template refers to a Machine Configuration and a Machine Image.

How a specific protocol mapping, or implementation, chooses to supply Templates as inputs to the instantiation process may vary. However, some common patterns should be considered:

1. By reference - allow Consumers to reference a Template (that exists as an entity in the Provider) as part of the instantiation operation.

2. By value - allow Consumers to dynamically provide the Template information as part of the instantiation operation.

3. Reference with overrides - allow Consumers to reference a Template (that exists as an entity in the Provider) and provide additional values that override the attributes of that Template as part of the instantiation operation.

4. REST/HTTP Protocol

4.1 Protocol Definition

All operations are based on the HyperText Transfer Protocol, version 1.1 [RFC2616]. Each request is sent using an HTTP verb such as PUT, GET, DELETE, HEAD or POST and includes a message body in either JSON or XML format. Each response uses a standard HTTP status code, overloaded with semantics by the context of the particular request that was made. Each entity in the model has a MIME standard ContentType that further contextualizes the operation requests and responses.

The entities in the model are identified by URIs and each entity's representation MUST contain a "uri" property that acts as a "self pointer". This URI SHALL be unique within the context of the Provider's implementation. Dereferencing (via an HTTP GET) the URI of an entity will yield a representation of the entity containing attributes and links to associated entities. To begin operations, a client must know the URI to the main entry point of a Cloud Provider - also known as the "Cloud Entry Point" entity. All other entities within the environment shall then be discoverable via the iterative following of links to associated resource within each resource retrieved.
4.1.1 Protocol Security

Cloud Providers SHALL support secure HTTP connections using TLS. Cloud Providers MAY support non-secure HTTP connections. TLS 1.0, which shall be implemented, is specified in [RFC2246], and the TLS 1.1 and TLS 1.2 should be implemented as specified in [RFC4346] and [RFC5246], respectively.

To ensure a minimum level of security and interoperability between implementations, all CIMI clients and servers shall support the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA cipher suite (hexadecimal value {0x0013}), which is also the mandatory cipher suite for TLS 1.0 (see [RFC2246] Section 9, Mandatory Cipher Suites), as well as the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite (hexadecimal value {0x002F}) shall be implemented, which is the mandatory cipher suite for both TLS 1.1 and TLS 1.2.

Finally, the TLS_RSA_WITH_AES_128_CBC_SHA256 cipher suite (hexadecimal value {0x003C}) should be included with all recommended TLS 1.2 implementations to meet the transition to a security strength of 112 bits (guidance is provided in NIST Special Publication 800-57 [NIST 800-57] and NIST Special Publication 800-131A [NIST 800-131A]). Implementers are free to include additional cipher suites, but must prefer the mandatory ones in negotiation.

4.1.2 XML Namespaces

The following table lists the XML namespaces that are used in this specification. The choice of any namespace prefix is arbitrary and not semantically significant.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>XML Namespaces</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>cimi</td>
<td><a href="http://www.dmtf.org/cimi">http://www.dmtf.org/cimi</a></td>
<td>This specification</td>
</tr>
</tbody>
</table>

4.1.3 URI Space

While URIs returned by providers are to be treated as opaque by consumers, and consumers MUST NOT make assumptions about the layout of the URIs or the structures of the URIs of the resources, consumer may augment URIs with any well-defined query parameters which are supported by the provider as defined in section 4.1.6. Providers shall not use the CIMI-defined query parameter reserved namespace (i.e. names starting with "CIMI").

4.1.4 Media Types

In this specification, resource representations and request bodies are encoded in either JSON, as specified in [RFC4627] or in XML.

Each type of resource has its own media-type, which matches the pattern application/\text{Xxxxx}+json, where "Xxxxx" represents the portion of the identifier unique to a particular representation format for each resource (entity in the model). The identifier MUST either be a DMTF standard identifier as defined in this specification and as registered in accordance to [RFC4288], or it must be a vendor specific identifier that is globally unique (vendor extension).

The server implementation shall provide representations of all resources available in both JSON and XML as specified herein. The client implementation may thus use either JSON or XML to communicate with any server implementation.

4.1.5 Request Headers

This specification uses general-header, request-header, and entity-header headers as defined in HTTP 1.1 [RFC2616] in request messages to provide metadata about the message. Applications using messages defined in this specification shall use headers consistent with HTTP 1.1.
In addition to headers defined in HTTP 1.1, request messages may include a header defined by this specification to indicate the set of allowable versions of the CIMI API that server shall use to process the message.

```
api-version = "1.0"
```

The header allows for a list of `api-version` values to be specified (separated by commas). When more than one value is present the server shall choose one of those versions of the specification to process the message. Clients including more than one value are indicating that any of the specified values are acceptable.

If the server is unable to support any of the specified versions then it shall generate a fault and not process the message. Absence of this header indicates that the server may choose any version of this specification to process the message.

### 4.1.6 Request Parameters

The client can use request parameters in requests to formulate the following:

*Editors Note: These are example URL parameters for requests that apply across all resource types. This will need to be revisited down the road when we decide what support for this we need.*

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>?CIMISelect=attr1, attr2,…</td>
<td>Comma separated attribute names indicate that a subset of the resource is being identified. If an attribute is not part of the resource, then it would be ignored. If none of the attributes is part of the resource, then the resource would be returned in its complete form. The CIMISelect query parameter name may appear more than once in URI which is semantically equivalent to all of the attribute names appearing as values of a single CIMISelect query parameter. For example: ?CIMISelect=name&amp;CIMISelect=state is equivalent to: ?CIMISelect=name,state</td>
<td>MyMachine?CIMISelect=name,description,state</td>
</tr>
<tr>
<td>?[collapse]</td>
<td>This would collapse all the Collection attributes by not returning the individual elements</td>
<td>Only the Collection’s uri, name, and total would be returned for all the attributes that are of Collection type</td>
</tr>
<tr>
<td>?&lt;attr1&gt;:[collapse]</td>
<td>This would return only attr1, and if attr1 is a Collection, it would be collapsed. If attr1 is not a Collection, it would be ignored</td>
<td>?servers:[collapse] Would return Collection’s uri, name, and total only</td>
</tr>
<tr>
<td>?[verbose]</td>
<td>This would show all the fields of all the attributes, recursively, including the</td>
<td>For example, /assembly123?[verbose] would</td>
</tr>
</tbody>
</table>
collections | return the expanded list of all the components, including all the attributes

383 The client must URL encode the request parameters.

### 4.1.7 Response Headers

This specification uses general-header, response-header, and entity-header headers as defined in HTTP 1.1 [RFC2616] in response messages to provide metadata about the message. Applications using messages defined in this specification shall use headers consistent with HTTP 1.1.

In addition to headers defined in HTTP 1.1, response messages shall include a header defined by this specification to indicate the version of the CIMI API that the server used to process the message.

```
X-CIMI-Specification-Version = "X-CIMI-Specification-Version" "api-version
api-version = "1.0"
```

Additionally, if the server supports the Job entity then response messages shall include a header defined by this specification to indicate the URI for the job created to process the associated request message.

```
X-CIMI-Job-URI = "X-CIMI-Job-URI" string
```

### 4.1.8 HTTP Status Codes

Server implementations will return standard HTTP response codes as described in the following table, under the conditions listed in the description.

Editors Note: These are changes from the basic HTTP semantics that are overloaded for the whole specification. We will remove any status codes that are standard HTTP without overloading.

<table>
<thead>
<tr>
<th>HTTP Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 Continue</td>
<td>The client SHOULD continue with its request. This interim response is used to inform the client that the initial part of the request has been received and has not yet been rejected by the platform. The client SHOULD continue by sending the remainder of the request or, if the request has already been completed, ignore this response.</td>
</tr>
<tr>
<td>200 OK</td>
<td>The request was successfully completed. If this request created a new resource that is addressable with a URI, and a response body is returned containing a representation of the new resource, a 200 status will be returned with a Location header containing the canonical URI for the newly created resource.</td>
</tr>
</tbody>
</table>
| 201 Created | A request that created a new resource was completed, and no response body containing a representation of the new resource is being returned. A Location header containing the canonical URI for the newly created resource will be returned. Per the HTTP/1.1 specification:

    The origin server MUST create the resource before returning the 201 status code. If the action cannot be carried out immediately, the server SHOULD respond with 202 (Accepted) response instead. |
<table>
<thead>
<tr>
<th>Status Code</th>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>202 Accepted</td>
<td>The request has been accepted for processing, but the processing has not been completed. Per the HTTP/1.1 specification, the returned entity (if any) SHOULD include an indication of the request's current status. A Location header containing the canonical URI for the not-yet completed resource would be returned along with the Status attribute indicating its progress. If a service implementing this specification supports the Job entity then it SHOULD return the representation of a Job entity in the HTTP body of the response and shall include a 'X-CIMI-Job-URI' HTTP header indicating the URI of the Job entity itself. Per the HTTP/1.1 specification: The entity returned with this response SHOULD include an indication of the request's current status and either a pointer to a status monitor or some estimate of when the user can expect the request to be fulfilled.</td>
<td></td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>The request could not be processed because it contains missing or invalid information (such as validation error on an input field, a missing required value, and so on).</td>
<td></td>
</tr>
<tr>
<td>401 Unauthorized</td>
<td>The authentication credentials (TBD) included with this request are missing or invalid.</td>
<td></td>
</tr>
<tr>
<td>403 Forbidden</td>
<td>The server recognized your credentials, but you do not possess authorization to perform this request.</td>
<td></td>
</tr>
<tr>
<td>404 Not Found</td>
<td>The request specified a URI of a resource that does not exist.</td>
<td></td>
</tr>
<tr>
<td>405 Method Not Allowed</td>
<td>The HTTP verb specified in the request (DELETE, GET, HEAD, POST, PUT) is not supported for this request URI. This is used in the create/update/delete of MachineConfiguration and MachineImages to indicate that the provider only supports a fixed set of immutable entities.</td>
<td></td>
</tr>
<tr>
<td>406 Not Acceptable</td>
<td>The resource identified by this request is not capable of generating a representation corresponding to one of the media types in the Accept header of the request.</td>
<td></td>
</tr>
<tr>
<td>409 Conflict</td>
<td>A creation or update request could not be completed, because it would cause a conflict in the current state of the resources supported by the platform. This is used in MachineTemplate create/update to indicate that the MachineConfiguration cannot support the given MachineImage, for example.</td>
<td></td>
</tr>
<tr>
<td>410 Gone</td>
<td>The requested resource is no longer available at the server and no forwarding address is known. This condition is expected to be considered permanent. Clients with link editing capabilities SHOULD delete references to the Request-URI after user approval. If the server does not know, or has no facility to determine, whether or not the condition is permanent, the status code 404 (Not Found) SHOULD be used instead. This response is cacheable unless indicated otherwise.</td>
<td></td>
</tr>
<tr>
<td>412 Precondition Failed</td>
<td>The precondition given in one or more of the request-header fields evaluated to false when it was tested on the server. This response code allows the client to place preconditions on the current resource meta-information (header field data) and thus prevent the requested method from being applied to a resource other than the one intended.</td>
<td></td>
</tr>
</tbody>
</table>
# Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP

## 4.1.9 Serialization of References

References, as indicated by the type 'ref' in the model, are defined to be URIs in the REST/HTTP protocol mapping. In the JSON serialization they will appear as type "string" and in the XML serialization they will appear as type "xs:anyURI".

References in both JSON and XML have an extensibility point that allows for additional information (such as the target resource to be included "by value") if supported. For example, a reference to a Volume in this specification will appear like this in JSON:

```json
  "volume": { "href": string }
```

and this in XML:

```xml
  <volume href="xs:anyURI"/>
```

For convenience the JSON and XML, as shown above, excludes the implicit extensibility points that would allow for the attributes of the target Volume to be included if desired. So, technically the above should be written as:

```json
  "volume": { "href": string, ... }
```

and this in XML:

```xml
  <volume href="xs:anyURI"> xs:any </volume>
```

however, for brevity they are excluded.

## 4.1.10 Serialization of Arrays

Within this specification, arrays in JSON are serialized with a wrapper property. When serializing arrays, conformant implementations SHALL NOT include empty arrays (i.e. arrays that contain no child properties) in the JSON serialization. For example, an array of references to a list of Volumes attached to a Machine is serialized as:

```json
  "volumes": [ 
    { "volume": { "href": string }, 
      "attachmentPoint": string, 
      "protocol": string } + 
  ],
```

Notice that the child of the "volumes" property is defined with a "+", meaning at least one child is required. This is done to ensure that the JSON serialization is minimized and only includes the wrapping "volumes" element if, and only if, there are volumes.

## 4.2 Protocol Resource Operations

This section defines the set of common REST/HTTP operations that a Cloud Provider might expose. At its core there are four basic CRUD (Create, Read, Update and Delete) operations. The manner in which these are used is consistent across all resources within the model; therefore, their use is defined once.
and is to be applied consistently. Some resources support specialized operations that do not fit well into
a CRUD style of operation and those will all follow a similar high-level pattern but each operation is
allowed to have slight variations to accommodate its specific needs. The specifics of these special
operations are detailed within the section that defines the resource.

When appropriate some of the resource representations will include "operation" properties. These either
provide URI references that can be used to perform operations on the resource, or they are URI
references to other resources that are related to the current resource. Providers shall only include
"operation" properties when the specified operation or related resource is accessible to the current client
for that particular resource. This means that based on many factors (e.g. authorization rights of the
clients, current state of the resource, etc.) a different set of "operation" properties might be returned on
each serialization of the resource.

4.2.1 Operational Principles

4.2.1.1 Resource Navigation

The retrieval of the representation of a Resource using (GET <ResourceURI>) shall return the attributes
of the resource; these attributes might include a set of references to related resources. In that case, it is
possible to obtain every related resource by repeatedly applying the GET method on the retrieved
references.

Example:

If a resourceX contains an attribute “attrA” of string type and an attribute “attrB”, where the latter
references resourceY, the operation:

```
GET <ResourceURI_X> HTTP/1.1
Host: ...
Accept: application/CIMI...
X-CIMI-Specification-Version: 1.0
```

returns a message containing the following:

JSON serialization:

```json
{ "attrA": "hello",
 "attrB": { "href": "http://example.com/uriB" }
}
```

XML serialization:

```xml
<Resource_X xmlns="http://www.dmtf.org/cimi">
 <attrA> hello </attrA>
 <attrB href="http://example.com/uriB" />
</Resource_X>
```

Following the attrB uri ("uriB"), the operation:

```
GET <uriB> HTTP/1.1
Host: ...
Accept: application/CIMI...
X-CIMI-Specification-Version: 1.0
```

returns a message containing the following:

JSON serialization:

```json
{ "attrY": "bye" }
```
XML serialization:

```xml
<Resource_Y xmlns="http://www.dmtf.org/cimi">
  <attrY> bye </attrY>
</Resource_Y>
```

Notes:

1) It is possible that the retrieved reference obtained with the GET <ResourceURI> operation does not directly refer to a related resource, but to a list of homogeneous related resources. In such case, the result of the GET operation will be a list of URIs, and thus the result of a further GET on one of such URIs will return the related resource.

2) this makes it possible to navigate the CIMI resource hierarchy with just the knowledge of the root URI of the Cloud Provider.

4.2.1.2 Operations on a Resource

When it is possible to execute specific operations on the resource, then the response to the GET method on the resource URI shall contain information to perform such operations, in particular:

1. The operation name (typology) such as add, delete, edit, start, stop are described using the rel attribute in the "operation" element.

2. The URI to perform the above mentioned operation.

It is possible to understand that such a field isn’t a link to a related resource but instead is an operation by the presence of the attribute “operations” in JSON or the element “operation” in XML.

The operation shall be performed by invoking the REST specific function on the specified URI.

Example:

If for the resource X an operation is provided to edit that resource, the Get <ResourceURI> response message will contain the following:

JSON serialization:

```json
{ "operations": [ { "rel": "edit", "href": "editURI" } ] }
```

XML serialization:

```xml
<Resource_X xmlns="http://www.dmtf.org/cimi">
  <operation rel="edit" href="editURI"/>
</Resource_X>
```

In this example the operation will be performed with the HTTP PUT on editURI, as follows:

```
PUT <editURI> HTTP/1.1
Host: ...
Accept: application/CIMI-
Content-Type: application/CIMI-
X-CIMI-Specification-Version: 1.0

<serialization of request to update the resource>
```
4.2.2 Common CRUD (Create Read Update and Delete) Operations

Each of the resources supported by this protocol will adhere to the interaction patterns defined in the following sections. Section 5 then defines resource specific information such as the serialization of each resource's properties and which specific actions are supported.

4.2.2.1 Creating a new Resource

To create a new instance of a resource type, an HTTP POST request is sent to a designated "addURI" for that resource type. In many cases, the Collection resource that maintains, or groups, all instances of that resource type will contain and "addLink" property which contains the "addURI" that is to be used.

The request will be of the following form:

```plaintext
POST <addURI> HTTP/1.1
Host: ...
Accept: application/CIMI-...
Content-Type: application/CIMI-...
X-CIMI-Specification-Version: 1.0

<serialization of request to create a new resource>
```

The following provides additional constraints on the request message:

- **X-CIMI-Specification-Version**: This optional HTTP header specifies the list of versions of this specification that the server shall choose from to process this message.

The response will be of the following form:

```plaintext
HTTP/1.1 201 Created
Location: ...
Content-Type: application/CIMI-...
X-CIMI-Specification-Version: 1.0

<serialization of new resource>
```

The following provides additional constraints on the response message:

- **X-CIMI-Specification-Version**: This REQUIRED HTTP header specifies the version of this specification that was used to process this message.

The HTTP response will also include a status code, as described in the following table:

<table>
<thead>
<tr>
<th>HTTP Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201 Created</td>
<td>The new resource was created</td>
</tr>
<tr>
<td>202 Accepted</td>
<td>The resource is in the process of being created. Investigate Job to determine the current status of the operation.</td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>Invalid parameter or field names in the request.</td>
</tr>
<tr>
<td>401 Unauthentic</td>
<td>Incorrect or missing authentication credentials.</td>
</tr>
<tr>
<td>403 Unauthorized</td>
<td>Client lacks the proper authorization to perform this request.</td>
</tr>
</tbody>
</table>

4.2.2.2 Reading a Resource

To retrieve the representation of resource, an HTTP GET request is sent to the URI of that resource.
The request will be of the following form:

```
GET <ResourceURI> HTTP/1.1
Host: ...
Accept: application/CIMI...
X-CIMI-Specification-Version: 1.0
```

The following provides additional constraints on the request message:

**X-CIMI-Specification-Version**
This optional HTTP header specifies the list of versions of this specification that the server shall choose from to process this message.

The response will be of the following form:

```
HTTP/1.1 200 OK
Content-Type: application/CIMI...
X-CIMI-Specification-Version: 1.0
```

The following provides additional constraints on the response message:

**X-CIMI-Specification-Version**
This REQUIRED HTTP header specifies the version of this specification that was used to process this message.

The HTTP response will also include a status code, as described in the following table:

<table>
<thead>
<tr>
<th>HTTP Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Unauthenticated (Incorrect or missing authentication credentials).</td>
</tr>
<tr>
<td>403</td>
<td>Unauthorized (Client lacks the proper authorization to perform this request).</td>
</tr>
</tbody>
</table>

**4.2.2.3 Updating a Resource**

To update the representation of a resource, an HTTP PUT request is sent to a designated "editURI" for that resource type. In many cases, this "editURI" will be the same as the URI of resource itself - retrieving the resource representation MUST include an "editLink" property, which contains the "editURI" that is to be used, if the requester is allowed to modify the resource.

While processing a PUT request if the server detects that an attempt is being made to update a read-only, or immutable, attribute then it SHALL silently ignore that attribute update request and SHALL NOT generate an error. This applies to resource partial updates as well.

Due to potential conflicts that might occur due to multiple concurrent updates, Consumers should use the partial update mechanism, defined in 4.2.2.3.1, to avoid mistakenly updating attributes with out-of-date data.

The request will be of the following form:

```
PUT <editURI> HTTP/1.1
Host: ...
Accept: application/CIMI...
Content-Type: application/CIMI...
X-CIMI-Specification-Version: 1.0
```

The following provides additional constraints on the request message:
X-CIMI-Specification-Version
This optional HTTP header specifies the list of versions of this specification that the server shall choose from to process this message.

The response will be of the following form:

```plaintext
HTTP/1.1 200 OK
Content-Type: application/CIMI...
X-CIMI-Specification-Version: 1.0
<serialization of updated resource>
```

The following provides additional constraints on the response message:

X-CIMI-Specification-Version
This REQUIRED HTTP header specifies the version of this specification that was used to process this message.

The HTTP response message body shall include the updated version of the resource representation.

The HTTP response will also include a status code, as described in the following table:

<table>
<thead>
<tr>
<th>HTTP Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202 Accepted</td>
<td>The resource is in the process of being created. Investigate Job to determine the current status of the operation.</td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>Invalid parameter or field names in the request.</td>
</tr>
<tr>
<td>401 Unauthorized</td>
<td>Incorrect or missing authentication credentials.</td>
</tr>
<tr>
<td>403 Unauthorized</td>
<td>Client lacks the proper authorization to perform this request.</td>
</tr>
</tbody>
</table>

4.2.2.3.1 Partial Updates to a Resource
To update only certain top-level attributes of a resource a consumer MAY do so by including only the changes attributes in the representation of the resource within the HTTP request body. When this is done the URI to the resource SHALL include the attributes to be modified as a comma separated list of query parameters - in other words the URI will be of the form:

```
http://example.com/resource?CIMISelect=attribute1,attribute2,...
```

Only the attributes listed in the URI’s query parameters will be modified; attributes not listed in the URI are not directly modified by the request. Note that this does not preclude the modification of one attribute causing side-effects that result in the modification of an attribute not listed in the query parameters.

Any attribute listed in the URI but not included within the HTTP request body are reset to a resource specific value (e.g. removed).

From an HTTP perspective, the updated subsetted resource is a distinct one; the semantics of a normal HTTP PUT are adhered to - it is a complete replacement update of the specified resource. From the Consumer's perspective, the partial update is interpreted and executed by the Cloud Service Provider, and some part of the resource is/are changed.

For example, the following request will update just the name and description attributes of a Machine:

```
PUT /machines/myMachine?CIMISelect=name,description HTTP/1.1
Host: ...
Accept: application/CIMI-Machine
```
In this example, the "name" attribute is set to "My New Machine" and the "description" attribute is erased.

### 4.2.2.4 Deleting a Resource

To delete a resource, an HTTP DELETE request is sent to a designated "deleteURI" for that resource type. In many cases, this "deleteURI" will be the same as the URI of resource itself - retrieving the resource representation MUST include a "deleteLink" property, which contains the "deleteURI" that is to be used, if the requester is allowed to delete the resource.

The request will be of the following form:

```plaintext
DELETE <deleteURI> HTTP/1.1
Host: ...
X-CIMI-Specification-Version: 1.0
```

The following provides additional constraints on the request message:

- **X-CIMI-Specification-Version**
  - This optional HTTP header specifies the list of versions of this specification that the server shall choose from to process this message.

The response will be of the following form:

```plaintext
HTTP/1.1 200 OK
X-CIMI-Specification-Version: 1.0
```

The following provides additional constraints on the response message:

- **X-CIMI-Specification-Version**
  - This REQUIRED HTTP header specifies the version of this specification that was used to process this message.

The HTTP response will also include a status code, as described in the following table:

<table>
<thead>
<tr>
<th>HTTP Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202 Accepted</td>
<td>The resource is in the process of being created. Investigate Job to determine the current status of the operation.</td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>Invalid parameter or field names in the request.</td>
</tr>
<tr>
<td>401 Unauthenticated</td>
<td>Incorrect or missing authentication credentials.</td>
</tr>
<tr>
<td>403 Unauthorized</td>
<td>Client lacks the proper authorization to perform this request.</td>
</tr>
</tbody>
</table>

### 4.2.2.5 Other Operations

While some modifications to the resources in the model can be done via a simple update (PUT) operation to the resource's "editURI", sometimes a more complex set of actions need to be taken. In these cases, the operations will be modeled as HTTP POSTs to the operation specific "Link" property/URI of the resource.
For each of the resources that define additional operations, a description of the HTTP request and response bodies will be provided. However, the general HTTP interaction will be as described below.

The request will be of the following form:

```
POST <operationLinkURI> HTTP/1.1
Host: ...
Accept: application/CIMI-...
Content-Type: application/CIMI-...
X-CIMI-Specification-Version: 1.0

<serialization of request to perform some action>
```

The following provides additional constraints on the request message:

```
X-CIMI-Specification-Version
```

This optional HTTP header specifies the list of versions of this specification that the server shall choose from to process this message.

The form of the response will vary depending on the operation and will be defined by the operation itself.

Note that the definition of the "Create" operation (see section 4.2.2.1) follows this same pattern - it is just called out for ease of reference.

### 4.2.2.6 Synchronous Operations

If a Provider supports the Job entity then each incoming PUT, DELETE, POST request SHALL result in a Job entity being created and a reference to that Job entity SHALL be returned back to the client via the X-CIMI-Job-URI HTTP Header in the HTTP response message:

```
X-CIMI-Job-URI: <uri-to-Job>
```

In this case, the requested operation shall be complete and the Job URI SHALL point to a completed Job.

If the Job is not complete, then the server SHALL return a 202 and follow the instructions for Asynchronous Operations.

### 4.2.2.7 Asynchronous Operations

In some cases, an operation requested by the client may take an undetermined amount of time to complete. For example, creating a new Machine or starting an existing Machine, may take a relatively long time to complete. In these cases, it is not practical to complete these operations within a reasonable HTTP request timeout interval. For these cases, the Provider SHALL return an HTTP "202 Accepted" response code.

As with synchronous operations, if a Provider supports the Job entity then it SHALL create a Job entity for the incoming request and return a reference to that Job entity back to the client via the X-CIMI-Job-URI HTTP Header in the HTTP response message. Additionally, in the case of a "202 Accepted" response code and a Job URI being returned, the Provider MAY also return a representation of the Job entity in the body of the HTTP response message. If the request did not include the Job MIME type in the HTTP Accept header, then the encoding style (json vs xml) of the response SHOULD match the encoding style of the request message.

Note that the decision as to whether any particular operation will be synchronous or asynchronous is at the server's discretion.

### 5 Model

This model assumes that a business relationship has already been established between the Cloud Consumer and the Cloud Provider. This relationship may include financial terms, creating separately
administered clouds that the consuming organization is paying for, and the establishment of
authentication credentials to access the administrative entry point for each cloud. This scope of this
model is one separately administered cloud.

5.1 Identifiers

All identifiers (e.g. entity names, attributes, operations, parameter names) defined by this specification, or
defined via an extension, shall adhere to the following:

- Identifier names shall be treated as case sensitive
- Identifier names shall only use the following set of characters:
  - Upper case ASCII (U+0041 through U+005A)
  - Lower case ASCII (U+061 through U+007A)
  - Digits (U+0030 through U+0039)
  - Underscore (U+005F)

5.2 Attribute "Properties"

Each attribute of the entities in the CIMI model is augmented by a set of "Properties" that further qualify
the attribute being defined. The following describes the possible "Properties".

Optionality:

The entity definition tables contain an indicator as to whether the specified attribute (and its
corresponding feature) is required to be supported by Cloud providers. Possible values are:

- Optional - indicates that the specified attribute/feature may be supported
- Mandatory - indicates that the specified attribute/feature shall be supported
- Conditional - indicates the specified attribute/feature is mandatory if the condition is satisfied. The
  condition will be described in the description cell. If the condition is not met, the attribute is
  optional.

Mutability:

Attributes are either “Immutable” (their values are fixed for the lifetime of the entity), or “Mutable” (their
values may change over the lifetime of the entity). Unless otherwise noted, all attributes are mutable.

Writability:

Mutable attributes are either “Writeable” (their value may be changed by the Consumer) or “Read-Only”
(their value may only be changed by the Provider). Unless otherwise noted, all mutable attributes are
writeable.

5.3 Data Types and Values

The following describes the data types and values that are using within the model definition tables.

URIs:

Note that the format and syntax of the attributes of type “URI” is defined by RFC 3986 [RFC3986] with the
following, additional constraints: Relative URIs MUST start with a “/”, otherwise the URI is assumed to be
absolute and no URI processing (to determine the full path) will be performed. Relative URIs are interpreted as being relative to the root URI of the CloudEntryPoint.

Units:

Some of the entities defined by this specification have attributes that describe an amount of something that belongs to, or is associated with that entity. For example, the Machine entity has a memory attribute which describes “the size of the memory allocated to this machine”. This specification adopts the convention of representing such attributes via a duple consisting of a quantity (represented as an integer) and units (represented as a string). The allowable values for units are listed in the following table. Their meaning is defined in IEC 80000-13:2008 [IEC 80000-13:2008]. Their numerical equivalents are provided here for convenience:

<table>
<thead>
<tr>
<th>String</th>
<th>Numerical Value</th>
<th>String</th>
<th>Numerical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>kilobyte</td>
<td>$10^3$</td>
<td>kibibyte</td>
<td>$2^{10}$</td>
</tr>
<tr>
<td>megabyte</td>
<td>$10^6$</td>
<td>mebibyte</td>
<td>$2^{20}$</td>
</tr>
<tr>
<td>gigabyte</td>
<td>$10^9$</td>
<td>gibibyte</td>
<td>$2^{30}$</td>
</tr>
<tr>
<td>terabyte</td>
<td>$10^{12}$</td>
<td>tebibyte</td>
<td>$2^{40}$</td>
</tr>
<tr>
<td>petabyte</td>
<td>$10^{15}$</td>
<td>pebibyte</td>
<td>$2^{50}$</td>
</tr>
<tr>
<td>exabyte</td>
<td>$10^{18}$</td>
<td>exbibyte</td>
<td>$2^{60}$</td>
</tr>
<tr>
<td>zettabyte</td>
<td>$10^{21}$</td>
<td>zebibyte</td>
<td>$2^{70}$</td>
</tr>
<tr>
<td>yottabyte</td>
<td>$10^{24}$</td>
<td>yobibyte</td>
<td>$2^{80}$</td>
</tr>
</tbody>
</table>

5.4 Relationship Semantics

A reference between two entity instances has the semantics of a simple “association”. In particular, unless specified otherwise, (a) the same referred instance can be referred by other entity instances, i.e. be “shared”, and (b) the referred entity instance is not affected when deleting the referring entity instance (i.e. the Delete operation is a “shallow delete” by default).

The embedding of a sub-entity inside another entity, has the semantics of a “composition” (or whole-part relationship in UML). In particular, unless specified otherwise: (a) an embedded sub-entity cannot be shared by several entity instances, (b) when deleting an embedding entity instance, the embedded sub-entity instances are also deleted.

5.5 Alternative Model Formats

Since it is expected that this specification will be implemented using a variety of technologies, as a convenience, the definition of the model elements are provided in alternative formats that are easily consumable by technology-specific tooling.

This model is available in a CIM/MOF format [CIMI-CIM] as well as a RelaxNG format [CIMI-RNG].

*Note: the CIMI-RNG document is not yet available.*

In the event of inconsistencies between the various formats, the normative text within this specification takes precedence over the XML Schemas and alternative formats, which in turn take precedence over examples.
5.6 Entities

The following sections detail the attributes of the entities defined by the CIMI model.

5.6.1 Common Attributes

The entities described by this document share the following, common attributes.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>ref</td>
<td>The unique self-reference to this entity; assigned upon entity creation. This attribute value shall be unique in the provider’s cloud.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The human readable name of this entity; assigned by the creator as a part of the entity creation input.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The human readable description of this entity; assigned by the creator as a part of the entity creation input.</td>
</tr>
<tr>
<td>created</td>
<td>DateTimeUTC</td>
<td>The timestamp when this entity was created. The format should be unambiguous, and the value is immutable.</td>
</tr>
<tr>
<td>properties</td>
<td>map</td>
<td>A list of name/value pairs, some of which may control one or more aspects this entity. Properties may also serve as an extension point, allowing consumers and providers to record configuration and control information for features and capabilities beyond those defined by this specification.</td>
</tr>
</tbody>
</table>

The same "name" SHALL NOT be used more than once within a "properties" attribute.

Individual properties may be either Mutable or Immutable and, if mutable, Writeable or Read-Only, depending upon the nature of the property and the underlying cloud implementation.

Properties: Optional / Mutable

5.7 Entity Metadata

Implementations of this specification SHOULD allow for Consumers to discover the metadata associated with each supported entity. Doing so allows for the discovery of Provider defined constraints on the CIMI defined attributes as well as discovery of any new extension attributes that the Provider may have defined. The mechanism by which this metadata is made available will be protocol specific.

Each entity's metadata will contain the following pieces of information:
Name: EntityMetadata

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>self</td>
<td>ref</td>
<td>The unique self-reference to this entity; assigned upon entity creation. This attribute value is <strong>immutable</strong>, and shall be <strong>unique</strong> in the provider’s cloud.</td>
</tr>
<tr>
<td>typeURI</td>
<td>URI</td>
<td>A unique URI associated with, and denoting, this entity type.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the entity type.</td>
</tr>
<tr>
<td>attributes</td>
<td>attribute[]</td>
<td>A set of Provider defined metadata that can be used by clients to discover any metadata associated with each attribute, as well as the set of extension attributes. Each attribute will contain the following nested data:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Type</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
</tr>
<tr>
<td>namespace</td>
<td>URI</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
</tr>
<tr>
<td>required</td>
<td>boolean</td>
</tr>
<tr>
<td>constraints</td>
<td>abstract</td>
</tr>
<tr>
<td>-------------</td>
<td>----------</td>
</tr>
</tbody>
</table>

**Properties:** Optional / Mutable

<table>
<thead>
<tr>
<th>operations</th>
<th>operation[]</th>
<th>A set of Provider defined operations that can be used by clients to act on the entity.</th>
</tr>
</thead>
</table>

Each operation will contain the following nested data:

<table>
<thead>
<tr>
<th>Name</th>
<th>operation</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Data</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>name</th>
<th>string</th>
<th>The name of the operation.</th>
</tr>
</thead>
</table>

**Properties:** Mandatory / Mutable

<table>
<thead>
<tr>
<th>uri</th>
<th>URI</th>
<th>A URI that uniquely identifies the operation at a global level.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>description</th>
<th>string</th>
<th>The human readable description of the semantic of the operation.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>method</th>
<th>string</th>
<th>The method to use to perform the operation.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>inputMessage</th>
<th>string</th>
<th>The body mimeType of the request message, it may depend on the model format choosen by the provider.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>outputMessage</th>
<th>string</th>
<th>The body mimeType of the response message, it may depend on the model format choosen by the provider.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

**Properties:** Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-EntityMetadata+json

**JSON serialization:**

```json
{ "self": string,
  "typeURI": URI,
  "name": string,
  "attributes": [ ]
}
```
{ "name": "string", 
  "namespace": "string", ?, 
  "type": "string", ?, 
  "required": boolean, ?
  ...constraints...? } *
}, ?
"operations": [ 
  { "name": "string" 
    "description": "string", ? 
    "method": "string",
    "inputMessage": "string", ? 
    "outputMessage": "string" ? }, *
  ] ?
...}

**XML media type:** application/CIMI-EntityMetadata+xml

**XML serialization:**

```xml
<EntityMetadata xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name>
  <typeURI> xs:anyURI </typeURI>
  <attribute name="xs:string" namespace="xs:anyURI"? type="xs:string"
    required="xs:boolean"? >
    ...constraints...?
  </attribute> *
  <operation name="xs:string" description="xs:string"? method="xs:string"
    inputMessage="xs:string"? outputMessage="xs:string"? /> *
  <xs:any>*
</EntityMetadata>
```

Additional metadata about the entity or attributes MAY be included by the Provider.

### 5.7.1 Attribute Types

The following describes the constraint metadata corresponding to the attribute's "type" value.

**type="string"**

The JSON SHALL be of the form:

```
"values": [ "string", + ] ?
```

The XML SHALL be of the form:

```xml
<value> xs:string </value> *
```

**type="integer"**

The JSON SHALL be of the form:

```
"values": [ integer, + ], ?
"ranges": [ [ "low": integer, "high": integer ], + ] ?
```

The XML SHALL be of the form:

```xml
<value> xs:integer </value> *
<range low="xs:integer" high="xs:integer"/> *
```

The total value space of an 'integer' attribute is the accumulation of all values and ranges.
The JSON SHALL be of the form:

```
"value": boolean ?
```

The XML SHALL be of the form:

```
<value> xs:boolean </value> ?
```

Only one ‘value’ is permitted which indicates whether the attribute is required to be either ‘true’ or ‘false’.

### 5.7.2 Examples

The following shows a sample metadata document for a VolumeConfiguration entity in XML that as been extended with a "Location" string attribute:

```
<EntityMetadata xmlns="http://www.dmtf.org/cimi">
  <self> http://example.org/types/VC </self>
  <typeURI> http://www.dmtf.org/cimi/VolumeConfiguration </typeURI>
  <name> VolumeConfiguration </name>
  <attribute name="Location" namespace="http://example.org/" type="string"/>
</EntityMetadata>
```

The following shows the same VolumeConfiguration but the "Location" attribute is restricted to a set of values and is required:

```
<EntityMetadata xmlns="http://www.dmtf.org/cimi">
  <self> http://example.org/types/VC </self>
  <typeURI> http://www.dmtf.org/cimi/VolumeConfiguration </typeURI>
  <name> VolumeConfiguration </name>
  <attribute name="Location" namespace="http://example.org/" type="string" required="true">
    <value> NYC </value>
    <value> LAX </value>
  </attribute>
</EntityMetadata>
```

The following shows the same VolumeConfiguration serialized in JSON:

```
{
  "self": "http://example.org/types/VC",
  "typeURI": "http://www.dmtf.org/cimi/VolumeConfiguration",
  "name": "VolumeConfiguration",
  "attributes": [
    {
      "name": "Location",
      "namespace": "http://example.org",
      "type": "string",
      "required": true,
      "values": [ "NYC", "LAX" ]
    }
  ]
}
```

The following shows a Volume serialized in JSON which provides an operation of data compression. In this specific example the method returned (POST) is for the CIMI REST protocol; should another protocol be implemented (e.g SOAP) the "method" will be different:

```
{
  "uri": "http://example.org/types/V",
  "typeURI": "http://www.dmtf.org/cimi/Volume",
  "name": "Volume",
  "operations": [
    {
      "name": "compress",
```
5.8 Cloud Entry Point

The Cloud Entry Point represents the entry point into the cloud defined by the CIMI Model. The Cloud Entry Point implements a catalog of entities such as Systems, System Templates, Machines, Machine Templates, etc. that can be queried and browsed by the Consumer.

<table>
<thead>
<tr>
<th>Name</th>
<th>CloudEntryPoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmf.org/cimi/CloudEntryPoint">http://www.dmf.org/cimi/CloudEntryPoint</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemTemplates</td>
<td>ref</td>
<td>A reference to the System Template Collection of this CloudEntry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>systems</td>
<td>ref</td>
<td>A reference to the System Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>machineTemplates</td>
<td>ref</td>
<td>A reference to the Machine Template Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>machineConfigs</td>
<td>ref</td>
<td>A reference to the Machine Configuration Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>machineImages</td>
<td>ref</td>
<td>A reference to the Machine Image Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>machineAdmins</td>
<td>ref</td>
<td>A reference to the Machine Admin Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>machines</td>
<td>ref</td>
<td>A reference to the Machine Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>volumeTemplates</td>
<td>ref</td>
<td>A reference to the Volume Template Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>volumeConfigs</td>
<td>ref</td>
<td>A reference to the Volume Configuration Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>volumImages</td>
<td>ref</td>
<td>A reference to the Volume Image Collection of this Cloud Entry Point.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>
| volumes | ref | A reference to the Volume Collection of this Cloud Entry Point.  
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Properties:</strong> Optional / Mutable</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| networkTemplates | ref | A reference to the Network Template Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| networkConfigs | ref | A reference to the Network Configuration Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| networks | ref | A reference to the Network Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| vspTemplates | ref | A reference to the VSP Template Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| vspConfigs | ref | A reference to the VSP Configuration Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| vsps | ref | A reference to the VSP Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| routingGroups | ref | A reference to the Routing Group Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| meterTemplates | ref | A reference to the Meter Template Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| meters | ref | A reference to the Meter Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| eventLogs | ref | A reference to the Event Log Collection of this Cloud Entry Point.  
| **Properties:** Optional / Mutable |
| events | ref | A reference to the Event Collection of this Cloud Entry Point  
| **Properties:** Optional / Mutable |
| jobTime | long | This value is Provider specific and is the minimum amount of time a Job will be retained by the system after the completion of the Job.  
| **Properties:** Optional / Mutable |
| entityMetadata | ref[] | List of references to EntityMetadata entities supported by the Provider. If an entity does not have any metadata then it will not appear in this list - e.g. it has no constraints beyond what the CIMI specification defines nor does it have any extension attributes.  
| **Properties:** Optional / Mutable |
The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-CloudEntryPoint+json

**JSON serialization**:

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "systemTemplates": { "href": string }, ?,
  "systems": { "href": string }, ?,
  "machineTemplates": { "href": string }, ?,
  "machineConfigs": { "href": string }, ?,
  "machineImages": { "href": string }, ?,
  "machineAdmins": { "href": string }, ?,
  "machines": { "href": string }, ?,
  "volumeTemplates": { "href": string }, ?,
  "volumeConfigs": { "href": string }, ?,
  "volumeImages": { "href": string }, ?,
  "volumes": { "href": string }, ?,
  "networkTemplates": { "href": string }, ?,
  "networkConfigs": { "href": string }, ?,
  "networks": { "href": string }, ?,
  "vspTemplates": { "href": string }, ?,
  "vspConfigs": { "href": string }, ?,
  "vspas": { "href": string }, ?,
  "routingGroups": { "href": string }, ?,
  "meterTemplates": { "href": string }, ?,
  "meters": { "href": string }, ?,
  "eventLogs": { "href": string }, ?,
  "events": { "href": string }, ?,
  "job_time": number, ?
  "entityMetadata": [?
    { "href": string }, +
  ], ?,
  "operations": [?
    { "rel": "edit", "href": string }, ?,
  ]?
... }
```

**XML media type**: application/CIMI-CloudEntryPoint+xml

**XML serialization**:

```xml
<CloudEntryPoint xmlns="http://www.dmtf.org/cimi" >
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:dateTime </created>
  <property name="xs:string"> xs:string </property> *
  <systemTemplates href="xs:anyURI"/> ?
  <systems href="xs:anyURI"/> ?
  <machineTemplates href="xs:anyURI"/> ?
  <machineConfigs href="xs:anyURI"/> ?
  <machineImages href="xs:anyURI"/> ?
  <machineAdmins href="xs:anyURI"/> ?
  <machines href="xs:anyURI"/> ?
  <volumeTemplates href="xs:anyURI"/> ?
  <volumeConfigs href="xs:anyURI"/> ?
  <volumeImages href="xs:anyURI"/> ?
</CloudEntryPoint>
```
5.8.1 Operations
This entity supports the Read and Update operations.

5.9 System Entities and Relationships
The following diagram illustrates the entities involved in constructing a System and their relationships. Although this drawing is in the style of an Entity Relationship diagram, the use of UML is neither rigorous nor normative.

Figure 1 - System Entities
5.9.1 System Template

The System Template contains configuration values for realizing a System. A System Template can be used to create multiple Systems.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>SystemTemplate</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/SystemTemplate">http://www.dmtf.org/cimi/SystemTemplate</a></td>
<td></td>
</tr>
<tr>
<td>volumeTemplates</td>
<td>ref[]</td>
<td>List of volume templates referenced in this System Template. Items in this list are used to create Volumes as part of the System creation process. Properties: Optional / Mutable</td>
</tr>
<tr>
<td>machineTemplates</td>
<td>ref[]</td>
<td>List of Machine Templates referenced in this System Template. Items in this list are used to create Machines as part of the System creation process. Properties: Optional / Mutable</td>
</tr>
<tr>
<td>networkTemplates</td>
<td>ref[]</td>
<td>List of NetworkTemplates referenced in this System Template. Items in this list are used to create Networks as part of the System creation process. Properties: Optional / Mutable</td>
</tr>
</tbody>
</table>

**JSON media type:** application/CIMI-SystemTemplate+json

**JSON serialization:**

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "volumeTemplates": [  
    { "href": string }, +
  ], ?,
  "machineTemplates": [  
    { "href": string }, +
  ], ?,
  "networkTemplates": [  
    { "href": string }, +
  ], ?,
  "operations": [  
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?
  ] ?
  ...
}
```

**XML media type:** application/CIMI-SystemTemplate+xml

**XML serialization:**

```xml
<SystemTemplate xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <volumeTemplate href="xs:anyURI"/> *
</SystemTemplate>
```
5.9.1.1 Operations
This entity supports the Read, Update and Delete operations. Create is supported via the System Template Collection entity.

5.9.2 System Template Collection
A System Template Collection entity represents the collection of System Template entities within a Provider. This resource can be used to locate and create System Templates.

<table>
<thead>
<tr>
<th>Name</th>
<th>SystemTemplateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/SystemTemplateCollection">http://www.dmtf.org/cimi/SystemTemplateCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemTemplates</td>
<td>ref[]</td>
<td>An array of references to the set of System Templates in the Provider.</td>
</tr>
<tr>
<td>Properties:</td>
<td></td>
<td>Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-SystemTemplateCollection+json

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "systemTemplates": [
    { "href": string }, +
  ], ?
  "operations": [
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string } ?
  ] ?
...
}
```

**XML media type:** application/CIMI-SystemTemplateCollection+xml

```xml
<SystemTemplateCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <systemTemplate href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
</xs:any>*
```

5.9.2.1 Operations
This entity supports the Read and Update operations. Creation of new System Template entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.9.3 System
A System is a realized entity that consists of for example: one or more Machines, Volumes, and Networks (and others) that could be connected and associated with each other. A System can be operated and managed as a single entity and usually forms a stack of service. For example, a shopping cart system consists of machines for web servers and databases, network addresses for public access, and volumes for database files. A System may directly provide a user facing component, or may provide an infrastructure component.

<table>
<thead>
<tr>
<th>Name</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/System">http://www.dmtf.org/cimi/System</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>string</td>
<td>Current state of the System entity as last known. This is a label containing lifecycle state (e.g. INITIATED, CREATING, CREATED, DESTROYING, DESTROYED). This value is read-only and will change based on the state of the System. Properties: Mandatory / Mutable</td>
</tr>
<tr>
<td>machines</td>
<td>ref[]</td>
<td>The list of Machines contained in this System. Adding an item (of type Machine) to this list is logically equivalent to associating the Machine to this System with a &quot;containment relationship&quot;. Removing an item from this list is logically equivalent to de-associating the Machine from this System. Properties: Optional / Mutable</td>
</tr>
<tr>
<td>volumes</td>
<td>ref[]</td>
<td>The list of Volumes contained in this System. Adding an item (of type Volume) to this list is logically equivalent to associating the Volume to this System with a &quot;containment relationship&quot;. Removing an item from this list is logically equivalent to de-associating the Volume from this System. Properties: Optional / Mutable</td>
</tr>
<tr>
<td>networks</td>
<td>ref[]</td>
<td>The list of Networks contained in this System. Adding an item (of type Network) to this list is logically equivalent to associating the Network to this System with a &quot;containment relationship&quot;. Removing an item from this list is logically equivalent to de-associating the Network from this System. Properties: Optional / Mutable</td>
</tr>
<tr>
<td>meters</td>
<td>ref[]</td>
<td>A list of references to Meters monitored for this System. Properties: Optional / Mutable</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
<td>A reference to the EventLog of this System. Properties: Optional / Mutable</td>
</tr>
</tbody>
</table>
JSON media type: application/CIMI-System+json

JSON serialization:

```json
{  "self": string,
    "name": string, ?
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
    "state": string,
    "machines": [  
        { "href": string }, +
    ], ?
    "volumes": [  
        { "href": string }, +
    ], ?
    "networks": [  
        { "href": string }, +
    ], ?
    "meters": [  
        { "href": string }, +
    ], ?
    "eventLog": { "href": string }, ?
    "operations": [  
        { "rel": "edit", "href": string }, ?
        { "rel": "delete", "href": string } ?
    ] ?
    ...}
```

XML media type: application/CIMI-System+xml

XML serialization:

```xml
<System xmlns="http://www.dmtf.org/cimi">
    <self> xs:anyURI </self>
    <name> xs:string </name> ?
    <description> xs:string </description> ?
    <created> xs:string </created>
    <property name="xs:string"> xs:string </property> *
    <state> xs:string </state>
    <machine href="xs:anyURI"/> *
    <volume href="xs:anyURI"/> *
    <network href="xs:anyURI"/> *
    <meter href="xs:anyURI"/> *
    <eventLog href="xs:anyURI"/> ?
    <operation rel="edit" href="xs:anyURI"/> ?
    <operation rel="delete" href="xs:anyURI"/> ?
    <xs:any>*
</System>
```

5.9.3.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the System Collection entity.

5.9.4 System Collection

A System Collection entity represents the collection of System entities within a Provider. This entity can be used to locate and create Systems.

<table>
<thead>
<tr>
<th>Name</th>
<th>SystemCollection</th>
</tr>
</thead>
</table>

5.9.5.1 Properties

This entity supports the Read, Update and Delete operations. Create is supported via the System Collection entity.
The following describes the serialization of the entity in both JSON and XML:

### JSON media type: application/CIMI-SystemCollection+json

```json
   { "self": string, 
     "name": string, ?
     "description": string, ?
     "created": string, ?
     "properties": { "name": string, + }, ?
     "systems": [ 
       { "href": string }, +
     ], ?
     "operations": [ 
       { "rel": "add", "href": string }, ?
     } ?
   }
```

### XML media type: application/CIMI-SystemCollection+xml

```xml
   <SystemCollection xmlns="http://www.dmtf.org/cimi">
     <self> xs:anyURI </self>
     <name> xs:string </name> ?
     <description> xs:string </description> ?
     <created> xs:string </created>
     <property name="xs:string"> xs:string </property> *
     <system href="xs:anyURI"/> *
     <operation rel="add" href="xs:anyURI"/> ?
     <operation rel="edit" href="xs:anyURI"/> ?
     <xs:any>*
   </SystemCollection>
```

### 5.9.4.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

#### Creating a New System

```
   /link@rel: add
```

This operation will create a new System.

Input parameters: Either a reference to a System Template or a System Template itself.

Output parameters: A reference to a new System and optionally the representation of the System.
To create a new System a POST is sent to the "add" URI of the SystemCollection where the HTTP request body SHALL be as described below. Note this structure allows for certain properties to be passed in "by value" or by "reference". The definition of each property can be found in section 5.9.1.

**JSON media type:** application/CIMI-SystemCreate+json

**JSON serialization:**

```json
{
   "name": string,
   "description": string, ?
   "properties": { "name": string, + }, ?
   "systemTemplate": { "href": string, ?
   "properties": { "name": string, + }, ?
   "volumeTemplates": [
      { "href": string, ?
      "name": string, ?
      "description": string, ?
      "properties": { "name": string, + }, ?
      "volumeConfig": { "href": string, ?
      "properties": { "name": string, + }, ?
      "format": string, ?
      "capacity": { "quantity": number, "units": string }, ?
      "supportsSnapshots": boolean, ?
      "guestInterface": string, ?
      },
      "volumeImage": { "href": string, 
      "imageLocation": { "href": string }, ?
      "imageData": string, ?
      "bootable": boolean ?
      }, +
      ]
   }, +
   "machineTemplates": [
      { "href": string, ?
      "name": string, ?
      "description": string, ?
      "properties": { "name": string, + }, ?
      "machineConfig": { "href": string, ?
      "cpu": string, ?
      "memory": { "quantity": integer, "units": string }, ?
      "disks": [
      { "capacity": { "quantity": integer, "units": string },
      "guestInterface": string }, +
      ]
      },
      "machineImage": { "href": string, 
      "imageLocation": { "href": string }, ?
      "imageData": string, ?
      }, +
      "machineAdmin": { "href": string, ?
      <provider specific data> ?
      }, ?
      "volumes": [
      { "href": string, "attachmentPoint": string, "protocol": string}, +
      ], ?
      "volumeTemplates": [
      { "href": string, ?
      "name": string, ?
      "description": string, ?
      "properties": { "name": string, + }, ?
      "attachmentPoint": string, "protocol": string,
      "volumeConfig": { "href": string, ?
      "properties": { "name": string, + }, ?
      
```
```
"format": string,?
"capacity": { "quantity": number, "units": string },?
"supportsSnapshots": boolean,?
"guestInterface": string,?
"volumeImage": { "href": string,?
  "properties": { "name": string, + },?
  "imageLocation": { "href": string },?
  "imageData": string,?
  "bootable": boolean ?
} ?,

"networkTemplates": [?
  { "href": string,?
    "name": string,?
    "description": string,?
    "properties": { "name": string, + },?
    "networkConfig": { "href": string },
    "routingGroup": { "href": string }?
  }
],?

"machineTemplates": [?
  { "href": string,?
    "name": string,?
    "guestInterface": string,?
    "volumeImages": [?
      { "href": string,?
        "properties": { "name": string, + },?
        "imageLocation": { "href": string },?
        "imageData": string,?
        "bootable": boolean ?
      },
      +
    ],?
  ],?

XML media type: application/CIMI-SystemCreate+xml

XML serialization

```xml
<SystemCreate>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <property name="xs:string"> xs:string </property> *
  <systemTemplate href="xs:anyURI"?>
    <property name="xs:string"> xs:string </property> *
    <volumeTemplate href="xs:anyURI"?>
      <name> xs:string </name>
      <description> xs:string </description> ?
      <property name="xs:string"> xs:string </property> *
      <volumeConfig href="xs:anyURI"?>
        <property name="xs:string"> xs:string </property> *
        <format> xs:string </format> ?
        <capacity quantity="xs:integer" units="xs:string"/> ?
        <supportsSnapshots> xs:boolean </supportsSnapshots> ?
        <guestInterface> xs:string </guestInterface> ?
      </volumeConfig>
      <volumeImage href="xs:anyURI">
        <property name="xs:string"> xs:string </property> *
        <imageLocation href="xs:anyURI"/> ?
        <imageData> xs:any* </imageData> ?
        <bootable> xs:boolean </bootable> ?
      </volumeImage> ?
    </volumeTemplate> *
  </systemTemplate>
...```

"networkTemplates": [
  { "href": string,?
    "name": string,?
    "description": string,?
    "properties": { "name": string, + },?
    "networkConfig": { "href": string },
    "routingGroup": { "href": string }?
  }
],?

"machineTemplates": [?
  { "href": string,?
    "name": string,?
    "guestInterface": string,?
    "volumeImages": [?
      { "href": string,?
        "properties": { "name": string, + },?
        "imageLocation": { "href": string },?
        "imageData": string,?
        "bootable": boolean ?
      },
      +
    ],?
  ],?

```
The serialization of some reference properties are specified such that a request MAY either include a reference ("href") to an existing entity or to include the entity “inline” as a set of additional properties. Requests SHALL NOT include both a reference and the inlined set of properties.

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the System entity.
### 5.10 Machine Entities and Relationships

The following diagram illustrates the entities involved in constructing a Machine and their relationships.

Although this drawing is in the style of an Entity Relationship diagram, the use of UML is neither rigorous nor normative.

![Machine Entities and Relationships Diagram]

**Figure 2 - Machine Entities**

### 5.10.1 Machine Template

A Machine Template represents the set of metadata and instructions used in the creation of a Machine.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MachineTemplate</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineTemplate">http://www.dmtf.org/cimi/MachineTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machineConfig</td>
<td>ref</td>
<td>A reference to the Machine Configuration that will be used to create a Machine from this Machine Template.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>machineImage</td>
<td>ref</td>
<td>A reference to the Machine Image that will be used to create a Machine from this Machine Template.</td>
</tr>
<tr>
<td>machineAdmin</td>
<td>ref</td>
<td>A reference to the Machine Admin that will be used to create the initial login credential for the new Machine.</td>
</tr>
<tr>
<td>volumes</td>
<td>volume[]</td>
<td>A list of references to existing Volumes that will be attached to the Machine during its creation. Each volume attribute has the following sub-attributes which describe aspects of the way in which the Machine will be attached to the Volume:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>volumeTemplates</td>
<td>volumeTemplate[]</td>
<td>A list of references to Volume Templates that will be used to create a set of new Volumes that will to be attached to the Machine during its creation. If the Machine is created as part of a System creation, the Volumes created from these templates will be considered as part of that System without the need for these Volume Templates to also be listed in the volumeTemplates attribute of the relevant System Template. If the same Volume Template reference is listed in both the volumeTemplates attribute of a System Template and in the volumeTemplates attribute of a Machine Template contained by that System Template, this means that multiple, distinct Volume instances will be created as part of the overall System creation. Each volumeTemplate attribute has the following sub-attributes which describe aspects of the way in which the Machine will be attached to the Volume instance that will be created from the template:</td>
</tr>
<tr>
<td>Name</td>
<td>volumeTemplate</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>attachmentPoint</td>
<td>String</td>
<td>File system path where the Volume will be attached.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>protocol</td>
<td>String</td>
<td>Protocol that will be used to access this Volume (e.g. NFS, iSCSI).</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>volumeTemplate</td>
<td>ref</td>
<td>Reference to the Volume Template that will be used to create a new Volume.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>networkInterfaces</td>
<td>networkInterface[]</td>
<td>A list of sub-entities that define the network interfaces that will be created on Machines instantiated from this template.</td>
</tr>
<tr>
<td>Name</td>
<td>networkInterface</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>vsp</td>
<td>ref</td>
<td>A reference to the VSP (Virtual Switch Port) for this network interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note this is a reference to a VSP and not a VSPTemplate. It is expected that VSPs and Networks will be defined separately and prior to the Machines that connect to them.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>hostname</td>
<td>string</td>
<td>DNS resolvable name associated with this network interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>While this attribute can be specified in most cases it is expected to be supplied by the Provider. Specifying this value is typically only done when the Template is only used for one particular Machine.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>macAddress</td>
<td>string</td>
<td>Address assigned by the hypervisor when a machine is</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
<td>The state of an interface configurable to be &quot;Active&quot; or &quot;Standby&quot;.</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>Selected network protocol such as - IPv4 or IPv6.</td>
</tr>
<tr>
<td>allocation</td>
<td>string</td>
<td>The option for &quot;Dynamic Host Allocation Protocol&quot; or static.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>The IP address assigned to a virtual interface.</td>
</tr>
<tr>
<td>defaultGateway</td>
<td>string</td>
<td>An IP address to a firewall or router that serves other networks.</td>
</tr>
<tr>
<td>dns</td>
<td>string</td>
<td>The IP address of the Domain Name Service from host name to IP resolution.</td>
</tr>
<tr>
<td>maxTransmissionUnit</td>
<td>integer</td>
<td>To set the largest supported value.</td>
</tr>
</tbody>
</table>

While this attribute can be specified in most cases it is expected to be supplied by the Provider. Specifying this value is typically only done when the Template is only used for one particular Machine.

**Properties:**

- state: Optional / Mutable
- protocol: Mandatory / Mutable
- allocation: Mandatory / Mutable
- address: Optional / Mutable
- defaultGateway: Optional / Mutable
- dns: Optional / Mutable
- maxTransmissionUnit: Optional / Mutable
The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-MachineTemplate+json

**JSON serialization:**

```json
{
  "self": string,
  "name": string,
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "machineConfig": { "href": string },
  "machineImage": { "href": string },
  "machineAdmin": { "href": string }, ?
  "volumes": [
    { "href": string, "attachmentPoint": string, "protocol": string }, +
  ], ?,
  "volumeTemplates": [
    { "href": string, "attachmentPoint": string, "protocol": string }, +
  ], ?,
  "networkInterfaces": [
    { "vsp": {"href": string}, "hostname": string, "macAddress": string,
      "state": string, "protocol": string, "allocation": string,
      "address": string, "defaultGateway": string, "dns": string,
      "maxTransmissionUnit": integer }, +
  ], ?,
  "operations": [
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?
  ] ...
}
```

**XML media type:** application/CIMI-MachineTemplate+xml

**XML serialization:**

```xml
<MachineTemplate xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <machineConfig href="xs:anyURI"/>
  <machineImage href="xs:anyURI"/>
  <machineAdmin href="xs:anyURI"/>?
  <volume href="xs:anyURI">
    attachmentPoint="xs:string" protocol="xs:string" /> *
  <volumeTemplate href="xs:anyURI"
    attachmentPoint="xs:string" protocol="xs:string" /> *
  <networkInterface>
    <vsp href="xs:anyURI"/>
    <hostname> xs:string </hostname>
    <macAddress> xs:string </macAddress>
    <state> xs:string </state>
    <protocol> xs:string </protocol>
    <allocation> xs:string </allocation>
```
5.10.1.1 Operations
This entity supports the Read, Update and Delete operations. Create is supported via the Machine Template Collection entity.

5.10.2 Machine Template Collection
A Machine Template Collection entity represents the collection of Machine Template entities within a Provider. This entity can be used to locate and create Machine Templates.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineTemplateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineTemplateCollection">http://www.dmtf.org/cimi/MachineTemplateCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machineTemplates</td>
<td>ref[]</td>
<td>An array of references to the set of Machine Templates in the Provider. Properties: Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-MachineTemplateCollection+json

**JSON serialization**:

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "machineTemplates": [
    { "href": string }, +
  ], ?
  "operations": [
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string } ?
  ] ?
  ...}
```

**XML media type**: application/CIMI-MachineTemplateCollection+xml

**XML serialization**:

```xml
<MachineTemplateCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <machineTemplate href="xs:anyURI"/>
```
5.10.2.1 Operations

This entity supports the Read and Update operations. Creation of new Machine Template entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.10.3 Machine Configuration

The Machine Configuration entity represents the set of configuration values that define the (virtual) hardware resources of a to-be-realized Machine Instance. Machine Configurations are created by Providers and MAY, at the Providers discretion, be created by Consumers.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpu</td>
<td>Indicates the amount of CPU (based on standard CPU measurement) that a Machine realized from this configuration would have, by default.</td>
</tr>
<tr>
<td>memory</td>
<td>Indicates the amount of RAM that a Machine realized from this configuration will have. This attribute has the following sub-attributes which serve to describe it:</td>
</tr>
<tr>
<td>disks</td>
<td>Contains the list of metadata of the disks that will be created upon the instantiation of a Machine from this configuration. The disks are local storages to the Machine. Each disks attribute has the following sub-attributes:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capacity</td>
<td>Indicates the initial capacity of the disk described by this attribute. This property has the following, sub-attributes.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td>quantity</td>
<td>integer</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>units</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>format</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>attachmentPoint</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: The disk attributes "format" and "attachmentPoint" will not appear on Machine entities because once the Machine is created the user of the Machine will be able to modify those attributes of a disk without the Provider's knowledge - therefore it is no longer an aspect of the Machine that the Provider can reliably manage.

**JSON media type:** application/CIMI-MachineConfiguration+json

**JSON serialization:**

```json
{
    "self": string,
    "name": string,
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?,
    "cpu": string,
    "memory": { "quantity": integer, "units": string },
    "disks" : [
        { "capacity": { "quantity": integer, "units": string }, +
        ], ?,
    "operations": [
        { "rel": "edit", "href": string }, ?,
        { "rel": "delete", "href": string } ?
        ] ?
    } ...
}
```

**XML media type:** application/CIMI-MachineConfiguration+xml

**XML serialization:**

```xml
<MachineConfiguration xmlns="http://www.dmtf.org/cimi">
  </self> xs:anyURI </self>
```
Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP

Version 0.0.46a
DMTF — Work In Progress - not a DMTF Standard

5.10.3.1 Operations
This entity supports the Read, Update and Delete operations. Create is supported via the Machine Configuration Collection entity.

5.10.4 Machine Configuration Collection
A Machine Configuration Collection entity represents the collection of Machine Configuration entities within a Provider. This entity can be used to locate and create Machine Configurations.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineConfigurationCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineConfigurationCollection">http://www.dmtf.org/cimi/MachineConfigurationCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machineConfigurations</td>
<td>ref[]</td>
<td>An array of references to the set of Machine Configurations in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-MachineConfigurationCollection+json

```json
{
  "self": string, 
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "machineConfigurations": [
    { "href": string }, +
  ], ?,
  "operations": [
    { "rel": "add", "href": string }, ?,
    { "rel": "edit", "href": string } ?,
  ] ?,
  ...
}
```

**XML media type:** application/CIMI-MachineConfigurationCollection+xml

```xml
<MachineConfigurationCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
</MachineConfigurationCollection>
```
<machineConfiguration href="xs:anyURI"/> *
<operation rel="add" href="xs:anyURI"/> ?
<operation rel="edit" href="xs:anyURI"/> ?
<x:any>*
</MachineConfigurationCollection>

5.10.4.1 Operations
This entity supports the Read and Update operations. Creation of new Machine Configuration entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.10.5 Machine Image
This entity represents the information (e.g. an Open Virtualization Format (OVF) package) necessary for hardware virtualized resources to create a Machine Instance; it contains configuration data such as startup instructions, including possible combinations of the following:

- the software image (i.e. a copy of an installed Machine) which is to be instantiated on the disk and other virtual resources
- installation software, which, when executed on the hardware (virtual) resources, builds the machine instance
- both a disk image and a set of software and parameters in order to install new components not included in original disk image

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineImage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineImage">http://www.dmtf.org/cimi/MachineImage</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageLocation</td>
<td>ref</td>
<td>A reference to the location of the binary data that makes up this image. Either this attribute or imageData SHALL be present; however both values SHALL NOT be present simultaneously.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Optional / Immutable</td>
</tr>
<tr>
<td>imageData</td>
<td>byte[]</td>
<td>The binary data that makes up this image. Either this attribute or imageLocation SHALL be present; however both values SHALL NOT be present simultaneously.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Optional / Immutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

JSON media type: application/CIMI-MachineImage+json

JSON serialization:

```
{
    "self": string,
    "name": string,
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
    "imageLocation": { "href": string }, ?
    "imageData": string, ?
```
"operations": [
    { "rel": "edit", "href": "string" },
    { "rel": "delete", "href": "string" }
],

XML media type: application/CIMI-MachineImage+xml

XML serialization:

```
<MachineImage xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name>
  <description> xs:string </description>
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <imageLocation href="xs:anyURI"/>
  <imageData> xs:string </imageData>
  <operation rel="edit" href="xs:anyURI"/>
  <operation rel="delete" href="xs:anyURI"/>
  <xs:any>*
</MachineImage>
```

5.10.5.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Machine Image Collection entity.

5.10.6 Machine Image Collection

A Machine Image Collection entity represents the collection of Machine Image entities within a Provider. This entity can be used to locate and create Machine Images.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineImageCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineImageCollection">http://www.dmtf.org/cimi/MachineImageCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machineImages</td>
<td>ref[]</td>
<td>An array of references to the set of Machine Images in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-MachineImageCollection+json

**JSON serialization:**

```
{
  "self": "string",
  "name": "string",
  "description": "string",
  "created": "string",
  "properties": {
    "name": "string",
    "machineImages": [
      { "hhref": "string" }
    ],
  },
  "operations": [
    { "rel": "add", "hhref": "string" },
    { "rel": "edit", "hhref": "string" }
  ]
}
```
5.10.6.1 Operations

This entity supports the Read and Update operations. Creation of new Machine Image entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.10.7 Machine

An instantiated compute resource that encapsulates both CPU and Memory.

<table>
<thead>
<tr>
<th>Name</th>
<th>Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/Machine">http://www.dmtf.org/cimi/Machine</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>string</td>
<td>Indicates the operational state of the Machine. Allowable values include: <strong>STARTED</strong>: The Machine is available and ready for use. Allowable actions when in this state are: <strong>stop</strong> and <strong>restart</strong>. <strong>STOPPED</strong>: This is the virtual equivalent of powering off a physical Machine. There is no saved CPU or memory state. Allowable actions when in this state are: <strong>start</strong> and <strong>restart</strong>. Providers may define additional values. This value is <strong>read-only</strong> and will change based on the state of the Machine. <strong>Properties</strong>: Mandatory / Mutable</td>
</tr>
<tr>
<td>cpu</td>
<td>TBD</td>
<td>The size of the CPU allocated to this Machine to be used. This should adhere to the standard unit of measurement. For example, a Machine with 4 unit worth of CPU would allow the processes in the Machine to use up to 4 units worth of CPU (and be charged thereof). When this value is increased, it implies that the Machine is allocated more CPU to use, and vice versa when the value is decreased.</td>
</tr>
</tbody>
</table>
**Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP**

**Properties:** Mandatory / Mutable

**memory**  
Optional / Read-Only  
Mandatory / Read-Write  
Structure  
The size of the memory (RAM) allocated to this Machine. When this value is increased, it implies that the Machine is allocated more RAM, and vice versa when the value is decreased. This attribute has the following sub-attributes which serve to describe it:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quantity</td>
<td>integer</td>
<td>A numerical quantity expressed as an integer. <strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>units</td>
<td>string</td>
<td>An enumerated value that expresses the unit of measurement used. Allowable values are byte, kibibyte, mebibyte, gibibyte, tebibyte, pebibyte, exbibyte, zebibyte, and yobibyte. <strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
</tbody>
</table>

**Properties:** Mandatory / Mutable

**disks**  
Optional / Read-Only  
Mandatory / Read-Write  
Array  
The list of disks (local storages) that are part of the Machine. Adding an element to this list creates a disk. Each disk attribute has the following sub-attributes which describe aspects of the disk:

<table>
<thead>
<tr>
<th>Name</th>
<th>disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>capacity</td>
<td>Indicates the initial capacity of the disk described by this attribute. This property has the following sub-attributes.</td>
</tr>
<tr>
<td>quantity</td>
<td>integer</td>
</tr>
<tr>
<td>units</td>
<td>string</td>
</tr>
</tbody>
</table>
### volumes

<table>
<thead>
<tr>
<th>volume[]</th>
</tr>
</thead>
<tbody>
<tr>
<td>The list of networked volumes that are attached to this Machine. Adding a Volume to this list means that the Machine has some access to the data on the Volume. Removing a Volume from this list means that the Machine no longer has access to the data on the Volume. Each volume attribute has the following sub-attributes which describe aspects of the way in which the Machine is attached to the Volume:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>See &quot;volume&quot; in &quot;5.10.1 Machine Template&quot; for the definition of this sub-entity.</td>
<td></td>
</tr>
</tbody>
</table>

### networkInterfaces

<table>
<thead>
<tr>
<th>networkInterface[]</th>
</tr>
</thead>
<tbody>
<tr>
<td>A list of sub-entities that define the network interfaces on this Machine.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>networkInterface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>vsp</td>
<td>URI</td>
</tr>
<tr>
<td>hostname</td>
<td>string</td>
</tr>
<tr>
<td>macAddress</td>
<td>string</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
</tr>
<tr>
<td>allocation</td>
<td>string</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
</tr>
<tr>
<td>defaultGateway</td>
<td>string</td>
</tr>
<tr>
<td>dns</td>
<td>string</td>
</tr>
<tr>
<td>maxTransmissionUnit</td>
<td>integer</td>
</tr>
<tr>
<td>meters</td>
<td>ref[]</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-Machine+json

**JSON serialization**:

```json
{
  "self": string,
  "name": string,
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "state": string,
  "state": string,
  "networkInterfaces": [
    { "network": { "href": string }, "address": string, "hostname", string }, +
  ], ?,
  "cpu": string,
  "memory": { "quantity": integer, "units": string },
  "disks" : [ 
    { "capacity": { "quantity": integer, "units": string } }, +
  ], ?,
  "volumes" : [ 
    { "volume": { "href": string },
      "attachmentPoint": string,
      "protocol": string } +
  ], ?,
  "networkInterfaces": [ 
    { "vsp": {"href": string}, "hostname": string, "macAddress": string,
      "state": string, "protocol": string, "allocation": string,
      "address": string, "defaultGateway": string, "dns": string,
      "maxTransmissionUnit": integer }, +
```
DSP0263  Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP

```
}, ?
"meters": [
  { "href": "string " }, +
  ], ?
"eventLog": { "href": "string " }, ?
"operations": [ 
  { "rel": "edit", "href": "string " }, ?
  { "rel": "delete", "href": "string " }, ?
  { "rel": "http://www.dmtf.org/cimi/action/start", "href": "string " }, ?
  { "rel": "http://www.dmtf.org/cimi/action/stop", "href": "string " }, ?
  { "rel": "http://www.dmtf.org/cimi/action/restart", "href": "string " } ?
]
... 
```

XML media type: application/CIMI-Machine+xml

XML serialization:

```
<Machine xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string" xs:string </property> *
  <state> xs:string </state>
  <networkInterface network="xs:anyURI" address="xs:string"
                      hostname="xs:string"> *
    <cpu> xs:string </cpu>
    <memory quantity="xs:integer" units="xs:string"/>
    <disk>
      <capacity quantity="xs:integer" units="xs:string"/>
    </disk> *
    <volume href="xs:anyURI" attachmentPoint="xs:string" protocol="xs:string"/> *
    <networkInterface>
    <vsp href="xs:anyURI"/>
    <hostname> xs:string </hostname>
    <macAddress> xs:string </macAddress>
    <state> xs:string </state>
    <protocol> xs:string </protocol>
    <allocation> xs:string </allocation>
    <address> xs:string </address>
    <defaultGateway> xs:string </defaultGateway>
    <dns> xs:string </dns>
    <maxTransmissionUnit> xs:integer </maxTransmissionUnit>
    </networkInterface> *
    <meter href="xs:anyURI"/> *
    <eventLog href="xs:anyURI"/> ?
    <operation rel="edit" href="xs:anyURI"/> ?
    <operation rel="delete" href="xs:anyURI"/> ?
    <operation rel="http://www.dmtf.org/cimi/action/start" href="xs:anyURI"/> ?
    <operation rel="http://www.dmtf.org/cimi/action/stop" href="xs:anyURI"/> ?
    <operation rel="http://www.dmtf.org/cimi/action/restart" href="xs:anyURI"/> ?
    <xs:any>*
</Machine>
```

5.10.7.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Machine Collection entity.

The following custom operations are also defined:
Starting a Machine

/link@rel: http://www.dmtf.org/cimi/action/start

This operation will start a Machine.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation the Machine will be in the "STARTED" state. Starting a machine is the virtual equivalent of powering on a physical machine. There is no restored CPU or Memory state, so the guest OS will typically perform it's boot or installation tasks.

- HTTP/REST Protocol

To start a Machine a POST is sent to the "http://www.dmtf.org/cimi/start" URI of the Machine where the HTTP request body SHALL be as described below.

JSON media type: application/CIMI-Action+json

JSON serialization:

```
{ "action": "http://www.dmtf.org/cimi/action/start" ,
"properties": { "name": string, + } ?
... 
}
```

XML media type: application/CIMI-Action+xml

XML serialization

```
<Action xmlns="http://www.dmtf.org/cimi">
  <action> http://www.dmtf.org/cimi/action/start </action>
  <property name="xs:string"> xs:string </property> *
  <xs:any>*
</Action>
```

Upon successful processing of the request, the HTTP response body will be empty.

Stopping a Machine

/link@rel: http://www.dmtf.org/cimi/action/stop

This operation will stop, or shutdown, a Machine.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation the Machine will be in the "STOPPED" state. Stopping a machine is the virtual equivalent of powering off a physical machine. There is no saved CPU or Memory state.

- HTTP/REST Protocol

To stop a Machine a POST is sent to the "http://www.dmtf.org/cimi/stop" URI of the Machine where the HTTP request body SHALL be as described below.
JSON media type: application/CIMI-Action+json

JSON serialization:

```
{ "action": "http://www.dmtf.org/cimi/action/stop" ,
  "properties": { "name": string, + } ?
  ...
}
```

XML media type: application/CIMI-Action+xml

XML serialization

```
<Action xmlns="http://www.dmtf.org/cimi">
  <action> http://www.dmtf.org/cimi/action/stop </action>
  <property name="xs:string"> xs:string </property> *
  <xs:any>*
</Action>
```

Upon successful processing of the request, the HTTP response body will be empty.

Restarting a Machine

/link@rel: http://www.dmtf.org/cimi/action/restart

This operation will restart a Machine. If the Machine is in the "STARTED" state then this operation will have the semantic effect of executing the "stop" and then "start" operations. If the Machine is in the "STOPPED" state then this operation will have the semantic effect of executing the "start" operation.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation the Machine will be in the "STARTED" state. Restarting a machine is the virtual equivalent of powering off, then on a physical machine. There is no restored CPU or Memory state, so the guest OS will typically perform it’s boot or installation tasks.

- HTTP/REST Protocol

To restart a Machine a POST is sent to the "http://www.dmtf.org/cimi/restart" URI of the Machine where the HTTP request body SHALL be as described below.

JSON media type: application/CIMI-Action+json

JSON serialization:

```
{ "action": "http://www.dmtf.org/cimi/action/restart" ,
  "properties": { "name": string, + } ?
  ...
}
```

XML media type: application/CIMI-Action+xml

XML serialization

```
<Action xmlns="http://www.dmtf.org/cimi">
  <action> http://www.dmtf.org/cimi/action/restart </action>
  <property name="xs:string"> xs:string </property> *
  <xs:any>*
</Action>
```

Upon successful processing of the request, the HTTP response body will be empty.
5.10.8 Machine Collection

A Machine Collection resource represents the collection of Machine entities within a Provider. This resource can be used to locate and create Machines.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MachineCollection</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineCollection">http://www.dmtf.org/cimi/MachineCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machines</td>
<td>ref[]</td>
<td>An array of references to the set of Machines in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-MachineCollection+json

**JSON serialization**:

```json
{  
  "self": string,
  "name": string,  
  "description": string,  
  "created": string,  
  "properties": { "name": string, + },  
  "machines": [
    { "href": string },  
    ],  
  "operations": [
    { "rel": "add", "href": string },  
    { "rel": "edit", "href": string }  
    ],  
  ...
}
```

**XML media type**: application/CIMI-MachineCollection+xml

**XML serialization**:

```xml
<MachineCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <machine href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</MachineCollection>
```

5.10.8.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

- Creating a New Machine

/userset@rel: add
This operation will create a new Machine.

Input parameters: Either a reference to a Machine Template or a Machine Template itself.

Output parameters: A reference to a new Machine and optionally the representation of the Machine.

**HTTP/REST Protocol**

To create a new Machine a POST is sent to the "add" URI of the MachineCollection where the HTTP request body SHALL be as described below. Note this structure allows for certain properties to be passed in "by value" or by "reference". The definition of each property can be found in section 5.10.1.

**JSON media type:** application/CIMI-MachineCreate+json

**JSON serialization:**

```json
{
  "name": string,
  "description": string,
  "properties": { "name": string, + }, ?
  "machineTemplate": { "href": string, ?
    "properties": { "name": string, + }, ?
    "machineConfig": { "href": string, ?
      "cpu": string, ?
      "memory": { "quantity": integer, "units": string }, ?
      "disks": [
        { "capacity": { "quantity": integer, "units": string }, ?
          "guestInterface": string }, +
      ] ?
    }, ?
    "machineImage": { "href": string,
      "imageLocation": { "href": string }, ?
      "imageData": string, ?
    }, ?
    "machineAdmin": { "href": string, ?
      <provider specific data> ?
    }, ?
    "volumes": [
      { "href": string, "attachmentPoint": string, "protocol": string}, +
    ], ?
    "volumeTemplates": [
      { "href": string, ?
        "attachmentPoint": string, "protocol": string,
        "volumeConfig": { "href": string, ?
          "properties": { "name": string, + }, ?
          "format": string, ?
          "capacity": { "quantity": number, "units": string }, ?
          "supportsSnapshots": boolean, ?
          "guestInterface": string ?
        }, ?
        "volumeImage": { "href": string,
          "properties": { "name": string, + }, ?
          "imageLocation": { "href": string }, ?
          "imageData": string, ?
          "bootable": boolean ?
        } ?
      }, +
    ]
}
...
**XML media type:** application/CIMI-MachineCreate+xml

**XML serialization**

```xml
<MachineCreate>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <property name="xs:string"> xs:string </property> *

  <machineTemplate href="xs:anyURI"? >
    <property name="xs:string"> xs:string </property> *
  </machineTemplate>

  <machineConfig href="xs:anyURI"? >
    <property name="xs:string"> xs:string </property> *
    <cpu> xs:string </cpu> ?
    <memory quantity="xs:integer" units="xs:string"/> ?
    <disk>
      <capacity quantity="xs:integer" units="xs:string">?
        <guestInterface> xs:string </guestInterface>
      </disk> *
    </machineConfig>

  <machineImage href="xs:anyURI">? 
    <property name="xs:string"> xs:string </property> *
    <imageLocation href="xs:anyURI"/> ?
    <imageData> xs:string </imageData> ?
  </machineImage>

  <machineAdmin href="xs:anyURI"? >
    <property name="xs:string"> xs:string </property> *
    <machineAdmin> ? 
      -- provider specific data -->
    <volume href="xs:anyURI">
      attachmentPoint="xs:string" protocol="xs:string" />
    </volume>

  <volumeTemplate href="xs:anyURI">?
    attachmentPoint="xs:string" protocol="xs:string" >
  </volumeTemplate>

  <volumeConfig href="xs:anyURI">?
    property name="xs:string"> xs:string </property> *
    <format> xs:string </format> ?
    <capacity quantity="xs:integer" units="xs:string">? 
      <supportsSnapshots> xs:boolean </supportsSnapshots> ?
      <guestInterface> xs:string </guestInterface> ?
  </volumeConfig>

  <volumeImage href="xs:anyURI">?
    property name="xs:string"> xs:string </property> *
    <imageLocation href="xs:anyURI"/> ?
    <imageData> xs:any */ <imageData> ?
    <bootable> xs:boolean </bootable> ?
  </volumeImage> ?

  <volumeTemplate> *
  </volumeTemplate>
</MachineCreate>
```

The serialization of some reference properties are specified such that a request MAY either include a reference ("href") to an existing entity or to include the entity "inline" as a set of additional properties. Requests SHALL NOT include both a reference and the inlined set of properties.

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the Machine entity.
5.10.9 Machine Admin

A Machine Admin entity contains the information required to create the initial administrative super-user of a newly created Machine. Due to the variation between operating systems and Providers, this specification does not mandate one particular set of attributes that all implementations need to support. However, Providers are expected to extend this entity with additional attributes to meet their requirements.

For example, a Provider might extend this entity with username and password attributes, which would then be the login information for new Machines. These extension attributes would appear as siblings to the common attributes like 'name' and 'description'.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineAdmin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineAdmin">http://www.dmtf.org/cimi/MachineAdmin</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>string</td>
<td>The initial superuser's user name.</td>
</tr>
<tr>
<td>password</td>
<td>string</td>
<td>Initial superuser's password.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>byte[]</td>
<td>The digit of the public key for the initial superuser.</td>
</tr>
</tbody>
</table>

JSON media type: application/CIMI-MachineAdmin+json

```json
{ "self": string,  
  "name": string, ?  
  "description": string, ?  
  "created": string, ?  
  "properties": { "name": string, + }, ?  
  "operations": [  
    { "rel": "edit", "href": string } ?  
    { "rel": "delete", "href": string } ?  
  ] ?  
...  
}
```
XML media type: application/CIMI-MachineAdmin+xml

XML serialization:

```
<MachineAdminCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any> *</xs:any>
</MachineAdminCollection>
```

5.10.9.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Machine Admin Collection entity.

5.10.10 Machine Admin Collection

A Machine Admin Collection entity represents the collection of Machine Admin entities within a Provider. This entity can be used to locate and create MachineAdmins.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineAdminCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MachineAdminCollection">http://www.dmtf.org/cimi/MachineAdminCollection</a></td>
</tr>
</tbody>
</table>

```
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machineAdmins</td>
<td>ref[]</td>
<td>An array of references to the set of Machine Admins in the provider.</td>
</tr>
</tbody>
</table>
```

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

JSON media type: application/CIMI-MachineAdminCollection+json

```
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "machineAdmins": [ { "href": string }, + ], ?
  "operations": [ { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string } ? ]
    ...
}
```

XML media type: application/CIMI-MachineAdminCollection+xml

XML serialization:

```
<MachineAdminCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
```

Version 0.0.46a — DMTF — Work In Progress - not a DMTF Standard
### 5.10.10.1 Operations

This entity supports the Read and Update operations. Creation of new Machine Admin entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

### 5.11 Volume Entities and Relationships

The following diagram illustrates the entities involved in constructing a Volume and their relationships. Although this drawing is in the style of an Entity Relationship diagram, the use of UML is neither rigorous nor normative.

![Diagram of Volume Entities and Relationships]

**Figure 3 - Volume Entities**
5.11.1 Volume Template

This entity captures the configuration values for realizing a Volume. A Volume Template may be used to create multiple Volumes.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VolumeTemplate">http://www.dmtf.org/cimi/VolumeTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumeConfig</td>
<td>ref</td>
<td>A reference to the Volume Configuration that will be used to create a Volume from this Volume Template.</td>
</tr>
<tr>
<td>volumImage</td>
<td>ref</td>
<td>A reference to the Volume Image that will be used to create a Volume from this Volume Template.</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-VolumeTemplate+json

**JSON serialization:**

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "volumeConfig": { "href": string },
  "volumeImage": { "href": string },
  "operations": [
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?
  ] ?
... }
```

**XML media type:** application/CIMI-VolumeTemplate+xml

**XML serialization:**

```xml
<VolumeTemplate xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <volumeConfig href="xs:anyURI"/>
  <volumeImage href="xs:anyURI"/>
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeTemplate>
```

5.11.1.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Volume Template Collection entity.
5.11.2 Volume Template Collection

A Volume Template Collection entity represents the collection of VolumeTemplate entities within a Provider. This entity can be used to locate and create Volume Templates.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeTemplateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VolumeTemplateCollection">http://www.dmtf.org/cimi/VolumeTemplateCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumeTemplates</td>
<td>ref[]</td>
<td>An array of references to the set of Volume Templates in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-VolumeTemplateCollection+json

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "volumeTemplates": [
    { "href": string }, +
  ], ?,
  "operations": [
    { "rel": "add", "href": string }, ?,
    { "rel": "edit", "href": string }?
  ] ?,
  ...
}
```

**XML media type**: application/CIMI-VolumeTemplateCollection+xml

```xml
<VolumeTemplateCollection xmlns="http://www.dmtf.org/cimi">
  <self xs:anyURI /></self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <volumeTemplate href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeTemplateCollection>
```

5.11.2.1 Operations

This entity supports the Read and Update operations. Creation of new Volume Template entities is supported via a POST to the “addLink” URI as described in section 4.2.2.1.

5.11.3 Volume Configuration

The Volume Configuration entity represents the set of configuration values needed to create a Volume with certain characteristics. Volume Configurations are created by Providers and MAY, at the Providers discretion, be created by Consumers.
Name | VolumeConfiguration  
---|---  
Type URI | http://www.dmtf.org/cimi/VolumeConfiguration  

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| format | String | The format of the file system that will be placed on Volumes created from this configuration. This attribute is only meaningful for Volume Configurations that describe block devices. This attribute is optional; the absence of this attribute indicates that Volumes created from this configuration will not be formatted with a file system. Example values: "ext4", "ntfs".  
*Properties:* Mandatory / Mutable |
| capacity | structure | The default size, when limited, of the Volume created from this Volume Configuration.  
This attribute has the following, sub-attributes.  

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| quantity | integer | A numerical quantity expressed as an integer.  
*Properties:* Mandatory / Mutable |
| units | String | An enumerated value that expresses the unit of measurement used. Allowable values are byte, kilobyte, megabyte, gigabyte, terabyte, petabyte, exabyte, zettabyte, and yottabyte.  
*Properties:* Mandatory / Mutable |
| supportsSnapshots | boolean | This property indicates whether Volumes created from this Volume Configuration will support the ability to take snapshots.  
*Properties:* Mandatory / Mutable |
| guestInterface | String | This property indicates the interface that will be offered to a Machine instances by Volumes created from this Volume Configuration.  
*Properties:* Mandatory / Mutable |

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-VolumeConfiguration+json

**JSON serialization:**

```json
{  
  "self": string,  
  "name": string,  
  "description": string,  
  "created": string,  
  "properties": { "name": string, + },  
  "format": string,  
  "capacity": { "quantity": number, "units": string },  
}```
"supportsSnapshots": boolean,
"guestInterface": string,
"operations": [ 
  { "rel": "edit", "href": string }, ?
  { "rel": "delete", "href": string } ?
] ?
...

XML media type: application/CIMI-VolumeConfiguration+xml

XML serialization:

```xml
<VolumeConfiguration xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <format> xs:string </format>
  <capacity_quantity="xs:integer" units="xs:string"/>
  <supportsSnapshots> xs:boolean </supportsSnapshots>
  <guestInterface> xs:string </guestInterface>
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeConfiguration>
```

5.11.3.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Volume Configuration Collection entity.

5.11.4 Volume Configuration Collection

A Volume Configuration Collection entity represents the collection of Volume Configuration entities within a Provider. This entity can be used to locate and create Volume Configurations.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeConfigurationCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VolumeConfigurationCollection">http://www.dmtf.org/cimi/VolumeConfigurationCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumeConfigurations</td>
<td>ref[]</td>
<td>An array of references to the set of Volume Configurations in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

JSON media type: application/CIMI-VolumeConfigurationCollection+json

JSON serialization:

```json
{ "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "volumeConfigurations": [ 
    { "href": string }, +
  ]
```
XML media type: application/CIMI-VolumeConfigurationCollection+xml

XML serialization:

```
<VolumeConfigurationCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <volumeConfiguration href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeConfigurationCollection>
```

5.11.4.1 Operations

This entity supports the Read and Update operations. Creation of new Volume Image entities is supported via a POST to the “addLink” URI as described in section 4.2.2.1.

5.11.5 Volume Image

This entity represents an image that could be placed on a pre-loaded volume.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeImage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VolumeImage">http://www.dmtf.org/cimi/VolumeImage</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageLocation</td>
<td>ref</td>
<td>A reference to the location of the binary data that makes up this image. Either this attribute or imageData SHALL be present; however, both values SHALL NOT be present simultaneously.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Immutable</td>
</tr>
<tr>
<td>imageData</td>
<td>byte[]</td>
<td>The binary data that makes up this image. Either this attribute or imageLocation SHALL be present; however, both values SHALL NOT be present simultaneously.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Immutable</td>
</tr>
<tr>
<td>bootable</td>
<td>boolean</td>
<td>This property indicates whether Volumes created from this Volume Configuration will be bootable.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:
JSON media type: application/CIMI-VolumeImage+json

JSON serialization:

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "imageLocation": { "href": string }, ?
  "imageData": string, ?
  "bootable": boolean,
  "operations": [ 
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?
  ] ?
...
}
```

XML media type: application/CIMI-VolumeImage+xml

XML serialization:

```xml
<VolumeImage xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string" xs:string </property> *
  <imageLocation href="xs:anyURI"/> ?
  <imageData> xs:any </imageData> ?
  <bootable> xs:boolean </bootable>
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeImage>
```

5.11.5.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Volume Image Collection entity.

5.11.6 Volume Image Collection

A Volume Image Collection entity represents the collection of VolumeImage entities within a Provider. This entity can be used to locate and create Volume Images.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeImageCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VolumeImageCollection">http://www.dmtf.org/cimi/VolumeImageCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumImages</td>
<td>ref[]</td>
<td>An array of references to the set of Volume Images in the Provider.</td>
</tr>
<tr>
<td>Properties:</td>
<td></td>
<td>Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:
JSON media type: application/CIMI-VolumeImageCollection+json

JSON serialization:

```
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "volumeImages": [
    { "href": string }, +
  ], ?
  "operations": [?
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string }?
  ]?
  ...
}
```

XML media type: application/CIMI-VolumeImageCollection+xml

XML serialization:

```
<VolumeImageCollection xmlns="http://www.dmtf.org/cimi">
  <self>xs:anyURI</self>
  <name>xs:string</name> ?
  <description>xs:string</description> ?
  <created>xs:string</created>
  <property name="xs:string">xs:string</property> *
  <volumeImage href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeImageCollection>
```

5.11.6.1 Operations

This entity supports the Read and Update operations. Creation of new Volume Image entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.11.7 Volume

A Volume represents storage at either the block or file-system level. Volumes can be attached to Machines. Once attached, Volumes can be accessed by processes on that Machine.

<table>
<thead>
<tr>
<th>Name</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/Volume">http://www.dmtf.org/cimi/Volume</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capacity</td>
<td>structure</td>
<td>The maximum size, when limited, of the Volume. When this value is increased, the Volume can contain more data. Decreasing this value may require evaluations. This attribute has the following, sub-attributes.</td>
</tr>
<tr>
<td>quantity</td>
<td>integer</td>
<td>A numerical quantity expressed as an</td>
</tr>
</tbody>
</table>
The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-Volume+json

**JSON serialization:**

```json
{"self": string,
 "name": string, ?
 "description": string, ?
 "created": string, ?
 "properties": { "name": string, + }, ?
 "capacity": { "quantity": number, "units": string },
 "bootable": boolean,
 "supportsSnapshots": boolean,
 "snapshots": [
   { "href": string }, +
 ], ?
 "guestInterface": string,
 "meters": [
   { "href": string }, +
 ```
```
}, ?,
"eventLog": [ "href": string ], ?,
"operations": [ {
    "rel": "edit", "href": string }, ?,
    { "rel": "delete", "href": string } ] ?,
] ?,
...
```

**XML media type:** application/CIMI-Volume+xml

**XML serialization:**

```xml
<Volume xmlns="http://www.dmtf.org/cimi">
    <self xs:anyURI /></self>
    <name xs:string /></name> ?,
    <description> xs:string </description> ?,
    <created> xs:string </created>
    <property name="xs:string" xs:string /> *
    <capacity quantity="xs:integer" units="xs:string"/>
    <bootable> xs:boolean </bootable>
    <supportsSnapshots> xs:boolean </supportsSnapshots>
    <snapshot href="xs:anyURI"/> *
    <guestInterface> xs:string </guestInterface>
    <meter href="xs:anyURI"/> *
    <eventLog href="xs:anyURI"/> ?
    <operation rel="edit" href="xs:anyURI"/> ?
    <operation rel="delete" href="xs:anyURI"/> ?
    <xs:any>*
</Volume>
```

### 5.11.7.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Volume Collection entity.

### 5.11.8 Volume Collection

A Volume Collection entity represents the collection of Volumes within a Provider. This entity can be used to locate and create Volumes.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VolumeCollection">http://www.dmtf.org/cimi/VolumeCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumes</td>
<td>ref[]</td>
<td>An array of references to the set of Volumes in the provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-VolumeCollection+json

**JSON serialization:**

```json
{
    "self": string,
    "name": string, ?
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
}
"volumes": [  
  { "href": string }, +
}, ?
"operations": [  
  { "rel": "add", "href": string }, ?
  { "rel": "edit", "href": string } ?
} ?
...
]

**XML media type:** application/CIMI-VolumeCollection+xml

**XML serialization:**

```xml
<VolumeCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <volume href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VolumeCollection>
```

### 5.11.8.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

**Creating a New Volume**

`/link@rel`: add

This operation will create a new Volume.

Input parameters: Either a reference to a Volume Template or a Volume Template itself.

Output parameters: A reference to a new Volume and optionally the representation of the Volume.

**HTTP/REST Protocol**

To create a new Volume a POST is sent to the "add" URI of the VolumeCollection where the HTTP request body SHALL be as described below. Note this structure allows for certain properties to be passed in "by value" or by "reference". The definition of each property can be found in section 5.11.1.

**JSON media type:** application/CIMI-VolumeCreate+json

**JSON serialization:**

```json
{
  "name": string,
  "description": string, ?
  "properties": { "name": string, + }, ?
  "volumeTemplate": { "href": string, ?
    "properties": { "name": string, + }, ?
    "volumeConfig": { "href": string, ?
      "properties": { "name": string, + }, ?
      "format": string, ?
      "capacity": { "quantity": number, "units": string }, ?
      "supportsSnapshots": boolean, ?
      "guestInterface": string, ?
```
The serialization of some reference properties are specified such that a request MAY either include a reference ("href") to an existing entity or to include the entity "inline" as a set of additional properties. Requests SHALL NOT include both a reference and the inlined set of properties.

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the Volume entity.

5.12 Network Entities and Relationships

The following diagram illustrates the entities involved in constructing Networks and their Virtual Switch Ports (VSPs) and their relationships. Although this drawing is in the style of an Entity Relationship diagram, the use of UML is neither rigorous nor normative.
Figure 4 - Network Entities

5.12.1 Network Template

The Network Template is a set of configuration values for realizing a Network. An instance of Network Template may be used to create multiple Networks.

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkTemplate</th>
</tr>
</thead>
</table>

DSP0263 Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP
The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-Networ珵Template+json

**JSON serialization:**

```json
{
    "self": string,
    "name": string, ?
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
    "networkConfig": { "href": string },
    "routingGroup": { "href": string }, ?
    "operations": [ 
        { "rel": "edit", "href": string }, ?
        { "rel": "delete", "href": string }?
    ] ?
    ...
}
```

**XML media type:** application/CIMI-Networ珵Template+xml

**XML serialization:**

```xml
<NetworkTemplate xmlns="http://www.dmtf.org/cimi">
    <self> xs:anyURI </self>
    <name> xs:string </name> ?
    <description> xs:string </description> ?
    <created> xs:string </created>
    <property name="xs:string"> xs:string </property> *
    <networkConfig href="xs:anyURI"/>
    <routingGroup href="xs:anyURI"/> ?
    <operation rel="edit" href="xs:anyURI"/> ?
    <operation rel="delete" href="xs:anyURI"/> ?
    <xs:any>*
</NetworkTemplate>
```

### 5.12.1.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Network Template Collection entity.
5.12.2 Network Template Collection

A Network Template Collection entity represents the collection of NetworkTemplate as within a Provider.
This resource can be used to locate and create NetworkTemplates.

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkTemplateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/NetworkTemplateCollection">http://www.dmtf.org/cimi/NetworkTemplateCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networkTemplates</td>
<td>ref</td>
<td>An array of references to the set of Network Templates in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-NetworkTemplateCollection+json

**JSON serialization:**
```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "networkTemplates": [
    { "href": string }, +
  ], ?,
  "operations": [
    { "rel": "add", "href": string }, ?,
    { "rel": "edit", "href": string } ?
  ] ?,
...
}
```

**XML media type:** application/CIMI-NetworkTemplateCollection+xml

**XML serialization:**
```xml
<NetworkTemplateCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <NetworkTemplate href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</NetworkTemplateCollection>
```

5.12.2.1 Operations

This entity supports the Read and Update operations. Creation of new Network Template entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.12.3 Network Configuration

The set of configuration values representing the information needed to create a Network with certain characteristics.
Name: NetworkConfiguration

Type URI: http://www.dmtf.org/cimi/NetworkConfiguration

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>access</td>
<td>string</td>
<td>An indicator of whether or not the Machine entity has access to a Public or Private network. An indication of Public represents an open and Internet routable network. An indication of Private identifies a local non-routed network.</td>
</tr>
<tr>
<td>Properties:</td>
<td>Optional / Mutable</td>
<td></td>
</tr>
<tr>
<td>bandwidthLimit</td>
<td>integer</td>
<td>Maximum allowable bandwidth.</td>
</tr>
<tr>
<td>Properties:</td>
<td>Optional / Mutable</td>
<td></td>
</tr>
<tr>
<td>trafficPriority</td>
<td>integer</td>
<td>Indicates priority of traffic on this network.</td>
</tr>
<tr>
<td>Properties:</td>
<td>Optional / Mutable</td>
<td></td>
</tr>
<tr>
<td>maxTrafficDelay</td>
<td>integer</td>
<td>The requested maximum delay for end to end transmission specified in nanoseconds with uint64 (i.e. latency).</td>
</tr>
<tr>
<td>Properties:</td>
<td>Optional / Mutable</td>
<td></td>
</tr>
<tr>
<td>maxTrafficLoss</td>
<td>integer</td>
<td>The requested maximum percentage traffic loss for end to end transmission with uint8, min 0 max 100. The percentage of traffic lost in the transmission traffic. A value of zero indicates that a lossless transmission is requested. A value of 100 indicates a best effort transmission. The default value is 100.</td>
</tr>
<tr>
<td>Properties:</td>
<td>Optional / Mutable</td>
<td></td>
</tr>
<tr>
<td>maxTrafficJitter</td>
<td>integer</td>
<td>The requested maximum jitter for end to end transmission with uint32 when traffic is packetized. The variation between packets arriving specified in nanoseconds with uint64.</td>
</tr>
<tr>
<td>Properties:</td>
<td>Optional / Mutable</td>
<td></td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-NetworkConfiguration+json

**JSON serialization:**

```json
{
    "self": string,
    "name": string,  
    "description": string,  
    "created": string,  
    "properties": { "name": string, + },  
    "access": string,  
    "bandwidthLimit": number,  
    "trafficPriority": number,  
    "maxTrafficDelay": number,  
    "maxTrafficLoss": number,  
    "maxTrafficJitter": number,  
    "operations": [  
        { "rel": "edit", "href": string },  
        { "rel": "delete", "href": string }  
    ]
}
```
XML media type: application/CIMI-NetworkConfiguration+xml

XML serialization:

```
<NetworkConfiguration xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <access> xs:string </access> ?
  <bandwidthLimit> xs:string </bandwidthLimit> ?
  <trafficPriority> xs:integer </trafficPriority> ?
  <maxTrafficDelay> xs:integer </maxTrafficDelay> ?
  <maxTrafficLoss> xs:integer </maxTrafficLoss> ?
  <maxTrafficJitter> xs:integer </maxTrafficJitter> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</NetworkConfiguration>
```

5.12.3.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Network Configuration Collection entity.

5.12.4 Network Configuration Collection

A Network Configuration Collection entity represents the collection of Network Configurations within a Provider. This entity can be used to locate and create Network Configurations.

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkConfigurationCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/NetworkConfigurationCollection">http://www.dmtf.org/cimi/NetworkConfigurationCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networkConfigurations</td>
<td>ref[]</td>
<td>An array of references to the set of Network Configurations in the provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

JSON media type: application/CIMI-NetworkConfigurationCollection+json

JSON serialization:

```
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "networkConfigurations": [
    { "href": string }, +
  ], ?
  "operations": [
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string } ?
  ]
```

XML media type: application/CIMI-NetworkConfigurationCollection+xml

XML serialization:

```xml
<NetworkConfigurationCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <networkConfiguration href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  </xs:any>*
</NetworkConfigurationCollection>
```

5.12.4.1 Operations
This entity supports the Read and Update operations. Creation of new Network Collection entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.12.5 Network
A Network is a realized entity that represents an abstraction of a layer 2 broadcast domain.

<table>
<thead>
<tr>
<th>Name</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/Network">http://www.dmtf.org/cimi/Network</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>string</td>
<td>Indicates the operational state of the Network. For example, STARTED, STOPPED, SUSPENDED are operational states. This value is <strong>read-only</strong> and will change based on the state of the Network. <strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>access</td>
<td>string</td>
<td>An indicator of whether or not the Machine entity has access to a Public or Private network. An indication of Public represents an open and Internet routable network. An indication of Private identifies a local non-routed network. <strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>bandwidthLimit</td>
<td>integer</td>
<td>Maximum allowable bandwidth. <strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>trafficPriority</td>
<td>integer</td>
<td>Indicates priority of traffic on this network. <strong>Properties:</strong> Optional / Mutable</td>
</tr>
<tr>
<td>maxTrafficDelay</td>
<td>integer</td>
<td>The requested maximum delay for end to end transmission specified in nanoseconds with uint64 (i.e. latency). <strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>
maxTrafficLoss | integer | The requested maximum percentage traffic loss for end to end transmission with uint8, min 0 max 100. The percentage of traffic lost in the transmission traffic. A value of zero indicates that a lossless transmission is requested. A value of 100 indicates a best effort transmission. The default value is 100.  
**Properties**: Optional / Mutable

maxTrafficJitter | integer | The requested maximum jitter for end to end transmission with uint32 when traffic is packetized. The variation between packets arriving specified in nanoseconds with uint64.  
**Properties**: Optional / Mutable

routingGroup | ref | A reference to a RoutingGroup that this Network is part of.  
Note that Networks route to themselves, therefore this attribute will only appear in cases where the Network routes to one or more additional Networks.  
**Properties**: Optional / Mutable

meters | ref[] | A list of references to Meters monitored for this Network.  
**Properties**: Optional / Mutable

eventLog | ref | A reference to the EventLog of this Network.  
**Properties**: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-Network+json

**JSON serialization**:

```
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "state": string,
  "access": string, ?
  "bandwidthLimit": number, ?
  "trafficPriority": number, ?
  "maxTrafficDelay": number, ?
  "maxTrafficLoss": number, ?
  "maxTrafficJitter": number, ?
  "routingGroup": { "href": string }, ?
  "meters": [ 
    { "href": string }, +
  ], ?
  "eventLog": { "href": string }, ?
  "operations": [
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?
  ] ?
  ...
}
```
**XML media type:** application/CIMI-Network+xml

**XML serialization:**

```xml
<Network xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <state> xs:string </state>
  <access> xs:string </access> ?
  <bandwidthLimit> xs:integer </bandwidthLimit> ?
  <trafficPriority> xs:integer </trafficPriority> ?
  <maxTrafficDelay> xs:integer </maxTrafficDelay> ?
  <maxTrafficLoss> xs:integer </maxTrafficLoss> ?
  <maxTrafficJitter> xs:integer </maxTrafficJitter> ?
  <routingGroup href="xs:anyURI"/> ?
  <meter href="xs:anyURI"/> *
  <eventLog" href="xs:anyURI"/> ?
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</Network>
```

### 5.12.5.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Network Collection entity.

### 5.12.6 Network Collection

A Network Collection entity represents the collection of Networks within a Provider. This entity can be used to locate and create Networks.

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/NetworkCollection">http://www.dmtf.org/cimi/NetworkCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networks</td>
<td>ref[]</td>
<td>An array of references to the set of Networks in the Provider.</td>
</tr>
</tbody>
</table>

**Properties:** Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-NetworkCollection+json

**JSON serialization:**

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "networks": [ 
    { "href": string }, +
  ], ?
  "operations": [ 
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string } ?
  ]
}
```
XML media type: application/CIMI-NetworkCollection+xml

XML serialization:

```xml
<NetworkCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <network href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
</NetworkCollection>
```

5.12.6.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

**Creating a New Network**

/link@rel: add

This operation will create a new Network.

Input parameters: Either a reference to a Network Template or a Network Template itself.

Output parameters: A reference to a new Network and optionally the representation of the Network.

- HTTP/REST Protocol

To create a new Network a POST is sent to the "add" URI of the NetworkCollection where the HTTP request body SHALL be as described below. Note this structure allows for certain properties to be passed in "by value" or by "reference". The definition of each property can be found in section 5.12.1.

JSON media type: application/CIMI-NetworkCreate+json

JSON serialization:

```json
{
  "name": string,
  "description": string, ?
  "properties": { "name": string, + }, ?
  "networkTemplate": { "href": string, ?
    "properties": { "name": string, + }?
  }?
  "networkConfig": { "href": string, ?
    "properties": { "name": string, + }?,
    "access": string, ?
    "bandwidthLimit": number, ?
    "trafficPriority": number, ?
    "maxTrafficDelay": number, ?
    "maxTrafficLoss": number, ?
    "maxTrafficJitter": number, ?
  },
  "routingGroup": { "href": string }?
}
```
XML media type: application/CIMI-NetworkCreate+xml

XML serialization

```xml
<NetworkCreate>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <property name="xs:string"> xs:string </property> *
  <networkTemplate href="xs:anyURI"? >
    <property name="xs:string"> xs:string </property> *
    <networkConfig href="xs:anyURI"? >
      <access> xs:string </access> ?
      <bandwidthLimit> xs:string </bandwidthLimit> ?
      <trafficPriority> xs:integer </trafficPriority> ?
      <maxTrafficDelay> xs:integer </maxTrafficDelay> ?
      <maxTrafficLoss> xs:integer </maxTrafficLoss> ?
      <maxTrafficJitter> xs:integer </maxTrafficJitter> ?
    </networkConfig>
    <routingGroup href="xs:anyURI"/> ?
  </networkTemplate>
</NetworkCreate>
```

The serialization of some reference properties are specified such that a request MAY either include a reference ("href") to an existing entity or to include the entity "inline" as a set of additional properties. Requests SHALL NOT include both a reference and the inlined set of properties.

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the Network entity.

5.12.7 VSP (Virtual Switch Port) Template

The VSP Template is a set of Configuration values for realizing a VSP. A VSP Template may be used to create multiple VSPs.

<table>
<thead>
<tr>
<th>Name</th>
<th>VSPTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VSPTemplate">http://www.dmtf.org/cimi/VSPTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network</td>
<td>ref</td>
<td>A reference to the network to be associated with this VSP.</td>
</tr>
<tr>
<td>vspConfig</td>
<td>ref</td>
<td>A reference to the VSP Configuration that will be used to create a VSP from this VSP Template.</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

JSON media type: application/CIMI-VSPTemplate+json

JSON serialization:

```json
{ "self": string,
  "name": string, ?
  "description": string, ?
}```
5.12.7.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the VSP Template Collection entity.

5.12.8 VSP (Virtual Switch Port) Template Collection

A VSP Template Collection entity represents the collection of VSP Templates within a Provider. This entity can be used to locate and create VSP Templates.

<table>
<thead>
<tr>
<th>Name</th>
<th>VSPTemplateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VSPTemplateCollection">http://www.dmtf.org/cimi/VSPTemplateCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vspTemplates</td>
<td>ref[]</td>
<td>An array of references to the set of VSP Templates in the Provider.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-VSPTemplateCollection+json

```
{  
  "self": string,  
  "name": string,  
  "description": string,  
  "created": string,  
  "properties": {  
    "name": string,  
    "vspTemplates": [  
      {  
        "href": string,  
      }  
    ]  
  },  
  "network": {  
    "href": string,  
  },  
  "vspConfig": {  
    "href": string,  
  },  
  "operations": [  
    {  
      "rel": "edit",  
      "href": string,  
    },  
    {  
      "rel": "delete",  
      "href": string  
    }  
  ]  
}
```
XML media type: application/CIMI-VSPTemplateCollection+xml

XML serialization:

```
<VSPTemplateCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <vsTemplate href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VSPTemplateCollection>
```

5.12.8.1 Operations

This entity supports the Read and Update operations. Creation of new VSP Template entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.12.9 VSP (Virtual Switch Port) Configuration

The set of configuration values representing the information needed to create a VSP with certain characteristics.

<table>
<thead>
<tr>
<th>Name</th>
<th>VSPConfiguration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VSPConfiguration">http://www.dmtf.org/cimi/VSPConfiguration</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bandwidthReservation</td>
<td>integer</td>
<td>Minimum Bandwidth requirements.</td>
</tr>
<tr>
<td>trafficPriority</td>
<td>integer</td>
<td>Indicates priority of traffic on this network.</td>
</tr>
<tr>
<td>maxTrafficDelay</td>
<td>integer</td>
<td>The requested maximum delay for end to end transmission specified in nanoseconds with uint64 (i.e. latency).</td>
</tr>
<tr>
<td>maxTrafficLoss</td>
<td>integer</td>
<td>The requested maximum percentage traffic loss for end to end transmission with uint8, min 0 max 100. The percentage of traffic lost in the transmission traffic. A value of zero indicates that a lossless transmission is requested. A value of 100 indicates a best effort transmission. The default value is 100.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable
maxTrafficJitter | integer | The requested maximum jitter for end to end transmission with uint32 when traffic is packetized. The variation between packets arriving specified in nanoseconds with uint64.

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-VSPConfiguration+json

**JSON serialization**:

```json
{
    "self": string,
    "name": string,
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
    "bandwidthReservation": number, ?
    "trafficPriority": number, ?
    "maxTrafficDelay": number, ?
    "maxTrafficLoss": number, ?
    "maxTrafficJitter": number,?
    "operations": [ 
        { "rel": "edit", "href": string }, ?,
        { "rel": "delete", "href": string } ?
    ] ?
}
```

**XML media type**: application/CIMI-VSPConfiguration+xml

**XML serialization**:

```xml
<VSPConfiguration xmlns="http://www.dmtf.org/cimi">
    <self> xs:anyURI </self>
    <name> xs:string </name>
    <description> xs:string </description> ?
    <created> xs:string </created>
    <property name="xs:string"> xs:string </property> *
    <bandwidthReservation> xs:integer </bandwidthReservation> ?
    <trafficPriority> xs:integer </trafficPriority> ?
    <maxTrafficDelay> xs:integer </maxTrafficDelay> ?
    <maxTrafficLoss> xs:integer </maxTrafficLoss> ?
    <maxTrafficJitter> xs:integer </maxTrafficJitter> ?
    <operation rel="edit" href="xs:anyURI"/> ?
    <operation rel="delete" href="xs:anyURI"/> ?
    <xs:any>*
</VSPConfiguration>
```

**5.12.9.1 Operations**

This entity supports the Read, Update and Delete operations. Create is supported via the VSP Configuration Collection entity.

**5.12.10 VSP (Virtual Switch Port) Configuration Collection**

A VSP Configuration Collection entity represents the collection of VSP Configurations within a Provider. This entity can be used to locate and create VSP Configurations.

| Name                  | VSPConfigurationCollection |
Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP

<table>
<thead>
<tr>
<th>Type URI</th>
<th><a href="http://www.dmtf.org/cimi/VSPConfigurationCollection">http://www.dmtf.org/cimi/VSPConfigurationCollection</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>vspConfigurations</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-VSPConfigurationCollection+json

**JSON serialization:**

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "vspConfigurations": [ 
    { "href": string }, +
  ], ?,
  "operations": [ 
    { "rel": "add", "href": string }, ?,
    { "rel": "edit", "href": string }?
  ],
  ...
}
```

**XML media type:** application/CIMI-VSPConfigurationCollection+xml

**XML serialization:**

```xml
<VSPConfigurationCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <vspConfiguration href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VSPConfigurationCollection>
```

**5.12.10.1 Operations**

This entity supports the Read and Update operations. Creation of new VSP Configuration entities is supported via a POST to the “addLink” URI as described in section 4.2.2.1.

**5.12.11 VSP (Virtual Switch Port)**

A VSP represents the connection parameters of a network port.

<table>
<thead>
<tr>
<th>Name</th>
<th>VSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VSP">http://www.dmtf.org/cimi/VSP</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>string</td>
<td>An indicator of whether or not a specified port is on or off [Default =</td>
</tr>
</tbody>
</table>
Enabled].
This value is read-only and will change based on the state of the VSP.

**Properties:** Mandatory / Mutable

<table>
<thead>
<tr>
<th>network</th>
<th>ref</th>
<th>A reference to the network associated with this VSP.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bandwidthReservation</th>
<th>integer</th>
<th>Minimum Bandwidth requirements.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>trafficPriority</th>
<th>integer</th>
<th>Indicates priority of traffic on this network.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>maxTrafficDelay</th>
<th>integer</th>
<th>The requested maximum delay for end to end transmission specified in nanoseconds with uint64 (i.e. latency).</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>maxTrafficLoss</th>
<th>integer</th>
<th>The requested maximum percentage traffic loss for end to end transmission with uint8, min 0 max 100. The percentage of traffic lost in the transmission traffic. A value of zero indicates that a lossless transmission is requested. A value of 100 indicates a best effort transmission. The default value is 100.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>maxTrafficJitter</th>
<th>integer</th>
<th>The requested maximum jitter for end to end transmission with uint32 when traffic is packetized. The variation between packets arriving specified in nanoseconds with uint64.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>meters</th>
<th>ref[]</th>
<th>A list of references to Meters monitored for this VSP.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>eventLog</th>
<th>ref</th>
<th>A reference to the EventLog of this VSP.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-VSP+json

**JSON serialization:**
```
{ "self": string,
  "name": string,
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?,
  "network": [ "href": string ],
  "state": string, ?
  "bandwidthLimit": number, ?
  "trafficPriority": number, ?
```
5.12.11.1  Operations

This entity supports the Read, Update and Delete operations. Create is supported via the VSP Collection entity.

5.12.12  VSP (Virtual Switch Port) Collection

A VSP Collection entity represents the collection of VSPs within a Provider. This entity can be used to locate and create VSPs.

<table>
<thead>
<tr>
<th>Name</th>
<th>VSPCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/VSPCollection">http://www.dmtf.org/cimi/VSPCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vsps</td>
<td>ref[]</td>
<td>An array of references to the set of VSPs in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:
JSON media type: application/CIMI-VSPCollection+json

JSON serialization:

```json
{ "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "vspas": [
    { "href": string }, +
  ], ?
  "operations": [?
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string } ?
  ] ?,
...}
```

XML media type: application/CIMI-VSPCollection+xml

XML serialization:

```xml
<VSPCollection xmlns="http://www.dmtf.org/cimi">
  <self>xs:anyURI</self>
  <name>xs:string</name> ?
  <description>xs:string</description> ?
  <created>xs:string</created>
  <property name="xs:string">xs:string</property> *
  <vsp href="xs:anyURI"/> *
  <operation rel="add" href="xs:anyURI"/> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</VSPCollection>
```

5.12.12.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

Creating a New VSP

/link@rel: add

This operation will create a new VSP.

Input parameters: Either a reference to a VSP Template or a VSP Template itself.

Output parameters: A reference to a new VSP and, optionally, the representation of the VSP.

HTTP/REST Protocol

To create a new VSP a POST is sent to the "add" URI of the VSPCollection where the HTTP request body SHALL be as described below. Note this structure allows for certain properties to be passed in "by value" or by "reference". The definition of each property can be found in section 5.12.7.

JSON media type: application/CIMI-VSPCreate+json

JSON serialization:

```json
{ "name": string,
  "description": string, ?
...}
```
"properties": { "name": string, + }, ?
"vspTemplate": { "href": string, ?
   "properties": { "name": string, + } ?
"network": { "href": string }, ?
"vspConfig": { "href": string, ?
   "properties": { "name": string, + }, ?
"bandwidthReservation": number, ?
"trafficPriority": number, ?
"maxTrafficDelay": number, ?
"maxTrafficLoss": number, ?
"maxTrafficJitter": number ?,
} ?
}
...

XML media type: application/CIMI-VSPCreate+xml

XML serialization

<NSPCreate>
   <name> xs:string </name>
   <description> xs:string </description> ?
   <property name="xs:string"> xs:string </property> *
   <vspTemplate href="xs:anyURI"? >
      <property name="xs:string"> xs:string </property> *
      <network href="xs:anyURI" /> ?
      <vspConfig href="xs:anyURI">
         <property name="xs:string"> xs:string </property> *
         <bandwidthReservation> xs:integer </bandwidthReservation> ?
         <trafficPriority> xs:integer </trafficPriority> ?
         <maxTrafficDelay> xs:integer </maxTrafficDelay> ?
         <maxTrafficLoss> xs:integer </maxTrafficLoss> ?
         <maxTrafficJitter> xs:integer </maxTrafficJitter> ?
      </vspConfig> ?
   </vspTemplate>
</xs:any>*
</NSPCreate>

The serialization of some reference properties are specified such that a request MAY either include a reference ("href") to an existing entity or to include the entity "inline" as a set of additional properties. Requests SHALL NOT include both a reference and the inlined set of properties.

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the VSP entity.

5.12.13 Routing Group

A Routing Group represents a collection of Networks that route to each other..

<table>
<thead>
<tr>
<th>Name</th>
<th>RoutingGroup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/RoutingGroup">http://www.dmtf.org/cimi/RoutingGroup</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networks</td>
<td>ref[]</td>
<td>An array of references to the networks in this Routing Group.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:
5.12.13.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the RoutingGroup Collection entity.

5.12.14 Routing Group Collection

A Routing Group Collection entity represents the collection of Routing Groups within a Provider. This entity can be used to locate and create Routing Groups.

<table>
<thead>
<tr>
<th>Name</th>
<th>RoutingGroupCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/RoutingGroupCollection">http://www.dmtf.org/cimi/RoutingGroupCollection</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>routingGroups</td>
<td>ref[]</td>
</tr>
<tr>
<td>Properties: Optional/ Mutable</td>
<td></td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-RoutingGroupCollection+json

**JSON serialization:**

```json
{
    "self": string,
    "name": string,
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
    "networks": [
        { "href": string }, +
    ], ?
    "operations": [
        { "rel": "edit", "href": string }, ?
        { "rel": "delete", "href": string } ?
    ] ?
    ...}
```

**XML media type:** application/CIMI-RoutingGroupCollection+xml

**XML serialization:**

```xml
<VSP xmlns="http://www.dmtf.org/cimi">
    <self> xs:anyURI </self>
    <name> xs:string </name>
    <description> xs:string </description> ?
    <created> xs:string </created>
    <property name="xs:string"> xs:string </property> *
    <network href="xs:anyURI" > *
    <operation rel="edit" href="xs:anyURI"/> ?
    <operation rel="delete" href="xs:anyURI"/> ?
    <xs:any>*
</VSP>
```
5.12.14.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

Creating a New RoutingGroup

/link@rel: add

This operation will create a new RoutingGroup.

Input parameters: A RoutingGroup definition.

Output parameters: A reference to a new RoutingGroup and, optionally, the representation of the RoutingGroup.

HTTP/REST Protocol

To create a new RoutingGroup a POST is sent to the "add" URI of the RoutingGroupCollection where the HTTP request body SHALL be as described below. The definition of each property can be found in section 5.12.13.

JSON media type: application/CIMI-RoutingGroupCreate+json

JSON serialization:

```json
{  "name": string,  
"description": string, ?  
"properties": [  
"name": string, + ], ?  
"networks": [  
{  
"href": string }, +  
] ?
```
XML media type: application/CIMI-RoutingGroupCreate+xml

XML serialization

```
<RoutingGroupCreate>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <property name="xs:string"> xs:string </property> *
  <network href="xs:anyURI"/> *
  <xs:any>*
</RoutingGroupCreate>
```

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the RoutingGroup entity.

### 5.13 Monitoring Entities and Relationships

The following diagram illustrates the entities involved in tracking the progress of operations as well as metering and monitoring the status of other entities. Although this drawing is in the style of an Entity Relationship diagram, the use of UML is neither rigorous nor normative.
5.13.1 Job

This entity represents a process (i.e. a sequence of one or more operations directed to accomplish a specific goal) performed by the Provider.

If a Provider supports exposing Job entities to Consumers then each request from a Consumer that would result in a change to the environment MUST result in a Job entity being created and a reference to that Job entity MUST be made available to the requesting Consumer. Providers MAY create additional Job entities for Provider initiated operations.

<table>
<thead>
<tr>
<th>Name</th>
<th>Job</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/Job">http://www.dmtf.org/cimi/Job</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>targetEntity</td>
<td>ref</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------</td>
</tr>
<tr>
<td>affectedEntities</td>
<td>ref[]</td>
</tr>
<tr>
<td>action</td>
<td>URI</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
</tr>
<tr>
<td>progress</td>
<td>integer</td>
</tr>
<tr>
<td>statusMessage</td>
<td>string</td>
</tr>
<tr>
<td>timeOfStatusChange</td>
<td>DateTimeUTC</td>
</tr>
<tr>
<td>isCancellable</td>
<td>boolean</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-Job+json

**JSON serialization:**

```json
{  "self": string,
   "name": string,
   "description": string, ?
   "created": string, ?
   "properties": [ "name": string, + ], ?
   "targetEntity": [ "href": string ],
   "affectedEntities": [  
   { "href": string }, +
   ], ?
   "action": string,
   "status": string,
   "progress": integer,
   "statusMessage": string,
   "timeOfStatusChange": DateTimeUTC,
   "isCancellable": boolean
}
```
5.13.1.1 Operations

This entity supports the Read, Update and Delete operations.

5.13.2 Job Collection

A Job Collection entity represents the collection of Jobs within a Provider. This resource can be used to locate Jobs.

<table>
<thead>
<tr>
<th>Name</th>
<th>JobCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/JobCollection">http://www.dmtf.org/cimi/JobCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jobs</td>
<td>ref[]</td>
<td>An array of references to the set of Jobs in the Provider.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Optional / Mutable</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-JobCollection+json

```
{
  "self": string,
  "name": string,  
```

**XML media type**: application/CIMI-Job+xml

```
<Job xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <targetEntity href="xs:anyURI"/>
  <affectedEntity> xs:anyURI </affectedEntity> *
  <action> xs:anyURI </action>
  <status> xs:string </status>
  <progress> xs:integer </progress>
  <statusMessage> xs:string </statusMessage>
  <timeOfStatusChange> xs:dateTime </timeOfStatusChange>
  <isCancellable> xs:boolean </isCancellable>
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</Job>
```
5.13.2.1 Operations

This entity supports the Read and Update operations.

5.13.3 Meter Template

A Meter Template represents the definition of a Meter. A Meter Template can only be created by the Provider.

<table>
<thead>
<tr>
<th>Name</th>
<th>MeterTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MeterTemplate">http://www.dmtf.org/cimi/MeterTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>string</td>
<td>Name of the used units, e.g. kilobits per second, CPU usage percentage, etc. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>integer</td>
<td>It indicates the time between consecutive samples in seconds. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>associatedTo</td>
<td>URI[]</td>
<td>An array of URIs that indicate the entities to which a Meter created from this template can be applied. The value space of these URIs is identical to that of EntityMetadata.typeURI - a URI that uniquely identifies an entity type. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>timeScope</td>
<td>string</td>
<td>It indicates the time scope to which the Meter value applies.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>intervalDuration</td>
<td>string</td>
<td>It indicates the interval duration when the timeScope is set to “Interval”.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible values: hourly, daily, weekly, monthly or yearly. This attribute is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>read-only.</td>
</tr>
<tr>
<td>isContinuous</td>
<td>boolean</td>
<td>It indicates whether or not the Meter value is continuous or scalar.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Performance Meters are an example of a linear metric. This attribute is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>read-only.</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-MeterTemplate+json

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "units": string,
  "sampleInterval": integer,
  "associatedTo": [ ...

  { "href": string }, +
]

  "timeScope": string,
  "intervalDuration": string,
  "isContinuous": boolean,
  "operations": [ ...

  { "rel": "edit", "href": string }, ?

  { "rel": "delete", "href": string } ?

  ]

  ...
}
```

**XML media type:** application/CIMI-MeterTemplate+xml

```xml
<MeterTemplate xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <units> xs:string </units>
  <sampleInterval> xs:integer </sampleInterval>
  <associatedTo href="xs:anyURI"/> *
  <timeScope> xs:string </timeScope>
  <intervalDuration> xs:string </intervalDuration>
  <isContinuous> xs:boolean </isContinuous>
</MeterTemplate>
```
5.13.3.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Meter Template entity.

5.13.4 Meter Template Collection

A Meter Template Collection entity represents the collection of Meter Templates within a Provider. This entity can be used to locate Meter Templates.

<table>
<thead>
<tr>
<th>Name</th>
<th>MeterTemplateCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MeterTemplateCollection">http://www.dmtf.org/cimi/MeterTemplateCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meterTemplates</td>
<td>ref[]</td>
<td>An array of references to the set of Meter Templates in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-MeterTemplateCollection+json

**JSON serialization:**

```json
{
"self": string,
"name": string, ?
"description": string, ?
"created": string, ?
"properties": { "name": string, + }, ?
"meterTemplates": [
    { "href": string }, +
], ?
"operations": [
    { "rel": "add", "href": string } ?,
    { "rel": "edit", "href": string } ?
] ?
...
}
```

**XML media type:** application/CIMI-MeterTemplateCollection+xml

**XML serialization:**

```xml
<MeterTemplateCollection xmlns="http://www.dmtf.org/cimi">
    <self> xs: anyURI </self>
    <name> xs: string </name> ?
    <description> xs: string </description> ?
    <created> xs: string </created>
    <property name="xs: string"> xs: string </property> *
    <meterTemplate href="xs: anyURI"/> *
    <operation rel="add" href="xs: anyURI"/> ?
    <operation rel="edit" href="xs: anyURI"/> ?
    <xs: any>*
</MeterTemplateCollection>
```
5.13.4.1 Operations

This entity supports the Read and Update operations. Creation of new Meter Template entities is supported via a POST to the "addLink" URI as described in section 4.2.2.1.

5.13.5 Meter

This entity represents an available Meter of some property associated to a given entity.

<table>
<thead>
<tr>
<th>Name</th>
<th>Meter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/Meter">http://www.dmtf.org/cimi/Meter</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>string</td>
<td>Name of the used units, e.g. kilobits per second, CPU usage percentage, etc. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Mandatory / Mutable</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>integer</td>
<td>It indicates the time between consecutive samples in seconds. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Mandatory / Mutable</td>
</tr>
<tr>
<td>timeScope</td>
<td>string</td>
<td>It indicates the time scope to which this meter’s value applies.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Two possible values: &quot;Point&quot; indicates that the Meter applies to a point in time. &quot;Interval&quot; indicates that the Meter applies to a time interval. For instance, it would be possible to define a MeterTemplate which purpose is to provide the daily average CPU usage. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Mandatory / Mutable</td>
</tr>
<tr>
<td>intervalDuration</td>
<td>string</td>
<td>It indicates the interval duration when the timeScope is set to &quot;Interval&quot;. Possible values: hourly, daily, weekly, monthly or yearly. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Mandatory / Mutable</td>
</tr>
<tr>
<td>isContinuous</td>
<td>boolean</td>
<td>It indicates whether or not the Meter value is continuous or scalar. Performance Meters are an example of a linear metric. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Properties: Mandatory / Mutable</td>
</tr>
<tr>
<td>samples</td>
<td>sample[]</td>
<td>A list of taken samples</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Each sample attribute has the following sub-attributes:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>sample</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>timeStamp</td>
<td>DateTimeUTC</td>
<td>It indicates when the measure was taken (timeScope=&quot;Point&quot;). When the timeScope is &quot;Interval&quot;, it indicates the end of the time interval.</td>
</tr>
<tr>
<td><strong>Properties</strong>: Mandatory / Mutable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>value: string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>It indicates the sampled value of the measure.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Properties</strong>: Optional / Mutable</th>
</tr>
</thead>
<tbody>
<tr>
<td>minValue: string</td>
</tr>
<tr>
<td>It indicates the expected minimal measure value. This attribute is read-only.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Properties</strong>: Optional / Mutable</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxValue: string</td>
</tr>
<tr>
<td>It indicates the expected maximum measure value. This attribute is read-only.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Properties</strong>: Optional / Mutable</th>
</tr>
</thead>
<tbody>
<tr>
<td>stopTime: dateTimeUTC</td>
</tr>
<tr>
<td>It indicates a time from which the meter stops tracking samples. This attribute is writable.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Properties</strong>: Optional / Mutable</th>
</tr>
</thead>
<tbody>
<tr>
<td>expiresTime: dateTimeUTC</td>
</tr>
<tr>
<td>It indicates the time from which the Meter is not monitored anymore. It implies the deletion of the Meter after this time. This attribute is writable.</td>
</tr>
</tbody>
</table>

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-Meter+json

**JSON serialization**:

```json
{
    "self": string,
    "name": string, ?
    "description": string, ?
    "created": string, ?
    "properties": { "name": string, + }, ?
    "units": string,
    "sampleInterval": integer,
    "timeScope": string,
    "intervalDuration": string,
    "isContinuous": boolean,
    "samples": [ 
        { "timestamp": string, "value": string }, +
    ], ?,
    "minValue": string, ?
    "maxValue": string, ?
    "stopTime": string, ?
    "expiresTime": string, ?
    "operations": [ 
        { "rel": "edit", "href": string }, ?
        { "rel": "delete", "href": string } ?
    ]?
}
...```


**XML media type:** application/CIMI-Meter+xml

**XML serialization:**

```xml
<Meter xmlns="http://www.dmtf.org/cimi">
  <self xs:anyURI /></self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <units> xs:string </units>
  <sampleInterval> xs:integer </sampleInterval>
  <timeScope> xs:string </timeScope>
  <intervalDuration xs:string /></intervalDuration>
  <isContinuous> xs:boolean </isContinuous>
  <sample timestamp="xs:dateTime" value="xs:string"/> *
  <minValue> xs:string </minValue> ?
  <maxValue> xs:string </maxValue> ?
  <stopTime> xs:dateTime </stopTime> ?
  <startTime> xs:dateTime </startTime> ?
  <expiresTime> xs:dateTime </expiresTime> ?
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</Meter>
```

### 5.13.5.1 Operations

This entity supports the Read, Update and Delete operations. Create is supported via the Meter Collection entity.

The following custom operations are also defined:

#### Starting a Meter

`/link@rel: http://www.dmtf.org/cimi/action/start`

This operation will start a Meter.

**Input parameters:** None.

**Output parameters:** None.

Upon successful completion of this operation the Meter will begin to record samples related to its associated resource.

- **HTTP/REST Protocol**

To start a Meter a POST is sent to the "http://www.dmtf.org/cimi/start" URI of the Meter where the HTTP request body SHALL be as described below.

**JSON media type:** application/CIMI-Action+json

**JSON serialization:**

```json
{
  "action": "http://www.dmtf.org/cimi/action/start",
  "properties": { "name": string, + }?
  ...
}
```
5.13.6 Meter Collection

A Meter Collection entity represents the collection of Meters within a Provider. This entity can be used to locate and create Meters.

<table>
<thead>
<tr>
<th>Name</th>
<th>MeterCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/MeterCollection">http://www.dmtf.org/cimi/MeterCollection</a></td>
</tr>
</tbody>
</table>

Upon successful processing of the request, the HTTP response body will be empty.
The following describes the serialization of the entity in both JSON and XML:

### JSON media type: application/CIMI-MeterCollection+json

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "meters": [
    { "href": string }, +
  ], ?
  "operations": [?
    { "rel": "add", "href": string }, ?
    { "rel": "edit", "href": string }?
  ]?
...
}
```

### XML media type: application/CIMI-MeterCollection+xml

```xml
<MeterCollection xmlns="http://www.dmtf.org/cimi">
  <self>xs:anyURI</self>
  <name>xs:string</name>?
  <description>xs:string</description>?
  <created>xs:string</created>
  <property name="xs:string">xs:string</property> *
  <meter href="xs:anyURI"/>*
  <operation rel="add" href="xs:anyURI"/>?
  <operation rel="edit" href="xs:anyURI"/>?
  <xs:any>*
</MeterCollection>
```

### 5.13.6.1 Operations

This entity supports the Read and Update operations.

The following custom operations are also defined:

#### Creating a New Meter

/link@rel: add

This operation will create a new Meter.

Input parameters: Either a reference to a Meter Template or a Meter Template itself.

Output parameters: A reference to a new Meter and optionally the representation of the Meter.

- **HTTP/REST Protocol**

To create a new Meter a POST is sent to the "add" URI of the MeterCollection where the HTTP request body SHALL be as described below. Note this structure allows for certain properties to be passed in "by value" or by "reference". The definition of each property can be found in section 5.13.3.
JSON media type: application/CIMI-MeterCreate+json

```json
{
  "name": string,
  "description": string, ?
  "properties": { "name": string, + }, ?,
  "meterTemplate": { "href": string, ?
    "properties": { "name": string, + }, ?,
    "units": string, ?
    "sampleInterval": integer, ?
    "associatedTo": [ 
      { "href": string }, +
    ], ?,
    "timeScope": string, ?
    "intervalDuration": string, ?
    "isContinuous": boolean, ?
  }
}
```

XML media type: application/CIMI-MeterCreate+xml

```xml
<MeterCreate>
  <name> xs:string </name>
  <description> xs:string </description> ?
  <property name="xs:string"> xs:string </property> *
  <meterTemplate href="xs:anyURI"? >
    <property name="xs:string"> xs:string </property> *
    <units> xs:string </units> ?
    <sampleInterval> xs:integer </sampleInterval> ?
    <associatedTo href="xs:anyURI"/> *
    <timeScope> xs:string </timeScope> ?
    <intervalDuration> xs:string </intervalDuration> ?
    <isContinuous> xs:boolean </isContinuous> ?
  </meterTemplate>
  <xs:any>*
</MeterCreate>
```

The serialization of some reference properties are specified such that a request MAY either include a reference ("href") to an existing entity or to include the entity "inline" as a set of additional properties. Requests SHALL NOT include both a reference and the inline set of properties.

Upon successful processing of the request, the HTTP response body MAY either be empty or contain a serialization of the Meter entity.

### 5.13.7 Event Log

An entity that represents a registry of Events.

<table>
<thead>
<tr>
<th>Name</th>
<th>EventLog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/EventLog">http://www.dmtf.org/cimi/EventLog</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetEntity</td>
<td>ref</td>
<td>A reference to the entity to which the Events are related.</td>
</tr>
<tr>
<td>events</td>
<td>ref[]</td>
<td>A list of references to occurred Events.</td>
</tr>
</tbody>
</table>
**Properties:** Optional / Mutable

**persistence**  
string  
A value that indicates the persistence of the Events within the EventLog. For instance, daily, weekly, monthly or yearly.

**Properties:** Mandatory / Mutable

**summary**  
structure  
A summary of all the events present in the EventLog when the read operation is performed, grouped per severity.

Each summary attribute has the following sub-attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>low</td>
<td>integer</td>
<td>Number of occurred Events with a low severity.</td>
</tr>
<tr>
<td>medium</td>
<td>integer</td>
<td>Number of occurred Events with a medium severity.</td>
</tr>
<tr>
<td>high</td>
<td>integer</td>
<td>Number of occurred Events with a high severity.</td>
</tr>
<tr>
<td>critical</td>
<td>integer</td>
<td>Number of occurred Events with a critical severity.</td>
</tr>
</tbody>
</table>

**Properties:** Mandatory / Mutable

The following describes the serialization of the entity in both JSON and XML:

### JSON media type: application/CIMI-EventLog+json

```json
{  "self": string,  "name": string, ?,  "description": string, ?,  "created": string, ?,  "properties": { "name": string, + }, ?,  "targetEntity": { "href": string },  "events": [  { "href": string }, +  ], ?,  "persistence": string,  "summary": {  "low": number,  "medium": number,  "high": number,  "critical": number  }, ?,  "operations": [  { "rel": "edit", "href": string }, ?,  { "rel": "delete", "href": string }?  ] ?
...
```

Version 0.0.46a  
DMTF — Work In Progress - not a DMTF Standard  
111
XML media type: application/CIMI-EventLog+xml

XML serialization:

```xml
<EventLog xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <targetEntity href="xs:anyURI"/>
  <event href="xs:anyURI"/> *
  <persistence> xs:string </persistence>
  <summary>
    <low> xs:integer </low>
    <medium> xs:integer </medium>
    <high> xs:integer </high>
    <critical> xs:integer </critical>
  </summary>
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*</n>
</EventLog>
```

5.13.7.1 Operations

This entity supports the Read, Update and Delete operations.

5.13.8 Event Log Collection

A Event Log Collection entity represents the collection of Event Logs within a Provider. This resource can be used to locate EventLogs.

<table>
<thead>
<tr>
<th>Name</th>
<th>EventLogCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/EventLogCollection">http://www.dmtf.org/cimi/EventLogCollection</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventLogs</td>
<td>ref[]</td>
<td>An array of references to the set of Event Logs in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

JSON media type: application/CIMI-EventLogCollection+json

JSON serialization:

```json
{"self": string,
"name": string, ?
"description": string, ?
"created": string, ?
"properties": { "name": string, + }, ?
"eventLogs": [
  { "href": string }, +
], ?
"operations": [
  { "rel": "edit", "href": string } ?
] ?
```
Cloud Infrastructure Management Interface (CIMI) Model and REST Interface over HTTP

XML media type: application/CIMI-EventLogCollection+xml

XML serialization:

```xml
<EventLogCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <eventLog href="xs:anyURI"/> *
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</EventLogCollection>
```

5.13.8.1 Operations

This entity supports the Read and Update operations.

5.13.9 Event

An entity that represents the notification of an event within the managed infrastructure. Some examples of Events may be:

- Machine X has been rebooted by guest OS
- Machine X is not responding to platform services
- A new vCPU has been added to machine X following defined elasticity rules

The scope of the Event concept is any kind of information that the Provider is able to track within its infrastructure and that can constitute useful information for the consumer. Possible examples, but not limited to, are errors and inconveniences that occur in the (virtual) resources assigned to consumers, some provider initiated actions such as maintenance tasks, etc.

<table>
<thead>
<tr>
<th>Name</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/Event">http://www.dmtf.org/cimi/Event</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventTime</td>
<td>dateTimeUTC</td>
<td>The time and date of creation of the Event.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Immutable</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>A value that indicates the kind of Event (informational, error, alarm, etc.). This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>severity</td>
<td>string</td>
<td>A value indicating the Event severity. Possible values are: critical, high, medium or low. This attribute is read-only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Properties:</strong> Mandatory / Mutable</td>
</tr>
<tr>
<td>contact</td>
<td>string</td>
<td>An optional identifier that references a contact point to solve the problem (helpdesk, technical staff, etc.). This attribute is read-only.</td>
</tr>
</tbody>
</table>

Version 0.0.46a | DMTF — Work In Progress - not a DMTF Standard | 113
The following describes the serialization of the entity in both JSON and XML:

**JSON media type:** application/CIMI-Event+json

**JSON serialization:**
```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "eventTime": string,
  "type": string,
  "severity": string,
  "contact": string, ?
  "correlatedEvents": [ 
    { "href": string }, + 
  ], ?,
  "operations": [ 
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?,
  ] 
... }
```

**XML media type:** application/CIMI-Event+xml

**XML serialization:**
```xml
<Event xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <eventTime> xs:dateTime </eventTime>
  <type> xs:string </type>
  <severity> xs:string </severity>
  <contact> xs:string </contact> ?
  <correlatedEvent href="xs:anyURI"/> *
  <operation rel="edit" href="xs:anyURI"/> ?
  <operation rel="delete" href="xs:anyURI"/> ?
  <xs:any>*
</Event>
```

### 5.13.9.1 Operations

This entity supports the Read, Update and Delete operations.

### 5.13.10 Event Collection

An Event Collection entity represents the collection of Events within a Provider. This entity can be used to locate Events.
<table>
<thead>
<tr>
<th>Name</th>
<th>EventCollection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmtf.org/cimi/EventCollection">http://www.dmtf.org/cimi/EventCollection</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>events</td>
</tr>
<tr>
<td>Description</td>
<td>An array of references to the set of Events in the Provider.</td>
</tr>
</tbody>
</table>

Properties: Optional / Mutable

The following describes the serialization of the entity in both JSON and XML:

**JSON media type**: application/CIMI-EventCollection+json

**JSON serialization**:

```json
{
  "self": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "properties": { "name": string, + }, ?
  "events": [
    { "href": string }, +
  ], ?
  "operations": [
    { "rel": "edit", "href": string } ?
  ] ?
  ...
}
```

**XML media type**: application/CIMI-EventCollection+xml

**XML serialization**:

```xml
<EventCollection xmlns="http://www.dmtf.org/cimi">
  <self> xs:anyURI </self>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:string </created>
  <property name="xs:string"> xs:string </property> *
  <event href="xs:anyURI"/> *
  <operation rel="edit" href="xs:anyURI"/> ?
  <xs:any>*
</EventCollection>
```

### 5.13.10.1 Operations

This entity supports the Read and Update operations.

## 6 Scenarios

### 6.1 Initial Scenario

The following scenarios serves as an initial starting point:

A cloud consumer has a machine (e.g. a machine image plus some configuration information) that she wants to run on virtual resources provided by a cloud. The image for this machine already exists within the cloud infrastructure. Using that image, she requests the cloud to create a Machine instance. The request may specify configurable options, such as the number of virtual
This scenario is composed of the following use cases from the “Scoping Framework for Cloud Management Models and Protocol Requirements” document [ref needed]. Note that all statements to the effect of “he/she does foo” should be construed as expanding to “he/she causes some piece of consumer-side software to do foo”.

### 6.1.1 Create and deploy a Machine using a Provider created Machine Template

Steps to create a machine using an existing Machine Template:

1. Consumer makes a request to list all machine templates from the Provider.
2. The Provider gives the Consumer a list of machine templates that the Consumer has access to.
3. The Consumer browses the Machine Templates and gets details of each template. The details of the template contains the following information:
   - a. name of the template
   - b. a textual description of the template – this could include information such as operating system of the machine image that the template references
   - c. a unique reference to the template
   - d. the hardware profile or configuration referenced by the template such as # of CPUs, assigned disk capacity, and networks that the machine is configured to connect to
   - e. the name of the image that the template references - this could be the guest operating system and could include names of applications that may be built into the image by the Provider
4. Consumer selects the template of interest and makes a request to the Provider to create the machine by providing the following information
   - a. desired name of the machine
   - b. some description text for the machine
   - c. reference to the selected Machine Template
5. If the template indicates that the Consumer must provide specialization information such as administrator password, licensing keys etc, the Consumer supplies the information
6. The Provider accepts the request and provides a Job id to the Consumer. The Provider creates the Machine and deploys it to the Consumer’s cloud.
7. The Consumer can use the Job id to track progress of the machine creation.
8. Once the Job status shows Job as being completed successfully, the Consumer can start using the Machine. If the Consumer issues a Machine list query for his/her cloud, the created Machine is returned in the list.

### 6.1.2 Create a Machine by passing a Machine Template by value

Instead of creating and persisting a MachineTemplate, the Consumer may pass a MachineTemplate by value. The MachineTemplate may contain, for example:

- a. reference to a Machine Configuration
b. reference to a Machine Image  
c. reference to a Machine Admin entity  
d. reference to a Volume Template / Volume Configuration + Volume Image  
e. reference to a Network Template / Network Configuration  

6.1.3 Create a Machine using a Consumer created Machine Template  
Steps to create a Machine from a Machine Template that has been created by the Consumer:  
1. Create and save a Machine Template. See the scenarios for creating a Machine Template.  
2. Create a Machine by passing a reference to a Machine Template.  

6.1.4 Create a Machine Template by specifying individual components  
Consumer can create their own machine templates by browsing available resources. The steps are:  
1. Consumer chooses the components for the Machine Template, such as:  
   a. a Machine Configuration  
   b. a Machine Image  
   c. a Machine Admin  
   d. a Volume Template / Volume Configuration + Volume Image  
   e. a Network Template / Network Configuration  
2. Consumer makes a request to create a Machine Template by building a MachineTemplate that contains references to above chosen components. If a Consumer does not specify some components, the Provider can return an error or use default values.  
3. The Provider validates that required components are present and creates the Machine Template.  
   The Provider informs the Consumer of the location of the Machine Template. If the Consumer makes a request to browse Consumer created Machine Templates, this new template appears in the list.  

6.1.5 Create a Machine Template from a template file  
1. Consumer starts the create template process. He passes a reference to a file that contains metadata for creating a Machine Template.  
2. Provider accepts the request and provides a Job id to the Consumer.  
3. The Provider parses the template file, validates it and creates the template. If required components are not present, the Provider can return an error or use default values.  
4. The Provider informs the Consumer of the location of the template. If the Consumer makes a request to browse Consumer created Machine Templates, this new template appears in the list.  

6.1.6 Control Machine State (CMWG065)  
Our consumer controls the state of her Machine by updating the status attribute of the entity corresponding to that Machine. For example, updating the value to STOPPED would stop the Machine.
6.2 Machine Image Scenarios

6.2.1 Create new Machine Image from an image file

1. Consumer starts the create Machine Image process. She passes a reference to the image file (e.g. an OVF file).
2. Provider accepts the request, validates the request, creates the Machine Image and saves it to the Consumer’s cloud.
3. If the Consumer makes a request to browse Consumer created Machine Images, this new image appears in the list.

6.2.2 Create new Machine Image from Machine instance

1. Consumer selects a Machine instance from his Machine list
2. Consumer requests to make an image of the selected Machine instance by providing a name and a location to save the image to.
3. Provider accepts the request and creates the machine image and saves it to the location provided by the Consumer
4. If the Consumer makes a request to browse Consumer created machine images, this new image appears in the list

6.3 System Scenario

The following scenario serves to illustrate the creation of a System as a composite of Machines, Volumes, and Networks:

A cloud consumer has a system template (e.g., machine, machine image, network, storage) that it wants to run on virtual resources provided by a cloud. It will upload the template and request the cloud to deploy it. The request may specify configurable options, such as the number of virtual system targets. The provider will provision the required resources and deploy the template. The provisioned resources will be monitored and data made available to the cloud consumer. The consumer may start, modify, and stop the system. The consumer will terminate its use when it is complete.

6.3.1 List System Templates (CMWG010)

Since our cloud consumer is smart/lazy, the first thing she does is to verify that the System she wants to deploy does not already have an existing template in her target Site. By performing a Read operation on her target Site, our consumer obtains a list of references to the System Templates. Subsequent Read operations on the referenced System Templates allows our consumer to browse the available System Templates, and, via their name and description attributes, determine if her template is among them.

6.3.2 Create System Template

Having ascertained that a template of her desired System does not already exist within her target site, our consumer creates a System Template by posting a System Template to the System Template Collection.

6.3.3 Create and Deploy a System to a Site Using a System Template (CMWG017/CMWG035)

Once her template is created, our consumer creates her System via that entity’s Create operation using a reference to the recently created System Template. This action recursively creates Machines, Volumes,
and Networks (along with their interrelationships) for every Machine Template, Volume Template, and Network Templates contained by the System Template.

Our consumer monitors the progress of the Create operation by using the Read operation to poll state attribute of the newly created System and/or, optionally, using the Read operation to poll an associated Job entity.

6.3.4 Get Monitoring Information (CMWG066)

Our consumer monitors her System and its constituent Machines, Volumes, and Networks by Reading the appropriate entity.

6.3.5 Control System State (CMWG065)

Our consumer controls the state of her System by Updating the status attribute of the entity corresponding to that System. For example, updating the value to STOPPED would stop the System. This action recursively updates the status attributes of all Machines contained within her system, with corresponding effects on the operational state of those machines. Individual Machines can be controlled in a similar fashion.

6.3.6 Remove System from a Site (CMWG051)

When our consumer is finished with her System, she removes it by invoking the Delete operation on the entity corresponding to that System. This action recursively removes the Machines, Volumes, and Networks contained within her system. The removal of Volumes and Networks would obviously be contingent on whether or not these entities were shared by other, active Machines.

7 Security

This specification considers two separate but related security domains. The first domain, API-level security, concerns the protection of the entities modeled by this specification. For example, insuring that unauthorized users are not allowed to alter a Machine instance. The second domain, resource-level security, deals with the protection of the underlying resources represented by these entities. For example, insuring that unauthorized users cannot login to the Linux instance corresponding to that Machine.

7.1 API Level Security

7.1.1 Authentication

Except in cases where the access control policy allows for anonymous requests, the Provider SHALL authenticate all request messages and determine the identity of the Consumer. The techniques used to authenticate messages are outside the scope of this specification.

Protocol bindings of the CIMI Model specification are encouraged to include requirements for the most common authentication mechanisms applicable to that protocol (e.g. the use of BasicAuth for protocols using HTTP).

7.1.2 Message Integrity

Messages exchanged between the Consumer and the Provider SHOULD have message integrity protections applied. The mechanisms used to provide message integrity are outside the scope of this specification.

Protocol bindings of the CIMI Model specification are encouraged to include requirements for the most common integrity mechanisms applicable to that protocol (e.g. the use of TLS for protocols using HTTP).
7.1.3 Message Confidentiality

Messages exchanged between the Consumer and the Provider MAY have message confidentiality protections applied. The mechanisms used to provide message confidentiality are outside the scope of this specification.

7.1.4 Authorization

The Provider SHOULD process messages only if authorized by access control policy, which may reference the Consumer’s identity, the message type and content, and other contextual information when making this decision. The language in which this access control policy is expressed as well as the process by which these authorization decisions are made are outside the scope of this specification.

7.1.5 Multi-Tenancy

In cases where a Provider uses multi-tenancy to support a set of Consumers, the operations in this specification are modeled under the assumption that each Consumer's view of the system (i.e. which entities are visible, discoverable, and accessible) is scoped to those entities provisioned for or created by that Consumer. To the Consumer it appears that the Provider is implementing a sole-use instance of the CIMI API (albeit one who’s non-functional characteristics may be influenced the actions of invisible co-Consumers).

7.2 Resource Level Credentials

This specification intentionally avoids constraining the type, nature, or operation of the resources represented by the entities that it defines. It is therefore outside the scope of this specification to define the mechanism(s) used to access the resource represented by the Machine entity. There is, however, an integration point between this specification and such mechanisms, namely the management of the credentials (user names, passwords, keys, etc.) used to provision such access. This information is encapsulated by the Machine Admin entity (described in Section 0).
Bibliography


3750  DMTF DSP-IS0103, Distributed Management Task Force, Inc., *Use Cases and Interactions for Managing Clouds 1.0.0*, http://www.dmtf.org/sites/default/files/standards/documents/DSP-IS0103_1.0.0.pdf

3752  DMTF DSP-ISXXXX, Distributed Management Task Force, Inc., *Scoping Framework for Cloud Management Models and Protocol Requirements 0.1.5*,