Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol
An Interface for Managing Cloud Infrastructure
Copyright Notice

Copyright © 2012, 2013, 2016 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patent owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified after publication, and shall be indemnified and held harmless by any party implementing the standard from any and all claims of infringement by a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.

This document’s normative language is English. Translation into other languages is permitted.
CONTENTS

37  Foreword ......................................................................................................................... 7
38  1  Scope .......................................................................................................................... 11
39     1.1  Document structure ............................................................................................. 11
39     1.2  Document versioning scheme ................................................................................. 11
40     1.3  Typographical conventions ..................................................................................... 11
42  2  Normative references ................................................................................................... 12
43  3  Terms and definitions .................................................................................................. 14
44  4  HTTP-based protocol ................................................................................................... 17
45     4.1  Introduction .............................................................................................................. 17
46     4.1.1  Protocol evolution and client expectations ......................................................... 17
47     4.1.2  XML namespaces ............................................................................................... 17
48     4.1.3  URI space ............................................................................................................ 17
49     4.1.4  Media types .......................................................................................................... 18
50     4.1.5  Request headers ................................................................................................... 18
51     4.1.6  Request query parameters .................................................................................... 18
52     4.1.7  Response headers ............................................................................................... 24
53  4.2  Protocol operations ................................................................................................... 24
54     4.2.1  Common CRUD operations ............................................................................... 25
55     4.2.2  Error handling ..................................................................................................... 32
56  4.3  OVF support .............................................................................................................. 33
57  5  Model ............................................................................................................................ 33
58     5.1  Extensibility ............................................................................................................. 33
59     5.2  Identifiers ................................................................................................................ 34
60     5.3  Attribute constraints .............................................................................................. 34
61     5.4  Serialization of Resources ...................................................................................... 35
62     5.5  Data types and their serialization .......................................................................... 36
63      5.5.1  boolean ................................................................................................................. 36
64      5.5.2  dateTime ............................................................................................................... 36
65      5.5.3  duration ............................................................................................................... 36
66      5.5.4  integer .................................................................................................................. 36
67      5.5.5  string .................................................................................................................... 37
68      5.5.6  ref ........................................................................................................................ 37
69      5.5.7  map ....................................................................................................................... 38
70      5.5.8  structure .............................................................................................................. 38
71      5.5.9  byte[ ] .................................................................................................................. 39
72      5.5.10  URI ................................................................................................................... 39
73      5.5.11  Array .................................................................................................................. 39
74      5.5.12  Collection ......................................................................................................... 40
75      5.5.13  "Any" type ......................................................................................................... 46
76      5.5.14  valueScope ....................................................................................................... 46
77      5.5.15  Empty attribute values ...................................................................................... 49
78  5.6  Units ............................................................................................................................ 49
79  5.7  Resources ................................................................................................................... 49
80  5.7.1  Common Resource attributes ................................................................................. 49
81  5.8  Operations .................................................................................................................. 52
82  5.9  Alternative model formats ......................................................................................... 52
83  5.10  Relationships between Resources ......................................................................... 53
84  5.10.1  Referencing across Resources ............................................................................. 53
85  5.10.2  Composition relationship between Resources ...................................................... 53
86  5.11  Resource metadata ................................................................................................... 53
87  5.11.1  Capabilities ........................................................................................................... 60
88  5.11.2  ResourceMetadataCollection Resource ............................................................. 65
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

- 5.12 Cloud Entry Point
  - 5.12.1 Operations

- 5.13 System Resources and relationships
  - 5.13.1 System
  - 5.13.2 SystemCollection Resource
  - 5.13.3 SystemService Resource
  - 5.13.4 SystemTemplate Resource
  - 5.13.5 SystemTemplateCollection Resource
  - 5.13.6 Service-specific Descriptor attributes

- 5.14 Machine Resources and relationships
  - 5.14.1 Machine
  - 5.14.2 MachineCollection Resource
  - 5.14.3 MachineTemplate
  - 5.14.4 MachineTemplateCollection Resource
  - 5.14.5 MachineConfiguration Resource
  - 5.14.6 MachineConfigurationCollection Resource
  - 5.14.7 MachineImage Resource
  - 5.14.8 MachineImageCollection Resource
  - 5.14.9 Credential Resource
  - 5.14.10 CredentialCollection Resource
  - 5.14.11 CredentialTemplate Resource
  - 5.14.12 CredentialTemplateCollection Resource

- 5.15 Volume Resources and relationships
  - 5.15.1 Volume
  - 5.15.2 VolumeCollection Resource
  - 5.15.3 VolumeTemplate Resource
  - 5.15.4 VolumeTemplateCollection Resource
  - 5.15.5 VolumeConfiguration Resource
  - 5.15.6 VolumeConfigurationCollection Resource
  - 5.15.7 VolumeImage Resource
  - 5.15.8 VolumeImageCollection Resource

- 5.16 Network Resources and relationships
  - 5.16.1 Network
  - 5.16.2 NetworkCollection Resource
  - 5.16.3 NetworkTemplate Resource
  - 5.16.4 NetworkTemplateCollection Resource
  - 5.16.5 Segments
  - 5.16.6 ProtocolSegmentCollection Resource
  - 5.16.7 ProtocolSegmentTemplate Resource
  - 5.16.8 ProtocolSegmentTemplateCollection Resource
  - 5.16.9 Endpoints
  - 5.16.10 ProtocolEndpointCollection Resource
  - 5.16.11 ProtocolEndpointTemplate Resource
  - 5.16.12 ProtocolEndpointTemplateCollection Resource
  - 5.16.13 Interfaces
  - 5.16.14 NetworkInterfaceCollection Resource
  - 5.16.15 NetworkInterfaceTemplate Resource
  - 5.16.16 NetworkInterfaceTemplateCollection Resource
  - 5.16.17 Services
  - 5.16.18 NetworkServiceCollection Resource
  - 5.16.19 NetworkServiceTemplate Resource
  - 5.16.20 NetworkServiceTemplateCollection Resource

- 5.17 Monitoring Resources and relationships
  - 5.17.1 Job Resource
  - 5.17.2 JobCollection Resource
  - 5.17.3 Meter Resource
Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>XML namespaces</td>
<td>17</td>
</tr>
<tr>
<td>Table 2</td>
<td>Named structure</td>
<td>38</td>
</tr>
<tr>
<td>Table 3</td>
<td>Converting a relative URI to an absolute URI</td>
<td>39</td>
</tr>
<tr>
<td>Table 4</td>
<td>Numerical equivalents for attributes</td>
<td>49</td>
</tr>
<tr>
<td>Table 5</td>
<td>Common attributes</td>
<td>49</td>
</tr>
<tr>
<td>Table 6</td>
<td>ResourceMetadata attributes</td>
<td>54</td>
</tr>
<tr>
<td>Table 7</td>
<td>Capability URIs</td>
<td>61</td>
</tr>
<tr>
<td>Table 8</td>
<td>CloudEntryPoint attributes</td>
<td>66</td>
</tr>
<tr>
<td>Table 9</td>
<td>System attributes</td>
<td>72</td>
</tr>
<tr>
<td>Table 10</td>
<td>SystemService attributes</td>
<td>77</td>
</tr>
<tr>
<td>Table 11</td>
<td>SystemService attributes for HighReliability service</td>
<td>78</td>
</tr>
<tr>
<td>Table 12</td>
<td>RecoverableMachine accessory attributes</td>
<td>79</td>
</tr>
<tr>
<td>Table 13</td>
<td>SystemService attributes for DisasterRecovery service</td>
<td>81</td>
</tr>
<tr>
<td>Table 14</td>
<td>SystemTemplate attributes</td>
<td>82</td>
</tr>
<tr>
<td>Table 15</td>
<td>Additional parameters for HighReliability service</td>
<td>87</td>
</tr>
<tr>
<td>Table 16</td>
<td>Machine attributes</td>
<td>88</td>
</tr>
<tr>
<td>Table 17</td>
<td>Disk attributes</td>
<td>89</td>
</tr>
<tr>
<td>Table 18</td>
<td>locatedVolume accessory attributes</td>
<td>90</td>
</tr>
<tr>
<td>Table 19</td>
<td>MachineTemplate attributes</td>
<td>98</td>
</tr>
<tr>
<td>Table 20</td>
<td>MachineConfiguration attributes</td>
<td>101</td>
</tr>
<tr>
<td>Table 21</td>
<td>MachineImage attributes</td>
<td>102</td>
</tr>
<tr>
<td>Table 22</td>
<td>Credential attributes</td>
<td>104</td>
</tr>
<tr>
<td>Table 23</td>
<td>UserName/Password attributes</td>
<td>104</td>
</tr>
<tr>
<td>Table 24</td>
<td>Public key attributes</td>
<td>104</td>
</tr>
<tr>
<td>Table 25</td>
<td>CredentialTemplate attributes</td>
<td>105</td>
</tr>
<tr>
<td>Table 26</td>
<td>Volume attributes</td>
<td>106</td>
</tr>
<tr>
<td>Table 27</td>
<td>VolumeTemplate attributes</td>
<td>108</td>
</tr>
<tr>
<td>Table 28</td>
<td>VolumeConfiguration attributes</td>
<td>109</td>
</tr>
</tbody>
</table>
Table 29 – VolumeImage attributes.................................................................................................................... 110
Table 30 – Network attributes.............................................................................................................................. 112
Table 31 – NetworkTemplate attributes.............................................................................................................. 115
Table 32 – ProtocolSegment attributes.............................................................................................................. 117
Table 33 – IPv6 ProtocolSegment parameters.................................................................................................. 118
Table 34 – IPv4 ProtocolSegment parameters.................................................................................................. 118
Table 35 – Ethernet ProtocolSegment parameters.......................................................................................... 118
Table 36 – ProtocolSegmentTemplate attributes............................................................................................. 121
Table 37 – ProtocolEndpoint attributes.......................................................................................................... 122
Table 38 - IPv6 ProtocolEndpoint parameters................................................................................................. 123
Table 39 – IPv4 ProtocolEndpoint parameters................................................................................................. 123
Table 40 – Ethernet ProtocolEndpoint parameters.......................................................................................... 124
Table 41 – ProtocolEndpointTemplate attributes............................................................................................ 126
Table 42 – NetworkInterface attributes........................................................................................................ 128
Table 43 – NetworkInterfaceTemplate attributes.......................................................................................... 131
Table 44 – NetworkService attributes............................................................................................................ 133
Table 45 – NetworkServiceTemplate attributes............................................................................................. 137
Table 46 – Job attributes.................................................................................................................................... 138
Table 47 – Meter attributes................................................................................................................................ 141
Table 48 – Sample attributes.............................................................................................................................. 141
Table 49 – MeterTemplate attributes.............................................................................................................. 144
Table 50 – MeterConfiguration attributes..................................................................................................... 144
Table 51 – aspect URIs....................................................................................................................................... 145
Table 52 – EventLog attributes.......................................................................................................................... 146
Table 53 – EventLogTemplate attributes.......................................................................................................... 147
Table 54 – Event attributes................................................................................................................................. 148
Table 55 – type URIs........................................................................................................................................... 150
Foreword

The Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol specification (DSP0263) was prepared by the DMTF Cloud Management Working Group. It defines a logical model for the management of resources within the Infrastructure as a Service domain.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability.

Acknowledgments

The DMTF acknowledges the following individuals for their contributions to this document:

Editors (present and past):

- Jacques Durand – Fujitsu
- Marios Andreou – Red Hat (previous)
- Doug Davis – IBM (previous)
- Gilbert Pilz – Oracle (previous)

Contributors:

- Ghazanfar Ali – ZTE Corporation
- Marios Andreou – Red Hat
- Keith Bankston – Microsoft Corporation
- Winston Bumpus – VMware Inc.
- Nathan Burkhart – Microsoft Corporation
- Mark Carlson – Oracle
- Steve Carter – Novell
- Junsheng Chu – ZTE Corporation
- Josh Cohen – Microsoft Corporation
- Derek Coleman – Hewlett-Packard Company
- John Crandall – Brocade Communications Systems
- Doug Davis – IBM
- Jim Davis – WBEM Solutions
- Fernando de la Iglesia – Telefónica
- Hiroshi Dempo – NEC Corporation
- Jacques Durand – Fujitsu
- Yigal Edery – Microsoft Corporation
- George Ericson – EMC
- Colleen Evans – Microsoft Corporation
- Norbert Floeren – Ericsson AB
• Alan Sill – Texas Tech University
• Zhexuan Song – Huawei
• Marvin Waschke – CA Technologies
• Eric Wells – Hitachi, Ltd.
• Jeff Wheeler – Huawei
• Maarten Wiggers – Fujitsu
• Daniel Wilson – Ericsson AB
• Steve Winkler – SAP AG
• Jack Yu – Oracle
• Aaron Zhang – Huawei
• HengLiang Zhang – Huawei
Cloud Infrastructure Management Interface (CIMI) Model and
RESTful HTTP-based Protocol

1 Scope

This specification describes the model and protocol for management interactions between a cloud Infrastructure as a Service (IaaS) Provider and the Consumers of an IaaS service. The basic resources of IaaS (machines, storage, and networks) are modeled with the goal of providing Consumer management access to an implementation of IaaS and facilitating portability between cloud implementations that support the specification. This document specifies a Representational State Transfer (REST)-style protocol using HTTP. However, the underlying model is not specific to HTTP, and it is possible to map it to other protocols as well.

CIMI addresses the management of the life cycle of an infrastructure provided by a Provider. CIMI does not extend beyond infrastructure management to the control of the applications and services that the Consumer chooses to run on the infrastructure provided as a service by the Provider. Although CIMI may be to some extent applicable to other cloud service models, such as Platform as a Service (PaaS) or Storage as a Service (“SaaS”), these uses are outside the design goals of CIMI.

1.1 Document structure

This document defines a model and a RESTful HTTP-based protocol. The core REST patterns are defined first and, after each resource is defined, any HTTP-specific information for that resource is specified.

1.2 Document versioning scheme

This document adheres to the versioning scheme defined in clause 6.3 of DSP4014. As the specification changes over time certain features might be deprecated. These are identified in the specification and should not be supported. Each of these deprecated features is clearly denoted in the clause in which they were previously defined.

1.3 Typographical conventions

This specification uses the following conventions:

In the narrative text of the specification:

- The regular or narrative font is Arial.
- Proper CIMI nouns such as Resource names, attribute names, operation names, reserved variable names are in Courier font. (e.g., Machine, volumes, $expand). The plural form applies to such names to indicate several instances of such Resources (e.g., Machines, Systems).
- Example text is in small Courier font and over a darker background.
- Quotes are used for any text that needs be distinguished as a name or value of a particular concept (e.g., the “value constraints” attribute, the “Resource Name” column, a “false” value). In such cases, the string in quotes is always qualified by the concept it is an instance of.
- Names for CIMI concepts that may be common English words but have a very specific meaning in CIMI, are in narrative font but capitalized, e.g., Provider, Consumer, Resource, Collection.
When used in their common English sense they remain lowercase. However, CIMI modeling concepts that are used in a commonly understood manner remain in lowercase, such as: attribute, operation.

Inside tables describing the Resource data model:
- The narrative font is used for all terms, as the table structure qualifies them sufficiently.
- Where textual descriptions are introduced, the rules for narrative text apply.
- Names that are used as types (i.e., names of embedded structures as well as atomic types such as "integer", "string"), are in *italic*.
- Names that are just placeholders for actual names that may vary with each model instance are shown between < > (e.g., <componentTemplate>).

Where the serialization of Resources is described, a pseudo-schema notation is used with the following conventions:
- 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)
- Vertical bars, "|", denote choice. For example, "a|b" means a choice between "a" and "b".
- The characters {, }, [, and ] are block delimiters within the pseudo-schema. (Blocks may extend over multiple lines.)
- Parentheses, "(" and ")" are used in the pseudo-schema only to indicate the scope of the operators "?", "*", "+" and "|".
- 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.
- The scope of "?", "*", "+" and "[" follows these rules:
  - If immediately following a block delimiter or an array closing symbol e.g., "), ?" the scope is the entire block.
  - If not following any closing block delimiter, the scope is everything that precedes it on the same single line.

Operation names Create, Update, Delete, Read are abstract operations that convey the semantics of concrete corresponding operations, such as HTTP methods or CIMI operation URIs.

## 2 Normative references

The following referenced documents are indispensable for the application of this document. For dated or versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies. For references without a date or version, the latest published edition of the referenced document (including any corrigenda or DMTF update versions) applies.

DMTF DSP0223, *Generic Operations 1.0*,
http://www.dmtf.org/standards/published_documents/DSP0223_1.0.pdf
DSP0263  Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

380 DMTF DSP0243, *Open Virtualization Format Specification 1.1*,

382 DMTF DSP0262, *Cloud Audit Data Federation (CADF) -Data Format and Interface Definitions Specification version 1.0.0*,
383 http://dmtf.org/sites/default/files/standards/documents/DSP0262_1.0.0.pdf


388 DMTF DSP4014, *DMTF Release Process 2.3*,
389 http://www.dmtf.org/sites/default/files/standards/documents/DSP4014_2.3.0.pdf

390 IANA HTTP Header Registry,
391 http://www.iana.org/assignments/message-headers/perm-headers.html

393 http://www.iso.org/iso/catalogue_detail?csnumber=31898

396 http://standards.ieee.org/findstds/standard/802.3-2012.html

398 http://www.ietf.org/rfc/rfc791.txt

400 http://www.ietf.org/rfc/rfc2460.txt

403 http://www.ietf.org/rfc/rfc3986.txt

405 http://www.ietf.org/rfc/rfc4291.txt

406 IETF RFC4627, D. Crockford, *The application/json Media Type for JavaScript Object Notation (JSON)*, July 2006,
407 http://www.ietf.org/rfc/rfc4627.txt

410 http://www.ietf.org/rfc/rfc5246.txt

411 IETF RFC7230, R. Fielding et al, *Hypertext Transfer Protocol -(HTTP/1.1)*,
412 http://www.ietf.org/rfc/rfc7230.txt


417 http://www.iso.org/iso/catalogue_detail?csnumber=26153
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.

The terms defined in DSP4014, DSP0223 and DSP1001 apply to this document. The following additional terms are used in this document.

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 involve the use of username/password combination or public/private key pairs.
3.2 **authorization**
The process of verifying that an authenticated principal (person, service, etc.) has permission to perform certain operations (e.g., read, update) on specific Resources. (Also known as Access Control.)

3.3 **cloud**
Synonymous with "cloud computing" as defined in section 2 of the NIST Definition of Cloud Computing [SP800-145].

3.4 **Cloud Service Consumer**
(Or Cloud 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; purchases services within the business relationship; creates Service Users (including policies); allocates resources, such as computer and storage; generates reports, such as usage; etc.); and Service Users (who use service instances provided by a Cloud Service Provider). The term "Consumer" is used if 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 is used.

For purposes of comparison and alignment, it should be noted that a Cloud Service Consumer is equivalent to the "Cloud Consumer" actor defined in the NIST Reference Architecture [SP500-292].

3.5 **Cloud Service Provider**
(Or Cloud 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 if the indicated action or activity could involve one or more of the above actors. In cases where the distinction between the actors in the category is relevant, the more detailed term is used.

For purposes of comparison and alignment, it should be noted that a Cloud Service Provider is equivalent to the "Cloud Provider" actor defined in the NIST Reference Architecture [SP500-292].

3.6 **Collection**
A particular kind of Resource that contains a collection of other Resources and has a representation and serialization defined in this specification. Synonym for “CIMI collection”.

3.7 **Configuration**
A set of metadata, the values of which serve as the parameters of a discrete conformation of a specific type of virtual resource.
Endpoint
An element within a Network Segment from which communication can originate or to which communication can be sent. Endpoints have a unique, protocol specific, address within a Segment by which they are distinguished.

Infrastructure as a Service (IaaS)
A cloud computing service model defined in section 2 of the NIST Definition of Cloud Computing [SP800-145].

Interface
An abstract element of virtual hardware that enables connection to a Network via Endpoints.

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

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

Network
A construct that supports communications between elements within a Cloud using one or more protocol specific Segments that support addressable Endpoints.

Resource
A representation of an entity managed by the [Cloud Service] Provider that is generally available to the [Cloud Service] Consumer to access or operate on by way of the interface described in this specification. Synonym for “CIMI resource”.

Segment
A component of a Network that supports communication between Endpoints using a single protocol. Also referred to as a Protocol Segment to emphasize that Segments are always bound to a single communication protocol.
4 HTTP-based protocol

4.1 Introduction

All operations are based on the HyperText Transfer Protocol (HTTP), version 1.1 [RFC7230]. Each request is sent by 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, whose semantics are interpreted in the context of the particular request that was made. Each Resource in the model has a MIME type that further contextualizes the payload of requests and responses.

Resources in the model are identified by URIs, and each Resource's representation shall contain an "ID" attribute, of type URI, that acts as a "self-pointer." This URI shall be unique within the context of the Provider's implementation. Dereferencing (through an HTTP GET) the URI of a Resource yields a representation of the Resource containing attributes and links to associated Resources. To begin operations, a client shall know the URI to the main entry point of a Provider - also known as the "Cloud Entry Point" Resource. All other Resources within the environment shall then be discoverable by way of the iterative following of links to associated Resources within each Resource retrieved.

4.1.1 Protocol evolution and client expectations

Future versions of this specification structure changes in such a way that clients who conform to an earlier version of this specification continue to work, and are not adversely affected by the evolution of the protocol. Clients are expected to follow a few simple rules to ensure this compatibility:

1. Clients shall not assume that the serializations shown for responses in this specification are complete. In particular, clients shall accept responses that contain data mixed in with the serializations shown here, and shall ignore such data. However, per clause 4.2.1.3, clients shall include unknown data in PUT requests to update Resources.

2. Clients shall not assume anything about the operations supported by a server. They are expected to discover operations that are supported (and permissible) by navigating to Resources from the Cloud Entry Point. The serializations of Resources encountered indicate which operations are supported by the server.

4.1.2 XML namespaces

Table 1 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://schemas.dmtf.org/cimi/2">http://schemas.dmtf.org/cimi/2</a></td>
<td>This specification</td>
</tr>
<tr>
<td>xs</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
<td>XML Schema Part2</td>
</tr>
</tbody>
</table>

4.1.3 URI space

While URIs returned by Providers are to be treated as opaque by Consumers, and Consumers shall not make assumptions about the layout of the URIs or the structures of the URIs for the Resources, a Consumer may augment URIs with any well-defined query parameters that are supported by the Provider as defined in clause 4.1.6.
The sample URIs used in this specification are not normative and the patterns used shall not be interpreted as guidance for implementations. For example, any of the following URIs might be used by Providers to reference a particular Machine Resource:

- `http://example.com/machines/12345`
- `http://example.com/machines?id=12345`
- `http://example.com/12345`
- `http://example.com/Cloud/resource?id=12345`

### 4.1.4 Media types

In this specification, Resource and response representations are encoded either in JSON, as specified in [RFC4627](https://tools.ietf.org/html/rfc4627) or in XML. If serialized in JSON, the media-type for CIMI resources shall be "application/json". If serialized in XML, the media-type shall be "application/xml".

In the JSON serialization of CIMI representations sent by Providers, there shall be an additional attribute on the root object called "resourceURI" that contains the unique URI that is associated with the type of CIMI resource being serialized. Note that this requirement applies even if the `$select` attribute is used to subset the Resource being acted upon.

In the XML serialization of Collection representations sent by Providers, there shall be a `resourceURI` attribute, as shown in the example XML serialization of Collections in clause 5.5.12. This attribute is optional for Consumers to include. If included, this attribute's value shall match the "typeURI" attribute of the corresponding ResourceMetadata Resource (see clause 5.8), if ResourceMetadata is supported. This value shall also be equivalent to the wrapping element of the XML serialization; in other words, the namespace of the wrapper element concatenated a "/" and then its localName.

Any CIMI resource implemented by a Provider shall have representations in JSON and XML. The client implementation may thus use either JSON or XML in requests with any server implementation, and may request a specific serialization using server-driven content negotiation (using the Accept request header).

### 4.1.5 Request headers

This specification uses general-header, request-header, and entity-header headers as defined in [RFC7230](https://tools.ietf.org/html/rfc7230) in request messages to provide metadata about the message. Applications using messages defined in this specification shall use headers consistent with the requirements of [RFC7230](https://tools.ietf.org/html/rfc7230).

### 4.1.6 Request query parameters

Providers may choose to include query parameters as part of the URIs returned to Consumers. Consumers shall include those query parameters when sending messages to those URIs. CIMI-defined query parameters are prefixed with a dollar sign ("$"). If Providers choose to define query parameters, they shall not be prefixed with a dollar sign to avoid conflicts with current and future CIMI-defined query parameters.

To modify the behavior of the Provider when processing request messages, Consumers may augment request URIs as described in the following clauses. As stated in clause 4.1.3, URIs returned from Providers are to be treated as opaque by Consumers; however, it is the responsibility of the Consumer to understand the use of the query parameters defined in the following clauses and ensure correctness when making a request.
Unsupported, or unknown, query parameters shall be silently ignored by Providers. Consumers may examine the CloudEntryPoint's capabilities to determine whether support of these query parameters is enabled.

### 4.1.6.1 Filtering Collections

If retrieving the representation of a Collection, Consumers may include the `$filter` query parameter to reduce the number of entries of the Collection that are returned based on the data within the entries of the Collection. Providers shall interpret and process the `$filter` query parameter as described in this clause. The `$filter` parameter shall be of the form:

```
?$filter=expression
```

Where "expression" represents a mathematical expression denoting how the top-level attributes of the Resources within the Collection shall be filtered. The expression is defined by the following EBNF grammar:

```
Filter ::= AndExpr ( 'or' Filter )* ; 
AndExpr ::= Comp ( 'and' AndExpr )* 
Comp ::= Argument Op Value 
   | Value Op Attribute 
   | PropExpr 
   | '(' Filter ')' 
Op ::= '<' | '<=' | '=' | '>=' | '>' | '!=' 
Argument ::= ? resource attribute name ? 
Value ::= IntValue | DateValue | StringValue | BoolValue 
IntValue ::= /[0-9]+/ 
DateValue ::= ? as defined by XML Schema ? 
StringValue ::= "..." | '...' 
BoolValue ::= 'true' | 'false' 
PropExpr ::= 'property[' StringValue ']' Op StringValue
```

Where `PropExpr` is used to find Resources that contain a property with a certain key-value combination. The key is the `StringValue` within the square brackets (`[]`) and the value is the `StringValue` after the `Op`. The Resource shall be considered to satisfy the search criteria if any of the properties in the Resources match the specified `PropExpr`.

Each of these shall be percent encoded in the URL as appropriate.

The choice of which operator (including 'and' and 'or') is limited based on the type of the value and attribute. The following example describes the allowable operators:

```
'or', 'and' : Boolean value/attribute
'<', '<=', '>=', '>', '!' : Integer and date value/attribute
'=', '!' : String value/attribute
```

Consumers may include multiple filters within a single URI. Providers shall treat multiple filters as a series of "and" expressions where an entry of the Collection shall only be included in the response message if it satisfies all of the filter expressions specified.
Examples:

In the following examples, the following sample base URIs are used.

The URI to the MachineCollection of the Cloud Entry Point is as follows:

/machines

The URI to a Machine is as follows:

/machines/123

The URI to the DiskCollection of a Machine is as follows:

/machines/123/disks

The URI to the VolumeCollection of a Machine is as follows:

/machines/123/volumes

To filter the MachineCollection so that just Machines with a "name" attribute of "mine" are returned, use the following filter:

GET /machines?$filter=name='mine'

To filter a DiskCollection of a Machine so that just Disks with a format of "ntfs" are returned, the following filter would be used:

GET /machines/123/disks?$filter=format='ntfs'

If the $filter parameter is used, the Collection's "count" attribute shall contain the number of Resources matching the filter expression.

4.1.6.2 Subsetting Collections

If retrieving the representation of a Collection, Consumers may include query parameters to subset the number of entities of the Collection that are returned. Providers shall interpret and process these query parameters as described in this clause. While the previous clause discussed how to perform a filter over the data within the Collection, this clause uses ordinal position within the Collection to achieve the desired reduction.

This specification defined two query parameters that, if used, shall indicate the first and last ordinal positions of the entities within the Collection that are returned. The query parameters shall be of the form:

?$first=number

?$last=number

Where "$first" indicates the (1-based) ordinal position of the first entity of the Collection to return and "$last" indicates the (1-based) ordinal position of the last entity of the Collection to return. Consumers are not required to use both at the same time. If $first is specified but $last is not, the implied value for $last shall be the ordinal position of the last entity in the Collection. Conversely, if $last is specified but $first is not, the implied value for $first shall be 1.

If Consumers include these query parameters, the ordinal positions of entries in the collection before subsetting shall be stable when no changes are made to the collection or its entries. If filtering or sorting are used in the same query, the subsetting applies to the collection resulting from those operations.

If any part of the range as expressed by $first and $last is outside of the bounds of the Collection, just the Resources (if any) in the Collection that are contained within that range shall be returned. A fault shall not be generated if any part, or all, of the expressed range is outside the bounds of the Collection.
Note that if $first$ is larger than $last$, the range shall represent an empty range and therefore no 
Resources are returned.

If either $first$ or $last$ are specified, and a filter expression (as defined in clause 4.1.6.1) is also 
specified, the filter expression shall be performed first and then the ordinal constraints of $first$ and 
$last$ shall be applied.

The inclusion of $first$ or $last$ does not affect the value of the Collection’s returned "count" attribute: 
it shall contain the number of Resources in the Collection before subsetting. In case filtering is also used, 
"count" shall be the size of the Collection resulting from the filtering.

### 4.1.6.3 Subsetting Resources

If retrieving the representation of a Resource, Consumers may include the $select$ query parameter to 
specify a subset of the Resource to be acted upon. Providers shall interpret and process this query 
parameter as described in this clause. This subsetting shall have the semantic equivalence of referencing 
a different Resource whose attributes are a subset of the original Resource as specified by the attribute 
names listed in the $select$ query parameter. The format of a $select$ query parameter is:

```
?$select=attributeName,...
```

The value of the $select$ query parameter shall be a comma-separated list of top-level attribute names 
of the Resource, possibly including the string “operations” in case the intent is to select the operations 
available to the Consumer for this Resource. Any attribute name erroneously appearing in the list that is 
not part of the Resource shall be ignored by the Provider. An attribute name of "*" is equivalent to 
specifying all of the attributes of the Resource including its operations. Any attribute name explicitly 
appearing more than once in a URI shall have its second (and subsequent) appearances ignored.

The $select$ query parameter may appear more than once in a URI. This is semantically equivalent to 
all of the attribute names appearing as values of a single $select$ query parameter. For example:

```
?$select=name&$select=state
```

is equivalent to:

```
?$select=name,state
```

The order of attribute names in the $select$ query parameter is not relevant for serialization purposes. 
The attributes are serialized per the serialization rules/order as specified by the Resource definition.

Note that per clause 4.1.4, if a Resource representation is sent by a Provider it shall always include the 
resourceURI attribute even if it is not specified in the $select$ query parameter.

For example, to subset the list of Machine attributes being acted upon to just the "name" and 
"description", the following query parameter would be used:

```
?$select=name,description
```

See clause 4.2.1.3.1 for more information about the impact of using this query parameter when updating 
a Resource.

If $select$ is used in the URI for a Collection resource, the subsettings shall apply to the attributes of the 
Collection resource itself as for any other Resource. For example, to subset a Collection resource to only 
return the number of its items, plus the operations available on this Collection:

```
?$select=count,operations
```

However, exceptionally for Collection resources, if some attribute provided in the $select$ list is not a 
top-level attribute of the Collection resource but instead is an attribute of the entities that are items of the
Collection, the subsetting shall apply to each item of the Collection regarding this attribute. For example, if retrieving the DiskCollection, the following query parameter:

```
?$select=name,capacity
```

returns a collection of the Disks associated with a Machine but each entity of the collection just has the name and capacity attributes and nothing else, not even the operations or id attributes.

Optionally, an implementation may also support the alternative attribute name notation:

```
<collectionName>/<attributeName>
```

for subsetting the items inside a collection. For example, the following subsetting on items of a Disks Collection is equivalent to the one done in the previous example, while in addition listing the operations of the Collection resource itself (not of its items):

```
?$select=disks/name,disks/capacity,operations
```

This notation, if supported (see the "QueryPathNotation" capability in 5.11.1), allows for disambiguating subsettings if the same attribute name can be found for the Collection and for each item in the collection (which is always the case for id and operations).

### 4.1.6.4 Expanding references

If retrieving the representation of a Resource, Consumers may include the `$expand` query parameter to specify which of the top-level "reference" attributes of the Resource shall be "expanded". Providers shall interpret and process this query parameter as described in this clause. To expand a reference means that the attributes of the Resource being referenced shall be included in the serialization of that attribute. This feature allows for a more optimized retrieval of Resources.

The serialization shall be performed as follows:

**JSON serialization:**

```
"name": { "href": string }
```

shall be expanded to be:

```
"name": {
    "href": string,
    ... attributes of referenced resource...
}
```

**XML serialization:**

```
<name href="xs:anyURI"/>
```

shall be expanded to be:

```
<name href="xs:anyURI">
    ... attributes of the referenced resource...
</name>
```

Note that in the XML case the nested elements shall not contain the wrapper element of the referenced Resource (e.g., `<Machine>` in the case of a reference to a Machine Resource).

The format of a `$expand` query parameter shall be:

```
?$expand=attributeName,...
```

The value of the `$expand` query parameter is a comma-separated list of attribute names. Any attribute name erroneously appearing in the list that is not part of the Resource, or is not a reference, shall be ignored by the Provider. An attribute name of "+", or no attribute name list at all, is equivalent to specifying
all of the attributes. Any attribute name explicitly appearing more than once in a URI shall have its second (and subsequent) appearances ignored.

The $expand query parameter may appear more than once in a URI, which is semantically equivalent to all of the attribute names appearing as values of a single $expand query parameter.

If the Resource being retrieved is a Collection, the attribute names listed in the $expand shall apply to the attributes of the entities within the Collection. For example, specifying:

```
?$expand=volumes
```

if retrieving the MachineCollection has the same net effect as applying the "expand" semantics to the specified attribute ("volumes" in this example) of each Machine within the Collection. To be clear, $expand acts on the attributes of the Resources in the Collection, not on the wrapping Collection Resource itself.

### 4.1.6.5 Specifying the Resource format

If retrieving the representation of a Resource, the HTTP Accept header is used to specify the encoding style of the response. While it is recommended that Consumers use the Accept header, there might be situations where Consumers are unable to control the values specified in that header. In these cases Consumers may use the $format query parameter to override the Accept header values. Providers shall interpret and process the $format query parameter as described in this clause.

The $format parameter shall be of the form:

```
?$format=encoding
```

Where "encoding" is the requested representation of the response. This specification defines the following possible values: "json" and "xml". Providers may support others. The value of the $format query parameter shall not be case sensitive.

If both an Accept header and $format query parameter are present in a request message, the $format value shall take precedence. If the $format query parameter appears more than once, the second, and subsequent, appearances shall be ignored.

### 4.1.6.6 Sorting Collections

If retrieving the representation of a Collection, Consumers may include the $orderby query parameter to sort the entries of the Collection that are returned based on different attributes or in a different order (descending). Providers shall interpret and process the $orderby query parameter as described in this clause. The $orderby parameter shall be of the form:

```
?$orderby=attributeName[:asc|:desc], ...
```

The $orderby expression may include multiple, comma-separated attribute names. Each attribute name may be optionally followed immediately by a colon and "asc" to denote ascending order (default), or "desc" to denote descending order for that attribute. If neither asc nor desc is specified, the order shall be "ascending".

The attributes included in the $orderby shall be of the following types as defined in clause 5.5: boolean, dateTime, duration, integer, or string.

The sort shall be performed based on the attribute type.
The following rules apply to the ascending sort order:

- boolean – ‘false’ shall come before ‘true’.
- dateTime – An earlier datetime shall come before a later datetime.
- duration – A shorter duration shall come before a longer duration.
- integer – Smaller integers shall come before larger integers. Negative integers shall come before positive integers.
- string – Ordering is based on a binary comparison of the transformed strings according to the rules of the Normalization Form KD of the Unicode standard as defined in Unicode Standard Annex (UAX), annex #15.

For the desc sort order, the reverse of the above shall be performed.

**Examples:**

To sort the result set of the MachinesCollection Resource on the “created” attribute in descending order, the following expression would be used:

```plaintext
GET /machines?$orderby=created:desc
```

To sort the result set of the MachinesCollection Resource on the “cpu” attribute in descending order, followed by the “memory” attribute in ascending order, the following expression would be used:

```plaintext
GET /machines?$orderby=cpu:desc,memory:asc
```

If collection subsetting is used in the same query, the subsetting applies to the sorted collection. When no $orderby is specified, the order of entries in the returned Collection is not defined.

### 4.1.7 Response headers

As defined in [RFC7230](https://tools.ietf.org/html/rfc7230), this specification uses general-header, response-header, and entity-header headers in response messages to provide metadata about the message. Applications that use messages defined in this specification shall use headers consistent with the IANA HTTP Header Registry.

#### 4.1.7.1 Job header

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

```plaintext
CIMI-Job-URI = "CIMI-Job-URI" "::*" string
```

#### 4.1.7.2 ETag support

An ETag header may be provided by a Provider with each Resource as specified in [RFC7230](https://tools.ietf.org/html/rfc7230). If a Provider does provide an ETag header, it shall also support If-Match header processing on behalf of the Consumer.

### 4.2 Protocol operations

This clause defines the set of common HTTP operations that a Provider may 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 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 clause that defines the Resource.

If appropriate, some of the Resource representations include an "operations" attribute. Providers shall only include the "operations" attribute if the specified operations are accessible to the current client for that particular Resource. This situation means that based on many factors (e.g., authorization rights of the clients, current state of the Resource, etc.), a different set of "operations" shall be returned on each serialization of the Resource.

Each operation shall include a "rel" and an "href" field. The "rel" field shall uniquely identify the operation name (e.g., "add", "edit"), while the "href" field is the URI to which the operation's request message shall be sent. Note that the "href" field's URI may be different from the URI of the Resource itself. Each operation may have an "available" field to indicate that the operation can be performed by the Consumer. The "available" field is of type boolean with a default value of "true". If "available" is set to "false", it indicates that the operation is not currently available. This would normally indicate a temporary condition.

For example, some Machine operations may not be available depending on the state of the Machine.

The operations attribute shall be serialized as follows:

**JSON serialization:**
```
{ "operations": [
    { "rel": string, "href": string, ("available": boolean)? }, +
  ]
}
```

**XML serialization:**
```
<operations xmlns="http://schemas.dmtf.org/cimi/2">
  <operation rel="xs:anyURI" href="xs:anyURI" (available="xs:boolean")? /> *
</operations>
```

For example, the "edit" operation would appear as:

**JSON serialization:**
```
{ "operations": [
    { "rel": "edit", "href": "<editURI>" }
  ]
}
```

**XML serialization:**
```
<operations xmlns="http://schemas.dmtf.org/cimi/2">
  <operation rel="edit" href="<editURI>"/>
</operations>
```

Additional "rel" values may be defined by Providers; however, they shall be fully qualified URIs and not relative URIs.

### 4.2.1 Common CRUD operations

Each of the Resources supported by this protocol shall adhere to the interaction patterns defined in the following clauses.
4.2.1.1 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 includes an "add" operation. The "add" operation references the addURI that is to be used.

The HTTP POST request shall include:

- CIMI serialization of the request to create a new Resource in the HTTP Body
- HTTP Content-Type header
- HTTP Content-Length header

For example, the request can be:

```
POST <addURI> HTTP/1.1
Host: <hostname>
Accept: application/(json|xml)
Content-Type: application/(json|xml)
Content-Length: <length>

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

This example has an Accept header with one of the CIMI supported media types: application/json or application/xml. If the Provider chooses to reply with a serialization, this serialization should be of the specified media type. Omission of the Accept header allows the Provider to reply with a serialization of any media type. If the Resource has a "State" attribute, its value shall be "CREATING" while the Provider is processing this operation.

Many of the create requests are defined such that a Template of the new Resource is passed. These create requests allow for the Template to be passed in "by-reference" or "by-value." For example, creating a new Machine looks like this (here using XML):

```
<MACHINECREATE xmlns="http://schemas.dmtf.org/cimi/2">
  <NAME> xs:string </NAME> ?
  <DESCRIPTION> xs:string </DESCRIPTION> ?
  <PROPERTIES>
    <PROPERTY key="xs:string"> xs:string </PROPERTY> *
  </PROPERTIES>
  <MACHINETEMPLATE href="xs:anyURI"? >
    ... template attributes ... ?
  </MACHINETEMPLATE>
</MACHINECREATE>
```

Note that in the XML case the creation of a new Machine requires a wrapper element named MACHINECREATE per the rules specified in clause 5.5.12.1.

More generally, creating a new Resource shall follow one of these two serialization patterns (here illustrated in JSON):
(1) Resource creation by passing a Template by value:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/ResourceCreate",
  "name": "myResourceName",
  "description": "My resource description",
  "properties": { "prop1name": "prop1value", + },
  "resourceTemplate": {
    <here the template is passed by value>
  }
}
```

Where `resourceTemplate` is the actual name of the template for that Resource.

(2) Resource creation by passing a Template by reference:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/ResourceCreate",
  "name": "myResourceName",
  "description": "My resource description",
  "properties": { "prop1name": "prop1value", + },
  "resourceTemplate": { "href": string,
    <here some template attribute/value pairs may be added to override values in the referenced template>
  }
}
```

In case the created Resource is itself a Template, only the first creation pattern - by value - applies.

In both patterns (1) and (2) the `resourceURI` attribute specifies the operation here generically identified as "ResourceCreate", e.g., MachineCreate.

In both patterns (1) and (2) an element corresponding to the Resource Template (here identified generically as “resourceTemplate” e.g., MachineTemplate) is specifying the Template to be used, either by value (1) or by reference (2).

Direct setting of attributes in the new Resource:

In a creation request it is possible to set the value of some attributes of the newly created Resource, regardless of what values the Template instantiation might have set if used alone. Three common attributes of the newly created Resource may be set: `name`, `description`, and `properties`.

The semantics shall be same as of a partial update of the Resource for these attributes (described in the next subclause), immediately following the Resource creation from the Template alone.

Defining or referring to the Resource Template:

In pattern (1) above, the Provider may choose to create a Template Resource from the value given, but such creation is temporary in nature. The Provider shall not expose such a transient Resource to the Consumer and no such transient Resource shall be included in any query results back to the Consumer.
In pattern (2) above, additional attribute name/value pairs may be given inside the ResourceTemplate element that also contains the reference to the external (pre-existing) Template to override similar attributes defined in the Template. More precisely:

- Any top-level attribute of complex or simple type in the referred Template shall be overridden by providing its name/value pair in the create request inside the resourceTemplate element and immediately under it. For a top-level attribute of a complex type (e.g., arrays, Collections, structures), the provided complex value shall also set all underlying attributes — e.g., array elements.
- The semantics shall be same as of modifying (overriding) parts of the referred Template just before it is used for instantiation, but these overrides shall not persist in the referred Template and shall only concern this particular instantiation.

In pattern (2) above, Consumers may erase any Template attributes by specifying either "attribute": null for the attribute in the JSON serialization, or <attribute/> in the XML serialization for that attribute.

Some of the create requests allow for configuration type of Resources to be passed by-reference or by-value as well - e.g., Credential on a Machine create operation. The processing rules defined above apply in those cases as well.

If the response has a 201 status code, the response shall include:

- HTTP Location header with a reference to the new Resource

If the response to a create request includes a serialization of the new Resource, the response shall additionally include:

- HTTP Content-Type header
- HTTP Content-Length header

For example, the response can be:

```
HTTP/1.1 201 Created
Location: <location>
Content-Type: application/(json|xml)
Content-Length: <length>
<serialization of new resource>
```

4.2.1.2 Retrieving a representation of a Resource

To retrieve a representation of Resource, an HTTP GET request is sent to the Resource's URI.

For example, the request can be:

```
GET <ResourceURI> HTTP/1.1
Host: <hostname>
Accept: application/(json|xml)
```
If the response has a 200 status code, the response shall include:

- HTTP Content-Type header
- HTTP Content-Length header

For example, the response can be:

```plaintext
HTTP/1.1 200 OK
Content-Type: application/(json|xml)
Content-Length: <length>
<serialization of resource>
```

### 4.2.1.3 Updating a Resource

To update a Resource's state, an HTTP PUT request containing the complete, updated representation is sent to a designated editURI for that Resource type. Consumers shall include all nonempty attributes of the Resource in the PUT request - including ones that it might not support or understand that were returned in a GET response. This is to ensure that a client does not inadvertently modify (erase) data in a Resource by excluding it from the full representation of the Resource.

In many cases, this editURI is the same as the URI of Resource itself. Retrieving the Resource representation shall include an "edit" operation, 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, it shall silently ignore that attribute update request and shall not generate an error. This rule applies to Resource partial updates as well.

Because of potential conflicts that might occur due to multiple concurrent updates, Consumers should use the partial update mechanism, defined in 4.2.1.3.1, to reduce the chances of mistakenly updating attributes with out-of-date data.

The HTTP PUT request shall include:

- CIMI serialization of the updated Resource in the HTTP Body
- HTTP Content-Type header
- HTTP Content-Length header

For example, the request can be:

```plaintext
PUT <editURI> HTTP/1.1
Host: <hostname>
Accept: application/(json|xml)
Content-Type: application/(json|xml)
Content-Length: <length>

<serialization of request to update a resource>
```
If the response includes a serialization of the updated Resource and has a status code of 200, this response shall include:

- HTTP Content-Type header
- HTTP Content-Length header

For example, the response can be:

```plaintext
HTTP/1.1 200 OK
Content-Type: application/(json|xml)
Content-Length: <length>

<serialization of updated resource>
```

### 4.2.1.3.1 Partial updates to a Resource

For clarity, this clause explains how to use the `$select` query parameter (see clause 4.1.6.3) to subset a Resource for the purposes of only operating on a selected set of top-level attributes.

To update only certain top-level attributes of a Resource, a Consumer may include only the altered attributes in the representation of the Resource within the HTTP request body. If this request is made, 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 shall be of the form:

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

Only the attributes listed in the URI's query parameters shall be modified; attributes not listed in the URI shall not be directly modified by the request. Note that this circumstance 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 shall be 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 changed.

Adhering to the generic PUT semantics defined previously, any attribute of the original (full) Resource included within the HTTP request body shall result in an error being generated if that attribute is not listed in the `$select` query parameter - see clause 5.3. Note that this is due to these attributes being unknown to this subsetted Resource.
The following sample request updates just the name and description attributes of a Machine:

```
PUT /machines/myMachine?$select=name,description HTTP/1.1
Host: <hostname>
Accept: application/xml
Content-Type: application/xml
Content-Length: <length>

<Machine>
  <name>My New Machine</name>
</Machine>
```

The name attribute is set to "My New Machine" and the description attribute is erased.

### 4.2.1.4 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 is the same as the URI of Resource itself. Retrieving the Resource representation shall include a "delete" operation, which contains the deleteURI that is to be used, if the requester is allowed to delete the Resource.

For example, the request can be:

```
DELETE <deleteURI> HTTP/1.1
Host: <hostname>
```

If the Resource has a State attribute, its value shall be "DELETING", while the Provider is processing this operation.

For example, the response can be:

```
HTTP/1.1 200 OK
```

### 4.2.1.5 Other operations

While some modifications to the Resources in the model can be done by way of a simple update (PUT) operation to the Resource's editURI, sometimes a more complex set of actions needs to be taken. In these cases, the operations shall be modeled as HTTP POSTs to the operation specific URI of the Resource.

For each of the Resources that define additional operations, a description of the HTTP request and response bodies is provided. However, the general HTTP interactions are as described below.

The request shall be of the following form:

```
POST <operationURI> HTTP/1.1
Host: <hostname>
Accept: application/(json|xml)
Content-Type: application/(json|xml)
Content-Length: <length>

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

The form of the response varies depending on the operation and is defined by the operation itself.
Note that the definition of the Create operation (see clause 4.2.1.1) follows this same pattern. It is just called out for ease of reference.

### 4.2.1.6 Synchronous operations

If a Provider supports the Job Resource, each incoming PUT, DELETE, POST request shall result in a Job Resource being created and an absolute URI reference to that Job Resource shall be returned back to the client by way of the CIMI-Job-URI HTTP Header in the HTTP response message:

```plaintext
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, the server shall return a 202 and follow the instructions for Asynchronous operations.

### 4.2.1.7 Asynchronous operations

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

As with synchronous operations, if a Provider supports the Job Resource, it shall create a Job Resource for the incoming request and return a reference to that Job Resource back to the client by way of the CIMI-Job-URI HTTP Header in the HTTP response message. Additionally, in the case of a "202 Accepted" response code, the Provider may also return any of the following in the HTTP response body:

- A representation of the Job Resource, if one was created.
- A partial representation of the response message as if the operation were a synchronous operation. For example, when creating a new Machine, the response message may include a partial representation of the new Machine in the response message. The list of attributes of the Resource that is returned is implementation specific and based upon how much information is available at the time the response message is generated, but it shall be consistent with the definition of the full Resource representation. In the case of a create operation, the Provider may also include an HTTP Location header referencing the "to be created" Resource, if it is known.
- An empty response body.

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

### 4.2.2 Error handling

In cases where an error occurs during the processing of a request, the Provider shall include a representation of a Job Resource describing the status of the failed operation. This representation of a Job shall be included even in cases where the Provider does not expose Job Resources. This is to ensure that Consumers are provided with sufficient information, in a consistent manner, as to the reason for the failure. A transient Job Resource may be created by the Provider just for error reporting. In case a Job Resource is not intended to be used for more than error reporting, the returned "id" attribute shall be an empty path (i.e., "") and the nestedJobs array shall be expanded (see 4.1.6.4) to inline the representation of the pseudo subordinate Jobs.
4.3 OVF support

The Open Virtualization Format (OVF) Specification (DSP0243) describes an open, secure, portable, efficient, and extensible format for the packaging and distribution of software to be run in virtual machines. OVF support in CIMI allows an OVF package to be used to create CIMI management resources by importing the package. Additionally, CIMI management resources can be exported into an OVF package. The actual support for the OVF package is typically provided by a hypervisor that is managed by the CIMI provider. The import of an OVF package exposes CIMI specific constructs and parameters as a result of the import without altering the original OVF package. Thus the CIMI resources that are created as a result of the import form a “View” of what the hypervisor did; however, other (non-CIMI mapped) information from the OVF package may have been used by the hypervisor in its import. This other information is implementation dependent and is not further touched upon by this standard.

An OVF package can support single virtual machines (VMs) corresponding to a single CIMI Machine or MachineTemplate (see clause 5.14.1) or may also support a complex hierarchy of VMs and their related Resources corresponding to a CIMI System or SystemTemplate (see clause 5.13.1) and related CIMI management resources.

OVF support is covered in more detail in ANNEX A.

5 Model

This model assumes that a business relationship has already been established between the Consumer and the 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. The scope of this model is one separately administered cloud.

The CIMI model is described here by using a tabular representation. Each table is modeling a significant cloud resource for which independent access and manipulation is expected. Relationships between resources use a referential mechanism based on unique identifiers that is expected to be already supported by the implementation environment and protocol (e.g., URIs for HTTP).

The model is self-describing and allows for querying its own metadata, e.g., to discover which extensions have been implemented. The model is also extensible in different ways (see clause 5.1).

5.1 Extensibility

There are two types of extensibility mechanisms defined by the CIMI model; one is intended for use by Consumers while the other is to be used by Providers.

The first allows for a CIMI Consumer to add additional data to a Resource. Each Resource in the CIMI model has an attribute called "properties". Consumers, when creating or updating a Resource, may store any name/value pair in the properties attribute. CIMI Providers shall store and return these values to the Consumer. There is no obligation for the Provider to understand or take any action based on these values; they are there for the Consumer's convenience. Providers shall not add elements to this properties attribute.
The second type of extensibility mechanism allows for Provider defined extensions and this specification includes the ResourceMetadata Resource for this purpose. ResourceMetadata may be used to

- express constraints on the existing CIMI defined Resource attributes (e.g., express a maximum for the 'cpu' attribute of the MachineConfiguration Resource)
- introduce new attributes for CIMI defined Resources together with any constraints governing these (e.g., a new 'location' attribute for the Volume Resource that takes values from a defined set of strings)
- introduce new operations for any of the CIMI defined Resources (e.g., define a new 'compress' operation for the Volume Resource)
- express any Provider specific capabilities or features (e.g., the length of time that a Job Resource is retained after Job completion and before this is deleted)

It is recommended that Providers use the ResourceMetadata Resource to advertise these attributes, operations, and capabilities along with any constraints that might need to be understood by Consumers. The ResourceMetadata Resource is defined in clause 5.8.

5.2 Identifiers

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

- Identifier names shall be treated as case sensitive.
- Identifier names shall only use the following set of characters:
  - Uppercase ASCII (U+0041 through U+005A)
  - Lowercase ASCII (U+061 through U+007A)
  - Digits (U+0030 through U+0039)
  - Underscore (U+005F)
- Identifier names shall not start with a Digit (U+0030 through U+0039).

Note that these rules do not apply to the "name" common attribute defined in clause 5.7.1.

5.3 Attribute constraints

Each attribute of any Resource is further qualified by a set of Boolean constraints. In particular, these constraints govern the level of support and access for an attribute, for either the Provider or the Consumer. Such constraints may be explicitly stated in the model itself for some Resources (i.e., determined by this specification), but in general are specified in the metadata Resource associated with a Resource (i.e., configured in the implementation). These constraints are:

providerMandatory: (true/false)

If 'true', indicates that the attribute shall be supported by the Provider, i.e., always included as part of the Resource representation sent from Providers to Consumers, except if the attribute is empty. See clause 5.5.15 regarding empty attribute values. If present on a nested attribute, this attribute is required to be supported only if the parent attribute is supported. Default is 'true'.

consumerMandatory: (true/false)

If ‘true’, indicates the attribute shall always be supported by the Consumer when using such a Resource, i.e., included as part of the Resource representation sent from Consumers to Providers, except if the attribute is empty. See clause 5.5.15 regarding empty attribute values. If present on a nested attribute, this attribute is required to be supported only if the parent attribute is supported. Default is ‘false’.

mutable: (true/false)

If ‘true’, indicates that the attribute may be modified after initial creation of the Resource. If ‘false’, the attribute value will never change until the Resource is deleted. When the constrained attribute is a reference to another Resource, mutable = ‘false’ only means this reference will never change. It does not prevent updates on the referenced resource itself. Note that mutable = ‘false’ also implies consumerWritable = ‘false’. Default is ‘true’.

customerWritable: (true/false)

If ‘true’ – and if mutable is also ‘true’ - indicates that the attribute may be directly set or updated by Consumers (update request), after creation of the Resource. Note that some Consumer operations on the Resource may have the indirect effect of changing some attribute values (this is obvious for the updated attribute, for example, or for the state of a Resource), but these are not considered as “direct” updates. Consequently such indirect updates are not precluded by consumerWritable = ‘false’. Also, when the constrained attribute is a reference to another Resource, consumerWritable = ‘false’ only means this reference cannot be changed by the Consumer. It does not prevent updates on the referenced resource itself. Default is ‘true’.

Additional requirements for Provider and Consumer:

- If a Provider receives a message containing an unknown or unsupported attribute, it shall reject the request.
- If a Consumer receives a message containing an unknown or unsupported attribute, it shall silently ignore the attribute. However, Consumers are required to include those attributes in messages sent back to the Provider. Note in these cases the Consumer is not required to understand or process the unsupported attribute, but merely echo it back to the Provider.

5.4 Serialization of Resources

The serialization of Resource instances in the model follow these conventions. Consider the serialization of a Resource named “MyResource”:

JSON serialization:

The Resource is serialized as an object wrapping all its attributes, but without a wrapper name. The Resource includes a resourceURI with a URI for the type of Resource being serialized. For example:

```json
{  "resourceURI": "http://example.com/MyResource",
  "attribute": "value"
}
```

XML serialization:

The Resource is serialized as an element with name equal to the Resource name; for example:

```xml
<MyResource xmlns="http://example.com">
  <attribute> value </attribute>
</MyResource>
```
The serialization of attributes in a Resource follows the rules for the serialization of each data type, listed in clause 5.5.

5.5 Data types and their serialization

Unless specifically asked to not include certain attributes in the Resource representation, the absence of an optional attribute in the representation means that the attribute has no value (i.e., is undefined), meaning there is no notion of an optional attribute having an implied value. Note that a client cannot distinguish (from just looking at the returned representation) whether a particular attribute is not supported from one that does not exist. Likewise, an absent attribute from a Resource representation as the input to an update operation means that the Consumer is requesting that the Provider remove that attribute.

The following clauses describe the data types and values that are used within the model definition tables.

5.5.1 boolean

A value as defined by xs:boolean per XML Schema – Part 2, with the exception that the only allowable values are either "true" or "false." The value is case sensitive.

If serialized in JSON, these values shall be of JSON type: boolean
If serialized in XML, these values shall be of XML Schema type: xs:boolean

5.5.2 dateTime

A value as defined by xs:dateTime per XML Schema – Part 2, which is consistent with DSP4014 and ISO 8601. The timestamp should preserve time zone information, i.e., include a local time component and an offset from UTC.

Any constraints on the specific ranges allowed for any particular attribute are specified by that attribute's definition or at runtime by the Provider by way of the metadata discovery mechanisms defined by this specification.

For example, Monday, May 25, 2012, at 1:30:15 PM EST is represented as:

2012-05-25T13:30:15-05:00

If serialized in JSON, these values shall be of JSON type: string
If serialized in XML, these values shall be of XML Schema type: xs:dateTime

5.5.3 duration

A value as defined by xs:duration per XML Schema – Part 2. Any constraints on the specific ranges allowed for any particular attribute shall be specified by that attribute's definition or at runtime by the Provider by way of the metadata discovery mechanisms defined by this specification.

If serialized in JSON, these values shall be of JSON type: string
If serialized in XML, these values shall be of XML Schema type: xs:duration

5.5.4 integer

A value as defined by xs:integer per XML Schema – Part 2. Any constraints on the specific ranges allowed for any particular attribute shall be specified by that attribute's definition or at runtime by the Provider by way of the metadata discovery mechanisms defined by this specification.

If serialized in JSON, these values shall be of JSON type: number
If serialized in XML, these values shall be of XML Schema type: \textit{xs:integer}.

5.5.5 \textbf{string}

A value as defined by \textit{xs:string} per \textit{XML Schema -- Part 2}. Any constraints on this type for any particular attribute shall be specified by that attribute's definition or at runtime by the Provider by way of the metadata discovery mechanisms defined by this specification.

If serialized in JSON, these values shall be of JSON type: \textit{string}.

If serialized in XML, these values shall be of XML Schema type: \textit{xs:string}.

If serializing a string attribute of type string, the serialization shall omit this attribute in case of an empty string.

5.5.6 \textbf{ref}

A reference to another Resource.

References allow for Consumers to navigate to Resources. By starting at the Cloud Entry Point – the main entry point Resource for a Cloud Consumer that is associated with a Cloud Provider - and following the references that appear in the retrieved Resources, Consumers are able to recursively discover and navigate to all other Resources.

As a general rule, if an attribute is of type "ref", its value shall be held by an attribute named "href" (both in JSON and XML).

**JSON serialization:**

In the JSON serialization the \textit{href} property appears as of type "string." If an attribute is of type "ref", the name of this attribute shall appear as a key, with the \textit{href} property as a nested value. For example, a Resource attribute "myvolume" of type "ref" is serialized as:

\begin{verbatim}
"myvolume": { "href": string }
\end{verbatim}

**XML serialization:**

In the XML serialization the \textit{href} attribute appears as type "\textit{xs:anyURI}". If an attribute is of type "ref," the name of this attribute shall appear as name of an XML element with the \textit{href} property as an (XML) attribute. For example, a Resource attribute "myvolume" of type "ref" is serialized as:

\begin{verbatim}
<myvolume href="xs:anyURI"/>
\end{verbatim}

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 convenience, the JSON and XML representations, as shown above, exclude the implicit extensibility points that would allow for the attributes of the target Resource to be included if desired. So, more accurately the above representations might be written as follows:

For JSON:

\begin{verbatim}
"myvolume": { "href": string, ... }
\end{verbatim}

and in XML:

\begin{verbatim}
<myvolume href="xs:anyURI"> xs:any </myvolume>
\end{verbatim}

However, for brevity the extensibility points are excluded from the serialization of the Resources.
5.5.7 map

A list of key-value pairs. The same "key" shall not be used more than once within an attribute. The "key" is case sensitive.

If serializing an attribute of type map, the serialization shall omit this attribute in case of an empty map.

5.5.8 structure

Attributes of this type are complex attributes made up of a set of nested attributes. For each attribute of this type, there is an additional table defining those nested attributes.

A nested structure can be considered a complex type definition. Structures may be named or unnamed.

Table 2 is an example of named structure:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>low</td>
<td>number</td>
<td>Number of “low” occurrences</td>
</tr>
<tr>
<td>medium</td>
<td>number</td>
<td>Number of “medium” occurrences</td>
</tr>
<tr>
<td>high</td>
<td>number</td>
<td>Number of “high” occurrences</td>
</tr>
<tr>
<td>critical</td>
<td>number</td>
<td>Number of “critical” occurrences</td>
</tr>
</tbody>
</table>

JSON serialization:

In JSON, the name of the structure (i.e., of the type it represents) never appears. In other words, whether the structure is named or not does not matter. An attribute named “systemIncidents” of type “summary” (as above) is serialized as follows:

```json
"systemIncidents": {
    "low": number,
    "medium": number,
    "high": number,
    "critical": number
}
```

XML serialization:

In XML, the name of the structure (i.e., of the type it represents) never appears. In other words, whether the structure is named or not does not matter. The same previous "systemIncidents" example is serialized so that the structure sub-attributes become XML attributes of a `<systemIncidents>` XML element wrapper:

```xml
<systemIncidents low="xs:integer" medium="xs:integer" high="xs:integer"
    critical="xs:integer"/>
```

NOTE A large number of sub-attributes of atomic type in a structure may be represented alternatively as XML child elements for better readability. Both options are available; however, the same structure shall be serialized the same way across Resources.
5.5.9 byte

An arbitrary set of bytes meant to represent a block of binary data. Any constraints on this type for any particular attribute shall be specified by that attribute's definition or at runtime by the Provider by way of the metadata discovery mechanisms defined by this specification.

If serialized in JSON, these values shall be of JSON type: string

If serialized in XML, these values shall be of XML Schema type: xs:hexBinary

5.5.10 URI

The format and syntax of the attributes of type "URI" is defined by RFC3986.

Unless otherwise noted, this specification does not mandate whether Providers use relative or absolute URI in the HTTP response bodies.

If URIs are specified as relative URIs, they shall be relative to the baseURI.

The algorithm used for converting a relative URI to an absolute URI shall be as described in section 5.2 of RFC3986. Table 3 illustrates how relative URIs are resolved against base URIs:

<table>
<thead>
<tr>
<th>Base URI</th>
<th>Relative URI</th>
<th>Absolute URI</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://example.com/">http://example.com/</a></td>
<td>p1/file</td>
<td><a href="http://example.com/p1/file">http://example.com/p1/file</a></td>
</tr>
</tbody>
</table>

If relative URIs are used, the baseURI shall end with a trailing slash and relative URIs shall not begin with a leading slash. This format is consistent with most URI resolve utilities and produces the same results as a simple string concatenation algorithm.

If serialized in JSON, these values shall be of JSON type: string

If serialized in XML, these values shall be of XML Schema type: xs:anyURI

5.5.11 Array

An array represents an ordered list of items of the same type. An array shall always appear as an attribute of a Resource, and is only accessible as such (it is not a separately addressable Resource). If a Resource is deleted, the items in its arrays shall also be deleted. However, in case these items were just references to other Resources, these referred Resources are not affected. (See the semantics of references in 5.7.)

Attributes that are arrays are defined by using the notation itemType[], where itemType is the type name for each item of the array. If the type is a structure, not a simple data type, it is recommended as a convention in the model that the name of an array be the plural of a name that characterizes each item.

For example, an array of volume items or of references to these may be named "volumes."
JSON serialization:

Within this specification, arrays in JSON are serialized with a wrapper property. The wrapper name shall be same as the attribute name for the array. For example, a "things" attribute of type "thing[]" is serialized as:

```
"things" : [ 
  { ... }, + 
] ?
```

If the items in the array are structures, the structure name shall not be present in the JSON serialization.

In the case of an array of references, i.e., where the "ref" type applies to each element of the array, each element shall simply be serialized as an href property within a JSON array. For example, an array "things" of type "ref[]" is serialized as:

```
"things": [ 
  { "href": string }, + 
] ?
```

XML serialization:

The XML serialization of arrays requires each item of the array to be represented as an element. These elements shall be consecutive and contiguous in the serialization and the name of each element (tag name) shall be the name of the element type (the name that appears before "[]" in the array type). As in JSON, the serialized array has a wrapper element of same name as the array attribute name. For example, a "things" attribute shall be serialized as a list of items named "thing":

```
<things>
  <thing>
    ...
  </thing> *
</things>
```

In the case of an array of references, i.e., where the "ref" type applies to each element of the array, the array is serialized as a list of XML elements without wrapper. Each element is named per an array "item name" specified in the attribute's definition. For example, an array "things" of type "ref[]" where the array "item name" is "thing" is serialized as:

```
<thing href="xs:anyURI"/> +
```

5.5.12 Collection

A Collection is a group of Resources of the same type. In contrast with arrays, Collections are themselves Resources that have their own URI and can be independently accessed. Collections also allow for an optimized and convenient interaction pattern by providing a specialized set of operations that avoid replacing a large number of items when updating the set, as with arrays.
This specification uses Collections if the set of grouped items is modified often and potentially by multiple Consumers. Conversely, arrays are used if it is expected that the list of items is not modified often or can be easily modified by substitution of the entire list, and thus the overhead of managing these items as separate Resources might be unjustified and burdensome.

An item in a Collection, i.e., a Collection item, is an embedded structure that contains a reference to a Resource and optionally additional attributes (see “accessory” attributes, defined later). For convenience, the Resource referred to by a Collection item is called here a Resource item of the Collection.

A Resource may be referenced by more than one Collection. If such a Resource is deleted, every Collection that references this Resource shall remove the corresponding item. While different Collections contain entries of different Resource types, all Collections follow the pattern described below:

- A Collection shall contain an id attribute that acts as a "self-pointer." Retrieving the data at this reference shall return the Collection. In the XML representation, each Collection shall be wrapped by a <Collection> element.

- A Collection shall contain a count attribute that indicates the number of Resources in the Collection at the time the Collection was queried.

- Adding new Resources to the Collection shall be done either via the "add" operation defined within the Collection (when the Resource is also created) or via the "insert" operation (when the Resource already exists).

Deleting an item from the Collection shall be done either via a "delete" operation on the Resource item itself (if the Resource has to be discarded) or via the "remove" Collection operation (if the Resource must still exist outside the Collection). Collections that are attributes of other Resources are represented with attribute type "collection[itemType]." The Resource type of the Collection items are specified inside the brackets; for example an attribute that is a Collection of Machines is expressed as "collection[Machine]." Attributes of such types are serialized as a reference to a Collection Resource instead of holding the Collection itself as value. For brevity, while these attributes are "references" the word "ref" or "reference" does not appear in the model definition tables - instead the type of such an attribute is making abstraction of the reference and more explicitly shows as "collection[itemType]."

In the serializations below, the Collection items are represented by items in the ResourceSpecificGroupingName JSON array, and by ResourceSpecificElementName elements in the XML representation.

Serialization:

The serialization of Collections shall adhere to the following pattern:

**JSON serialization:**

```json
{
  "resourceURI": string,
  "id": string,
  "updated": string, ?
  "parent": string, ?
  "count": number,
  "resourceSpecificGroupingName": [
    {
      "resourceURI": string,
      "id": string,
      "name": string, ?
    }
  ]
}```
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

```
"description": string, ?
"created": string, ?
"updated": string, ?
"parent": string, ?
"properties": { string: string, + }, ?
... resource specific data ...
"operations": [
    { "rel": "edit", "href": string }, ?
    { "rel": "delete", "href": string } ?
] ?
... 
"operations": [
    { "rel": "add", "href": string } ?
    { "rel": "insert", "href": string } ?
    { "rel": "remove", "href": string } ?
]
... 

XML serialization:

```
<Collection resourceURI="xs:anyURI" xmlns="http://schemas.dmtf.org/cimi/2">
    <id> xs:anyURI </id>
    <updated> xs:dateTime </updated> ?
    <parent> xs:anyURI </parent> ?
    <count> xs:integer </count>
    <resourceSpecificGroupingName>
        <ResourceSpecificElementName>
            <id> xs:anyURI </id>
            <name> xs:string </name> ?
            <description> xs:string </description> ?
            <created> xs:dateTime </created> ?
            <updated> xs:dateTime </updated> ?
            <parent> xs:anyURI </parent> ?
            <property key="xs:string"> xs:string </property> *
            ... resource specific data ...
            <operations>
                <operation rel="edit" href="xs:anyURI"/> ?
                <operation rel="delete" href="xs:anyURI"/> ?
            </operations>
        </ResourceSpecificElementName>
    </resourceSpecificGroupingName>
</Collection>
```
Where the `resourceURI` attributes shall contain the Collection or Resource specific URIs for that type of Collection, and `resourceSpecificGroupingName` and `ResourceSpecificElementName` shall be replaced with the name of the Collection-specific Resource name, e.g., `machines` in JSON or `Machine` in XML.

The above serialization shows that each entry in a Collection may contain “resource-specific data” beside the reference to the Resource item and the common attributes. This placeholder represents two kinds of data:

a) Optionally some *accessory attributes* that represent accessory information for the use of this reference in the context of the Resource owning that Collection (the accessory attributes) – e.g., the “initial location” of a referenced `Volume`, in a Collection of Volumes associated with a `Machine`. Accessory attributes – if any - are part of the definition of each specific Collection.

b) All or a subset of the attributes of the corresponding Resource items. How much of the Resource item is expanded in the serialization of the Collection is controlled by expansion mechanisms described later.

If accessory attributes exist for items in a Collection, the “`resourceSpecificGroupingName` or `ResourceSpecificElementName`” is not just identifying the Resource type of Collection items, but is a unique name specific to this combination of accessory attributes and Resource type – e.g., for Volumes with initial location, it may be “locatedVolume”. Also the `resourceURI` of the Collection is unique to this combination. Because of this accessory attribute, the Collection of Volumes is said to be “enhanced”, as opposed to “basic” for a Collection without accessory attribute.

The serialization of Collections follows these additional rules:

- A Provider may limit the number of Resources returned in the Collection. The Consumer can determine this has occurred by comparing the number of returned Resources with the value of the "Count" attribute and any Collection subsetting query parameters it specified. In this case, the Consumer is advised to specify filter query parameters (see 4.1.6.1) to reduce the number of entries returned, or retrieve them in batches by issuing multiple requests with Collection subsetting query parameters (see 4.1.6.2)

- As with all Resources in the CIMI model, each Resource in the Collection shall have an `id` attribute that acts as a "self-pointer." Retrieving the data at this reference shall return just that one Resource and not any parent Resource, such as the Collection or array attribute.

- The serialization of a Collection may be controlled (see 4.1.6.4 `$expand` query parameter) to show more or less of each Resource item. By default, each entry in the Collection will show just a reference (URL) to the Resource item, along with the "common" attributes of the Resource.
item. Alternatively, the Resource item may be expanded partially or fully when querying the Collection.

- As with all arrays, if there are no Resources in the Collection, the serialization of the list shall be omitted.

### 5.5.12.1 Adding an Item to a Collection

Invoking the "add" operation of a Collection shall create a new Resource and add it to the Collection. The contents of the request body shall be either a representation of the new Resource being added to the Collection, or a representation of the Template associated with the new Resource being created and resource specific data attributes.

If creating a new Resource, the "add" operation shall contain:

- The "common attributes" as defined by clause 5.7.1
- The Resource specific data needed to create it. This data shall either be a reference to the Resource-specific Template Resource or be the Resource-specific Template Resource itself inlined.
- Accessory attributes—if any—that represent accessory information for the use of the reference in the context of the Resource owning that Collection (the associative attributes)
- In the XML case, a wrapper element (named after the pattern `<ResourceNameCreate>`) For example, to create a new Machine (which requires the use of a Template) and add it to the MachineCollection, the "add" operation of the MachineCollection shall be serialized as follows:

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/MachineCreate",
    "name": string,
    "description": string,
    "properties": {
        string: string,
    },
    "machineTemplate": {
        "href": string
    }
}
```

**XML serialization:**

```xml
<MachineCreate xmlns="http://schemas.dmtf.org/cimi/2">
    <name> xs:string </name> ?
    <description> xs:string </description> ?
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
    <machineTemplate href="xs:anyURI"/>
    <xs:any/>
</MachineCreate>
```

The MachineCollection has a new Machine:
JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Machine",
  "id": string,
  "name": string,
  ...
}
```

XML serialization:

```xml
<Machine xmlns="http://schemas.dmtf.org/cimi/2">
  <id>xs:anyURI</id>
  <name>xs:string</name>
  ...
</Machine>
```

The processing of the "add" operation shall adhere to the semantics defined in clause 4.2.1.1.

Regardless of whether a Template is used, the "add" operation shall create the new Resource and add it to the Collection and a reference (URI) to the new entry shall be returned in the response message in the HTTP Location header.

5.5.12.2 Inserting an item in a Collection

Invoking the "insert" operation of a Collection shall add, to the Collection, a new reference to an existing Resource. The contents of the request body shall specify the URL of the existing Resource being added and the accessory attributes in case of an "enhanced" collection.

To add an existing Volume to the volumes Collection of a Machine, the request body of the "insert" operation shall be serialized as follows:

JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/insert",
  "initialLocation": string,
  "volume": { "href": string }
}
```

XML serialization:

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action>http://schemas.dmtf.org/cimi/2/action/insert</action>
  <initialLocation>xs:string</initialLocation>
  <volume href="xs:string"/>
</Action>
```

Note that "initialLocation" is an accessory attribute to each reference of Volume. Because of this addition, the type of the collection items is distinguished from Volume, and called here locatedVolume.

The definition of the volumes Collection of the Machine Resource describes the accessory attribute(s) for this Collection.
5.5.12.3 Removing an item from a Collection

Invoking the "remove" operation of a Collection shall delete the specified item in the Collection, i.e., the Resource reference along with accessory attributes if any, without destroying the referenced Resource item itself. The contents of the request body shall be the URL of the Resource item being removed.

To remove a Volume from the volumes Collection of a Machine, the request body of the "remove" operation shall be serialized as follows:

**JSON serialization:**

```
{    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/remove",
    "volume": {    "href": string }
}
```

**XML serialization:**

```
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action>http://schemas.dmtf.org/cimi/2/action/remove</action>
    <volume href="xs:string"/>
</Action>
```

Removing the referenced Resource (here a Volume) deletes the related entry from the Collection. This deletes the reference but not the Resource itself.

5.5.12.4 Deleting an item in a Collection

Deleting the Resource referenced by a Collection item via a DELETE operation on the Resource itself (in the previous example, a Volume) also deletes the related entry from the Collections that reference this Resource – i.e., it has the effect of a “remove” on the Collection, in addition to deleting the referenced Resource.

5.5.13 "Any" type

Some attributes are polymorphic and can hold various data types, the list of which is indicated in their description. In such cases, the type of the attribute shall be indicated as "any" in the model representation.

5.5.14 valueScope

The valueScope type is a specialized map. Its goal is to define possible values for a list of attributes of a Resource. The possible values for an attribute are called the "value scope" of the attribute, and a combination of attribute value scopes (in form of a map) in a Resource or in the ResourceMetadata is called the value scope of the Resource.

Each item in a valueScope is a key-value pair where:

- The key is the name of an attribute of a Resource – or "scoped attribute" – for which a set of possible values is defined.

- The value is a structure that defines the "scope", i.e., a range, an enumeration or a single assigned value for the scoped attribute.
The scope structure:

A "scope" structure – or the value part of a key-value item in a valueScope – can take one of following forms:

1) An assigned single value, along with its (optional) units, e.g., for a scoped attribute named "cpu":

```
"cpu": { "value": 2000, "units": "megahertz" }
```

In the above example, value and units are reserved keywords for defining the value scope.

2) A range of values, along with its optional units, and an optional increment e.g., for a scoped attribute named "memory". The range may be open-ended: either the minimum or the maximum may be missing. The increment specifies the allowed values starting from the minimum and upward - i.e., the allowed values are of the form: minimum+N*(increment), where N>=0, or starting from the maximum and downward in case there is no minimum, i.e., allowed values are of the form: maximum-N*(increment).

```
"memory": { "minimum": 4000, "maximum": 10000, "units": "kibibytes", "default": 4000, "increment": 2000 }
```

In the above example, minimum, maximum, default, increment, and units are reserved keywords for defining the value scope.

3) An enumeration (or values), along with its (optional) units, e.g., for a scoped attribute named "cpuArch":

```
"cpuArch": { "values": [ "68000", "Alpha", "ARM", " PA_RISC"], "default": "PA_RISC"
```

4) Simply a required units, e.g., for a scoped attribute named "capacity":

```
"capacity": { "units": "megabytes" }
```

5) Any of the above, applying to the items in a collection, e.g., for a range of values that applies to the accessory attribute named "remoteLocation" of type URI for every item in a collection named machines:

```
"machines": { "item": { "remoteLocation": { "values": [ "URI1", "URI2", "URI3"], "default": "URI1" } }
```

In the above, item, values, and default are reserved keywords for defining the value scope.

If a valueScope is associated with a Resource type, it shall be in form of an attribute named "vscope", of type array of valueScope (i.e., valueScope[]).

An example of valueScope for the MachineConfiguration Resource:

```
"vscope": [ {
"cpu": { "value": 1 },
"memory": { "minimum": 4, "maximum": 32, "units": "GbB", "default": 4, "increment": 2 },
"cpuArch": { "values": [ "68000", "Alpha", "ARM", " PA_RISC", "i5"], "default": "i5" }
} ]
```

Semantics of valueScope array in a Resource

The value scope of a Resource shall be represented by an array of valueScope instances, even if, in many cases, this array will contain a single valueScope instance. This allows for expressing dependencies between values of different attributes of a same Resource. In such cases, the scoped attributes of the Resource must satisfy either valueScope instance in this array.
In the following example, `vscope` is an array of two valueScope items:

```
"vscope": [ {
  "cpuSpeed": { "minimum": 2, "maximum": 4, "units": "GHz", "default": 2.5},
  "memory": {"minimum": 2000000, "maximum": 10000000, "units": "KbB", "increment": 2000000 },
  "cpuArch": { "value": "i5" }
  }, {
  "memory": { "minimum": 4000000, "maximum": 32000000, "units": "KbB" },
  "cpuArch": { "values": [ "68000", "Alpha", " PA_RISC" ]
  }
}
```

This valueScope means that the Provider supports MachineConfigurations with either `cpuArch` of value `i5`, or `cpuArch` of a value that is one of `{ "68000", "Alpha", " PA_RISC" }`. In the first case (`i5`), the memory must be within the 2GbB-10GbB range and `cpuSpeed` must be between 2-4 GHz, while in the second case the memory must be within the 4GbB-32GbB range.

The following pseudo-schemas describe the serialization of the valueScope map in both JSON and XML:

**JSON serialization:**
```
( "value": any,
  "units": string ? ) |
( "values": [ any,+ ],
  "units": string ,? 
  "default": string ? ) |
( "minimum": number, ?
  "maximum": number, ?
  "units": string ,?
  "default": number, ?
  "increment": number ? )
```

**XML serialization:**
```
( <value> xs:any </value>
 <units> xs:string </units> ? ) |
( <value> xs:any </value> +
 <units> xs:string </units> ? 
 <default> xs:any </default> ? ) |
( <minimum> xs:integer </minimum> ?
 <maximum> xs:integer </maximum> ?
 <units> xs:string </units> ?
 <default> xs:integer </default> ?
 <increment> xs:integer </increment> ? )
```

A Provider who supports value scopes shall set the ValueScopes capability (ResourceMetadata) to “true”.

5.5.15 Empty attribute values

Attributes of the following types are omitted in cases where they have an empty value: string, map, array, and Collection. Apart from being “Provider optional” or “Consumer optional”, an empty value is the third reason that the serialization schema contains an ’?’ or an ‘*’ for an attribute.

Other attribute types do not have empty values and shall not be omitted from the serialization for this reason.

5.6 Units

Some of the Resources defined by this specification have attributes that describe an amount of something that belongs to, or is associated with, that Resource. For example, the Machine Resource has a memory attribute that describes “the size of the memory allocated to this machine.” The allowable units of these attributes are listed in Table 4. Their meaning is defined in 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>

5.7 Resources

CIMI Resources are representations of actual – either virtual or physical – resources available in a cloud. Resources are identified and separately accessible by their URI. Every Resource has a type that is described in this clause. A Resource type defines a set of attributes and operations.

5.7.1 Common Resource attributes

Resources, except for the Collection Resource, shall support the following common attributes defined in Table 5. A Collection Resource shall support the id attribute, the updated attribute and the parent attribute, as defined in Table 5.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>URI</td>
<td>The unique URI identifying this Resource; assigned upon Resource creation. This attribute value shall be <strong>unique</strong> in the Provider’s cloud. <strong>Constraints:</strong> providerMandatory: true consumerMandatory: true mutable: false consumerWritable: false</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The human-readable name of this Resource; assigned by the creator</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>as a part of the Resource creation input.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The human-readable description of this Resource; assigned by the creator as a part of the Resource creation input.</td>
</tr>
<tr>
<td>created</td>
<td>dateTime</td>
<td>The timestamp when this Resource was created. The format should be unambiguous, and the value is immutable.</td>
</tr>
<tr>
<td>updated</td>
<td>dateTime</td>
<td>The time at which the last explicit attribute update was made on the Resource. The initial value is the time the resource is created. Note, while operations, such as “stop”, do implicitly modify the ‘state’ attribute, they do not change the ‘updated’ time.</td>
</tr>
</tbody>
</table>
### Attribute Types and Descriptions

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>ref</td>
<td>A reference to a Resource of which this Resource is a child component (see &quot;composition&quot; relationship, clause 5.10.2) – i.e., a reference to its first parent Resource. Constraints: providerMandatory: true consumerMandatory: false mutable: true consumerWritable: true</td>
</tr>
<tr>
<td>properties</td>
<td>map</td>
<td>A map of key-value pairs (each entry called a &quot;property&quot;), some of which may control one or more aspects this Resource. Properties may also serve as an extension point, allowing Consumers to record additional information about the Resource. The same &quot;key&quot; shall not be used more than once within a &quot;properties&quot; attribute. Each property shall contain the following nested data: Name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Type Description</td>
</tr>
<tr>
<td>key</td>
<td>string</td>
<td>The name of the property.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The value of the property.</td>
</tr>
</tbody>
</table>

The following pseudo-schemas describe the serialization of these attributes in both JSON and XML:

**JSON serialization:**

```
"id": string,
"name": string, ?
"description": string, ?
"created": string, ?
"updated": string, ?
"properties": { string: string, + }, ?
"resourceMetadata" : ["href": string, "? "], ?
```
5.8 Operations

All Resource operations defined by this specification are optional for Providers to support. Consumers, by way of examination of a Resource's ResourceMetadata, can determine which operations are supported. However, even for those operations that are supported, Consumers still need to examine each Resource's representation to determine which operations are supported at that moment. Whether an operation is supported is based on a number of factors, including the state of the Resource and access control rights of the Consumer (see clause 4.2). Operations and states are coupled; i.e., if implementing a state-changing Resource operation defined in this specification, the corresponding state(s) shall also be implemented. See the Resource-specific "Operations" clauses for additional detail.

The "State" attribute of Resources that have this attribute shall only change value if

- an operation is performed on this Resource and this operation requires a state change, or
- an error occurred, in this case the “State” attribute shall obtain the value “ERROR”.

For example, for a 'start' operation on a Machine both the STARTING and the STARTED states are required to be supported by the Machine, while the Machine can only leave the STARTED state after another state-changing operation is requested, unless an error occurs.

Providers can define additional operations and states. Such extensions shall fall into one of these categories:

a) A new operation that starts from a CIMI-defined state, or leads to a CIMI-defined state, or both. In the latter case, if a CIMI-defined operation already exists for this transition between two CIMI-defined states, it shall also be supported by the Provider in addition to the new operation.

b) A new Resource state. In that case, a new operation that leads to that state shall also be created. In other words, a Provider-defined operation has to be performed before a Provider-defined state can be reached.

c) A new operation that transitions between two Provider-defined states.

5.9 Alternative model formats

It is expected that this specification is implemented by 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.

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.
5.10 Relationships between Resources

5.10.1 Referencing across Resources

Resources may refer each other. This referencing expresses a directional relationship in which there is a \textit{referring} Resource and a \textit{referred} Resource. Depending on the cardinality of such relationships, there are two representations:

- For 1-to-1 referencing, the URL of the referred Resource appears as an attribute in the referring Resource.
- For 1-to-n referencing, the referred Resources (all of the same type) are grouped in a Collection, the URL of which appears as an attribute in the referring Resource. In that case, the \textit{referring} Resource does not refer directly to the referred Resources, but instead to a Collection Resource that contains references to the \textit{referred} Resources.

If a \textit{referred} Resource is deleted but not the \textit{referring} Resource(s), then in case of a 1-to-1 relationship the reference shall be set to empty in every \textit{referring} Resource, and in case of a 1-to-n relationship the reference shall be removed from any Collection where it appears as an item.

5.10.2 Composition relationship between Resources

A Resource is a child component of another Resource if its \texttt{parent} attribute refers to the latter Resource. This relationship is transitive.

If a Resource is deleted, its child component Resources are also automatically deleted.

In case of a Collection Resource that is referred by a Resource \texttt{R}, expressing a composition relationship from the Collection Resource items to \texttt{R} is done by:

(a) setting the \texttt{parent} attribute of each Resource item to the Collection Resource, and
(b) by setting the \texttt{parent} attribute of the Collection Resource to the Resource \texttt{R}.

A Resource is said to be parent of its children components.

In any Resource description \texttt{R} throughout this specification, an attribute of type \texttt{"collection"]} refers to a Collection Resource that has the Resource \texttt{R} as a parent, unless indicated otherwise.

For example a Machine is parent of its related Disk Resources via the \texttt{disks Collection}: the \texttt{parent} attribute of a Disk is set to the \texttt{disks Collection}, and the \texttt{parent} attribute of the \texttt{disks Collection} is set to the Machine.

Some composed Resources – e.g., System - may have component Resources that are not their “children”. Such Resources are called associated components. Their \texttt{parent} attribute refers to another Resource or to the Cloud Entry Point (CEP), meaning the deletion of the composed Resource does not cause the deletion of its associated components, even if the associated components are still otherwise managed by the composed Resource.

5.11 Resource metadata

Implementations of this specification should allow for Consumers to discover the metadata associated with any Resource under the Cloud Entry Point. Doing so allows for the discovery of Provider-defined constraints on the attributes or operations of a Resource as well as discovery of any new extension attributes or operations that the Provider may have defined.

A \texttt{ResourceMetadata} instance contains metadata governing the attribute status (optionality, value constraints, access), the available operations, and other Provider-specific capabilities or features for a Resource or a set of Resources, called the target Resource(s) for that \texttt{ResourceMetadata} instance.
The target Resource contains a reference to its ResourceMetadata instance, which itself may be shared across several target Resources.

Any Resource under a CEP may have a ResourceMetadata instance associated with it. This association may be done in one of the following ways:

- A ResourceMetadata instance is defined for all Resources of a same type under the CEP. In such a case the ResourceMetadata instance is added as a Resource item in the resourceMetadata collection unique to the CEP. Unless overridden, it applies to all Resources of the targeted type under this CEP.

- A ResourceMetadata instance is defined for all Resources generated from a same template. In such a case, a Template-specific ResourceMetadata instance is provided and referred by this Template. This ResourceMetadata overrides any CEP-level ResourceMetadata (1) for the type of Resource generated from this Template.

- A ResourceMetadata instance may be created for a single particular Resource instance, or may be associated on a per-Resource basis. Such an association requires an explicit modification of the resourceMetadata attribute of the target Resource, canceling any former value it may have been given at creation time, e.g., in either of the above cases.

Each Resource's metadata shall contain the following pieces of information:

<table>
<thead>
<tr>
<th>Name</th>
<th>ResourceMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ResourceMetadata">http://schemas.dmtf.org/cimi/2/ResourceMetadata</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| typeURI     | URI  | A unique URI associated with, and denoting, the type of the described Resource target. **Constraints:**
|             |      | providerMandatory: true
|             |      | consumerMandatory: true
|             |      | mutable: true
|             |      | consumerWritable: true |
| name        | string | The name of the Resource target type (e.g., Machine). **Constraints:**
|             |      | providerMandatory: true
|             |      | consumerMandatory: true
|             |      | mutable: true
<p>|             |      | consumerWritable: true |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>ResourceMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ResourceMetadata">http://schemas.dmtf.org/cimi/2/ResourceMetadata</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributes</td>
<td>attribute[]</td>
<td>A set of metadata associated with each attribute (or target attribute) of the Resource target, including the set of extension attributes not defined in this specification. The metadata for each attribute target shall contain the following nested data:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Name</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Data</strong></td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>namespace</td>
<td>URI</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td></td>
</tr>
<tr>
<td>provider</td>
<td>boolean</td>
<td></td>
</tr>
<tr>
<td>consumer</td>
<td>boolean</td>
<td></td>
</tr>
<tr>
<td>mutable</td>
<td>boolean</td>
<td></td>
</tr>
<tr>
<td>consumer</td>
<td>boolean</td>
<td></td>
</tr>
</tbody>
</table>

Every above attribute in the nested attribute table has the following constraints:
providerMandatory: true
c consumerMandatory: true
mutable: true
c consumerWritable: true

The constraints for the attributes attribute of ResourceMetadata are:
Constraints:
providerMandatory: false
c consumerMandatory: false
mutable: true
c consumerWritable: true
<table>
<thead>
<tr>
<th>Name</th>
<th>ResourceMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ResourceMetadata">http://schemas.dmtf.org/cimi/2/ResourceMetadata</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| vscope    | valueScope[] | The `vscope` attribute applies to the attributes of the described – or target – Resource. The target Resource shall be of the type identified by the `typeURI` attribute. Consequently this value scope is about the list of attributes described in the `attributes` attribute. If an attribute of the target Resource is constrained by the `vscope`, a Consumer shall set a value (creation or update request) compatible with the value scope of this attribute. For any other case where the Consumer sets an incompatible value, the Provider shall return a 4xx error code. **Constraints:**  
providerMandatory: false  
consumerMandatory: false  
mutable: true  
consumerWritable: true |
### ResourceMetadata

**Type URI**
http://schemas.dmtf.org/cimi/2/ResourceMetadata

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capabilities</td>
<td>capability[]</td>
<td>A set of Provider-defined metadata that can be used by Consumer to discover any capability or feature provided by this Provider. Each capability shall contain the following nested data:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the capability.</td>
</tr>
<tr>
<td>uri</td>
<td>URI</td>
<td>A URI that uniquely identifies the capability at a global level. Constraints: consumerMandatory: true</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The human-readable description of the semantic of the capability.</td>
</tr>
<tr>
<td>value</td>
<td>any</td>
<td>The value of the capability. The specific type varies depending on the definition of the capability. If not present the capability defaults to a &quot;boolean&quot; type with a value of &quot;true&quot; indicating that the specific capability is supported by the Provider. Constraints: consumerMandatory: true</td>
</tr>
</tbody>
</table>

Every above attribute in the nested capability table has the following constraints by default (unless overridden per attribute): providerMandatory: true consumerMandatory: false mutable: true consumerWritable: true

The constraints for the capabilities attribute of ResourceMetadata are:

**Constraints:**
- providerMandatory: false
- consumerMandatory: false
- mutable: true
- consumerWritable: true
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

<table>
<thead>
<tr>
<th>Name</th>
<th>ResourceMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ResourceMetadata">http://schemas.dmtf.org/cimi/2/ResourceMetadata</a></td>
</tr>
</tbody>
</table>

**Attribute** | **Type** | **Description**                                                                                                                                                                                                 |
|--------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| actions      | action[] | A set of Provider-defined operations that can be used by consumers to act on the Resource. This set represents all operations defined for this described Resource type, which may be a superset of those operations a particular Consumer is actually allowed to use. The subset of allowed operations for a particular Consumer shall be those operations returned to this Consumer if querying an instance of the described Resource type. Note that this attribute is called "actions" so as not to conflict with the ResourceMetadata Resource’s own operations. Each operation shall contain the following nested data:

<table>
<thead>
<tr>
<th>Name</th>
<th>Data</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>name</td>
<td>string</td>
<td>The name of the operation.</td>
</tr>
<tr>
<td>uri</td>
<td>uri</td>
<td>URI</td>
<td>A URI that uniquely identifies the operation at a global level.</td>
</tr>
<tr>
<td>description</td>
<td>description</td>
<td>string</td>
<td>The human-readable description of the semantic of the operation. Constraints: consumerMandatory: false</td>
</tr>
<tr>
<td>method</td>
<td>method</td>
<td>string</td>
<td>The protocol-dependent verb to use to perform the operation.</td>
</tr>
<tr>
<td>inputMessage</td>
<td>inputMessage</td>
<td>string</td>
<td>The body mimeType of the request message; it may depend on the model format chosen by the Provider.</td>
</tr>
<tr>
<td>outputMessage</td>
<td>outputMessage</td>
<td>string</td>
<td>The body mimeType of the response message; it may depend on the model format chosen by the Provider.</td>
</tr>
</tbody>
</table>

Every above attribute in the nested action table has the following constraints by default (unless overridden per attribute):

- providerMandatory: true
- consumerMandatory: true
- mutable: true
- consumerWritable: true

The constraints for the actions attribute of ResourceMetadata are:

Constraints:
- providerMandatory: false
- consumerMandatory: false
- mutable: true
- consumerWritable: true

1892 When implementing or using ResourceMetadata, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 6 as well as in the tables describing embedded Resources or related Collections. Both Consumer and Provider shall serialize this Resource as described below. The following pseudo-schemas (see notation in 1.3) describe the serialization of the Resource in both JSON and XML:
JSON media type: application/json

JSON serialization:

```
{ "resourceURI": "http://schemas.dmtf.org/cimi/2/ResourceMetadata",
  "id": string,
  "typeURI": string,
  "name": string,
  "attributes": [  
    { "name": string,  
      "namespace": string, ?  
      "type": string, ?  
      "required": boolean, ? } *  
  ], ?,  
  "vscope": [ valueScope, * ], ?,  
  "capabilities": [  
    { "name": string, ?  
      "uri": string,  
      "description": string, ?  
      "value": any } *  
  ], ?,  
  "actions": [  
    { "name": string,  
      "uri": string,  
      "description": string, ?  
      "method": string,  
      "inputMessage": string, ?  
      "outputMessage": string ? }, *  
  ], ?,  
  "operations": [  
    { "rel": "edit", "href": string }, ?,  
    { "rel": "delete", "href": string } ?,  
  ] ?,  
} ...
```

XML media type: application/xml

XML serialization:

```
<ResourceMetadata xmlns="http://schemas.dmtf.org/cimi/2">
  <id> xs:anyURI </id>
  <name> xs:string </name>
  <typeURI> xs:anyURI </typeURI>
</ResourceMetadata>
```
Additional metadata about the Resource or attributes may be included by the Provider.

5.11.1 Capabilities

Table 7 describes the capability URIs defined by this specification. Providers may define new URIs and it is recommended that these URIs be dereferencable such that Consumers can discover the details of the new capability. The "Resource Name" column contains the name of the Resource that may contain the specified capability within its ResourceMetadata. The "Capability Name" column contains the name of the specified capability and shall be unique within the scope of the corresponding Resource. Each capability's URI shall be constructed by appending the "Resource Name", a slash (/), and the "Capability Name" to "http://schemas.dmtf.org/cimi/2/capability/". For example, the Machine's "InitialState" capability shall have a URI of:

http://schemas.dmtf.org/cimi/2/capability/Machine/InitialState

Capabilities that apply to the Provider in general, and are not specific to any one Resource, shall be associated with the CloudEntryPoint Resource (in case a capability applies only to the CloudEntryPoint Resource itself, its definition indicates this).

Each one of these capabilities may be set to some value, or may be absent. The meaning of an absent capability is defined as follows:

- For boolean-valued capabilities: same as a “false” value.
- For other capabilities that use a single value or a list of values among an enumeration: same as no particular preference or restriction being enforced for this value.
Table 7 – Capability URIs

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Capability Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CloudEntryPoint</td>
<td>ExpandParameter</td>
<td>If true, the Provider shall support the $expand query parameter.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>FilterParameter</td>
<td>If true, the Provider shall support the $filter query parameter.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>FirstParameter</td>
<td>If true, the Provider shall support both the $first and $last query parameters.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>SelectParameter</td>
<td>If true, the Provider shall support the $select query parameter.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>FormatParameter</td>
<td>If true, the Provider shall support the $format query parameter.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>OrderByParameter</td>
<td>If true, the Provider shall support the $orderby query parameter.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>QueryPathNotation</td>
<td>If true, the Provider shall support the use of path-like notation with query parameter $select (see 4.1.6.3) to disambiguate between attributes of a Collection Resource and attributes of each items in the Collection if subsetting.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>MaxPropertyItems</td>
<td>If set, the Provider shall support a ‘Properties’ attribute with a number of elements less than or equal to the size specified by this capability.</td>
</tr>
<tr>
<td>CloudEntryPoint</td>
<td>ValueScopes</td>
<td>If true, the Provider shall support the use of attributes of type valueScope, for any Resource that may be created via a template.</td>
</tr>
<tr>
<td>System</td>
<td>SystemComponentTemplateByValue</td>
<td>If true, the Provider shall support the specification of ComponentTemplates by value in SystemTemplates.</td>
</tr>
<tr>
<td>Machine</td>
<td>DefaultInitialState</td>
<td>If this capability is set, unless otherwise provided (e.g., by a MachineTemplate &quot;initialState&quot; attribute), the Provider shall set a new Machine to this state value, assuming the value is compatible with the InitialStates capability, if set.</td>
</tr>
<tr>
<td>Machine</td>
<td>InitialStates</td>
<td>If this capability is set, and if using a MachineTemplate that has an &quot;initialState&quot; attribute, a Consumer shall use an initialState value from the set of values of this capability.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineConfigByValue</td>
<td>If true, the Provider shall support specifying MachineConfigurations by value. If true, the MachineTemplateByValue capability shall also have the value true.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineCredentialByValue</td>
<td>If true, the Provider shall support specifying Credentials by value in Machine create operations. If true, the MachineTemplateByValue capability shall also have the value true.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineImageByValue</td>
<td>If true, the Provider shall support specifying MachineImages by value in Machine create operations. If true, the MachineTemplateByValue capability shall also have the value true.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineVolumeTemplatesByValue</td>
<td>If true, the Provider shall support specifying VolumeTemplates by value in Machine create operations. If true, the MachineTemplateByValue capability shall also have the value true.</td>
</tr>
<tr>
<td>Resource Name</td>
<td>Capability Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineTemplateByValue</td>
<td>If true, the Provider shall support specifying MachineTemplates by value in Machine create operations.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineStopForce</td>
<td>If true, the Provider shall support the “force” option on the stop and restart operations on Machines.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineStopForceDefault</td>
<td>If true, the Provider shall forcefully stop Machines if no other indication is provided. Otherwise, the Provider shall gracefully stop Machines.</td>
</tr>
<tr>
<td>Machine</td>
<td>RestoreFromImage</td>
<td>If true, the Provider supports restoring Machines from MachineImages that are not SNAPSHOT MachineImages.</td>
</tr>
<tr>
<td>Machine</td>
<td>UserData</td>
<td>If set, indicates which userData injection method shall be used by the Provider.</td>
</tr>
<tr>
<td>Machine</td>
<td>MachineAvailabilityLevel</td>
<td>If true, the Provider supports the notion of an availability level for the Machine Resource. The availability level and its value constraints are advertised as an extension attribute by way of the Machine and MachineTemplate ResourceMetadata.</td>
</tr>
<tr>
<td>Credential</td>
<td>CredentialTemplateByValue</td>
<td>If true, the Provider shall support specifying CredentialTemplates by value in Credential create operations.</td>
</tr>
<tr>
<td>Volume</td>
<td>SharedVolumeSupport</td>
<td>If true, the Provider shall support that a single Volume Resource can be shared by multiple Machines.</td>
</tr>
<tr>
<td>Volume</td>
<td>VolumeConfigByValue</td>
<td>If true, the Provider shall support specifying VolumeConfigurations by value in the Volume create operation. If true, the VolumeTemplateByValue capability shall have the value true.</td>
</tr>
<tr>
<td>Volume</td>
<td>VolumeImageByValue</td>
<td>If true, the Provider shall support specifying VolumeImages by value in the Volume create operation. If true, the VolumeTemplateByValue capability shall have the value true.</td>
</tr>
<tr>
<td>Volume</td>
<td>VolumeSnapshot</td>
<td>If true, the Provider shall support creating a new VolumeImage by referencing an existing Volume.</td>
</tr>
<tr>
<td>Volume</td>
<td>VolumeTemplateByValue</td>
<td>If true, the Provider shall support specifying the VolumeTemplates by value in Volume create operations.</td>
</tr>
<tr>
<td>Volume</td>
<td>VolumeAvailabilityLevel</td>
<td>If true, the Provider supports the notion of an availability level for the Volume Resource. The availability level and its value constraints are advertised as an extension attribute by way of the Volume and VolumeTemplate ResourceMetadata.</td>
</tr>
<tr>
<td>Network</td>
<td>NetworkTemplateByValue</td>
<td>If true, the Provider shall support specifying Network Templates by value in Network create operations.</td>
</tr>
<tr>
<td>Network</td>
<td>DefaultInitialState</td>
<td>If this capability is set, unless otherwise provided (e.g., by a NetworkTemplate “initialState” attribute), the Provider shall set a new Network to this state value, assuming the value is compatible with the InitialStates capability, if set.</td>
</tr>
<tr>
<td>Resource Name</td>
<td>Capability Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Network</td>
<td>InitialStates</td>
<td>If this capability is set, and if using a NetworkTemplate that has an “initialState” attribute, a Consumer shall use an initialState value from the set of values of this capability.</td>
</tr>
<tr>
<td>NetworkInterface</td>
<td>NetworkInterfaceTemplateByValue</td>
<td>If true, the Provider shall support specifying NetworkInterface Templates by value in NetworkInterface create operations.</td>
</tr>
<tr>
<td>NetworkInterface</td>
<td>DefaultInitialState</td>
<td>If this capability is set, unless otherwise provided (e.g., by a NetworkInterfaceTemplate “initialState” attribute), the Provider shall set a new NetworkInterface to this state value, assuming the value is compatible with the InitialStates capability, if set.</td>
</tr>
<tr>
<td>NetworkInterface</td>
<td>InitialStates</td>
<td>If this capability is set, and if using a NetworkInterfaceTemplate that has an “initialState” attribute, a Consumer shall use an initialState value from the set of values of this capability.</td>
</tr>
<tr>
<td>NetworkService</td>
<td>NetworkServiceTemplateByValue</td>
<td>If true, the Provider shall support specifying NetworkService Templates by value in NetworkService create operations.</td>
</tr>
<tr>
<td>NetworkService</td>
<td>DefaultInitialState</td>
<td>If this capability is set, unless otherwise provided (e.g., by a NetworkServiceTemplate “initialState” attribute), the Provider shall set a new NetworkService to this state value, assuming the value is compatible with the InitialStates capability, if set.</td>
</tr>
<tr>
<td>NetworkService</td>
<td>InitialStates</td>
<td>If this capability is set, and if using a NetworkServiceTemplate that has an “initialState” attribute, a Consumer shall use an initialState value from the set of values of this capability.</td>
</tr>
<tr>
<td>ProtocolEndpoint</td>
<td>ProtocolEndpointTemplateByValue</td>
<td>If true, the Provider shall support specifying ProtocolEndpoint Templates by value in ProtocolEndpoint create operations.</td>
</tr>
<tr>
<td>ProtocolEndpoint</td>
<td>DefaultInitialState</td>
<td>If this capability is set, unless otherwise provided (e.g., by a ProtocolEndpointTemplate “initialState” attribute), the Provider shall set a new ProtocolEndpoint to this state value, assuming the value is compatible with the InitialStates capability, if set.</td>
</tr>
<tr>
<td>ProtocolEndpoint</td>
<td>InitialStates</td>
<td>If this capability is set, and if using a ProtocolEndpointTemplate that has an “initialState” attribute, a Consumer shall use an initialState value from the set of values of this capability.</td>
</tr>
<tr>
<td>ProtocolSegment</td>
<td>ProtocolSegmentTemplateByValue</td>
<td>If true, the Provider shall support specifying ProtocolSegment Templates by value in ProtocolSegment create operations.</td>
</tr>
<tr>
<td>ProtocolSegment</td>
<td>DefaultInitialState</td>
<td>If this capability is set, unless otherwise provided (e.g., by a ProtocolSegmentTemplate “initialState” attribute), the Provider shall set a new ProtocolSegment to this state value, assuming the value is compatible with the InitialStates capability, if set.</td>
</tr>
<tr>
<td>ProtocolSegment</td>
<td>InitialStates</td>
<td>If this capability is set, and if using a ProtocolSegmentTemplate that has an “initialState” attribute, a Consumer shall use an initialState value from the set of values of this capability.</td>
</tr>
</tbody>
</table>
The following examples show the ResourceMetadata for a Machine that advertises some of its capabilities:

**JSON serialization:**

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/ResourceMetadata",
  "id": "http://example.com/types/Machine",
  "typeURI": "http://schemas.dmtf.org/cimi/2/Machine",
  "name": "Machine",
  "capabilities": [ 
    { "uri": "http://schemas.dmtf.org/cimi/2/capability/Machine/MachineConfigByValue",
      "value": true },
    { "uri": "http://schemas.dmtf.org/cimi/2/capability/Machine/MachineImageByValue",
      "value": true },
    { "uri": "http://schemas.dmtf.org/cimi/2/capability/Machine/DefaultInitialState",
      "value": "STARTED" } ]
}
```

**XML serialization:**

```xml
<ResourceMetadata xmlns="http://schemas.dmtf.org/cimi/2">
  <id> http://example.org/types/Machine </id>
  <typeURI> http://schemas.dmtf.org/cimi/2/Machine </typeURI>
  <name> Machine </name>
  <capabilities>
    <capability url="http://schemas.dmtf.org/cimi/2/capability/Machine/MachineConfigByValue">
      true
    </capability>
    <capability url="http://schemas.dmtf.org/cimi/2/capability/Machine/MachineImageByValue">
      true
    </capability>
  </capabilities>
</ResourceMetadata>
```
true
</capability>

```xml
<capability
  uri="http://schemas.dmtf.org/cimi/2/capability/Machine/DefaultInitialState">
  STARTED
</capability>
</capabilities>
</ResourceMetadata>

5.11.2 ResourceMetadataCollection Resource

A ResourceMetadataCollection Resource represents the Collection of ResourceMetadata Resources within a Provider and follows the Collection pattern defined in clause 5.5.12. Note that modifications of the Resources within this Collection are typically reserved for administrator types of CIMI Consumers. This Resource shall be serialized as follows:

JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/ResourceMetadataCollection",
  "id": "string",
  "count": number,
  "resourceMetadatas": [
    {
      "resourceURI": "http://schemas.dmtf.org/cimi/2/ResourceMetadata",
      "id": "string",
      ... remaining ResourceMetadata attributes ...
    }, +
  ],
  "operations": [ {
    "rel": "add",
    "href": "string" } ]
}
```
XML serialization:

```xml
<Collection
    resourceURI="http://schemas.dmtf.org/cimi/2/ResourceMetadataCollection"
    xmlns="http://schemas.dmtf.org/cimi/2">
    <id> xs:anyURI </id>
    <count> xs:integer </count>
    <resourceMetadatas>
        <ResourceMetadata>
            <id> xs:anyURI </id>
            ... remaining ResourceMetadata attributes ...
        </ResourceMetadata>
        *
    </resourceMetadatas>
    <operations>
        <operation rel="add" href="xs:anyURI"/>
    </operations>
    <xs:any>*
</Collection>
```

5.12 Cloud Entry Point

The Cloud Entry Point (CloudEntryPoint Resource) represents the entry point into the cloud defined by the CIMI Model. It provides a Consumer with a single address (URI) from which the Consumer can discover and access all Resources usable by this Consumer. A Cloud Provider may provide different Cloud Entry Points to different Consumers. The Cloud Entry Point (or CEP) implements a catalog of Resources, such as Systems, SystemTemplates, Machines, MachineTemplates, etc., that can be queried and browsed by the Consumer.

If a Consumer issues a read on the CloudEntryPoint Resource, the Provider shall return a CloudEntryPoint Resource that only catalogs Resources on which this Consumer is allowed to perform operations. Table 8 describes the attributes for the CloudEntryPoint Resource.

If the delete operation is advertised on the CEP, deleting the CloudEntryPoint Resource also deletes all referred Resources.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseURI</td>
<td>URI</td>
<td>An absolute URI that references the &quot;base URI&quot; of the Provider. This URI shall be used to convert relative URIs to Resources within this Provider to absolute URIs. See the &quot;URIs&quot; clause of 5.5. Constraints: providerMandatory: true consumerMandatory: true mutable: false consumerWritable: false</td>
</tr>
<tr>
<td>Name</td>
<td>Type URI</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://www.dmf.org/cimi/2/CloudEntryPoint">http://www.dmf.org/cimi/2/CloudEntryPoint</a></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type URI</td>
<td>Description</td>
</tr>
<tr>
<td>resourceMetadata</td>
<td>collection [Resource Metadata]</td>
<td>A reference to ResourceMetadata Collection of this Cloud Entry Point. The Collection contains a description of the Resources supported by the Provider. If a Resource does not have any metadata, it shall not appear in this list, e.g., it has no constraints beyond what the CIMI specification defines nor does it have any extension attributes.</td>
</tr>
<tr>
<td>systems</td>
<td>collection [System]</td>
<td>A reference to the SystemCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>systemTemplates</td>
<td>collection [System Template]</td>
<td>A reference to the SystemTemplateCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>credentials</td>
<td>collection [Credential]</td>
<td>A reference to the CredentialCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>credentialTemplates</td>
<td>collection [Credential Template]</td>
<td>A reference to the CredentialTemplateCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>volumes</td>
<td>collection [Volume]</td>
<td>A reference to the VolumeCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>volumeTemplates</td>
<td>collection [Volume Template]</td>
<td>A reference to the VolumeTemplateCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>networks</td>
<td>collection [Network]</td>
<td>A reference to the NetworkCollection of this Cloud Entry Point.</td>
</tr>
<tr>
<td>Name</td>
<td>Type URI</td>
<td>Attribute</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>networkTemplates</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>segmentTemplates</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>endPointTemplates</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>interfaces</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>interfaceTemplates</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>networkServices</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>networkServiceTemplates</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>jobs</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>meters</td>
</tr>
<tr>
<td>Name</td>
<td>CloudEntryPoint</td>
<td>meterTemplates</td>
</tr>
</tbody>
</table>
Name | CloudEntryPoint
---|---
Type URI | http://www.dmf.org/cimi/2/CloudEntryPoint

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meterConfigs</td>
<td>collection [Meter Configuration]</td>
<td>A reference to the MeterConfigurationCollection of this Cloud Entry Point.</td>
</tr>
</tbody>
</table>

Every above attribute of the CloudEntryPoint Resource has the following constraints by default (unless overridden per attribute):

- providerMandatory: false
- consumerMandatory: false
- mutable: true
- consumerWritable: true

Each of the Collections mentioned in Table 8 are defined within the related Resource definition clauses. For example, the MachineCollection Resource is defined in clause 5.14.2 as part of the Machine-related Resources. When implementing or using CloudEntryPoint, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 8 as well as in the tables describing embedded Resources or related Collections. Both Consumer and Provider shall serialize this Resource as described below. The following pseudo-schemas (see notation in 1.3) describe the serialization of the Resource in both JSON and XML:

**JSON media type:** application/json

**JSON serialization:**

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/CloudEntryPoint",
  "id": string,
  "name": string, ?
  "description": string, ?
  "created": string, ?
  "updated": string, ?
  "properties": { string: string, + }, ?
  "baseURI": string,
  "resourceMetadata": { "href": string }, ?
  "systems": { "href": string }, ?
  "systemTemplates": { "href": string }, ?
  "machines": { "href": string }, ?
  "machineTemplates": { "href": string }, ?
  "machineConfigs": { "href": string }, ?
```
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

```
"machineImages": { "href": string }, ?
"credentials": { "href": string }, ?
"credentialTemplates": { "href": string }, ?
"volumes": { "href": string }, ?
"volumeTemplates": { "href": string }, ?
"volumeConfigs": { "href": string }, ?
"volumeImages": { "href": string }, ?
"networks": { "href": string }, ?
"networkTemplates": { "href": string }, ?
"segments": { "href": string }, ?
"segmentTemplates": { "href": string }, ?
"endpoints": { "href": string }, ?
"endpointTemplates": { "href": string }, ?
"interfaces": { "href": string }, ?
"interfaceTemplates": { "href": string }, ?
"networkServices": { "href": string }, ?
"networkServiceTemplates": { "href": string }, ?
"jobs": { "href": string }, ?
"meters": { "href": string }, ?
"meterTemplates": { "href": string }, ?
"meterConfigs": { "href": string }, ?
"eventLogs": { "href": string }, ?
"eventLogTemplates": { "href": string }, ?
"operations": [?
    { "rel": "edit", "href": string }?
]?
...
```

XML media type: application/xml

XML serialization:

```
<CloudEntryPoint xmlns="http://schemas.dmtf.org/cimi/2">
  <id> xs:anyURI </id>
  <name> xs:string </name> ?
  <description> xs:string </description> ?
  <created> xs:dateTime </created> ?
  <updated> xs:dateTime </updated> ?
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties>
</CloudEntryPoint>
```
5.12.1 Operations

This Resource supports the Read and Update operations.
5.13 System Resources and relationships

5.13.1 System

A System is a realized Resource that consists of one or more Networks, Volumes, Machines, (and others) that could be connected and associated with each other. A System can be created from the interpretation of a SystemTemplate. A System can be operated and managed as a single Resource and usually forms a stack of service. For example, an online shopping cart system consists of machines for Web servers and databases, network addresses for public access, and volumes for database files. A System has several "top-level" attributes that are Collections of references to Resources of various types. Each one of these Collections shall contain references to Resource items of the related type that are components of the System. Each one of these System components may be either:

- a child component of the System (see 5.10.2)
- an associated component of the System

By default, all Resources that are created as the result of a System creation are also children components of the System. Some components of a System may pre-exist to the System – e.g., they would be referred to by the SystemTemplate used to create that System. Such component Resources are associated components of the System.

An example of an associated component in a System is of a Network created independently from the System, directly by POSTing to the networks CEP collection. A Consumer may then want the System to reuse that Network as a component while keeping the Network managed separately from the System, in particular not to be deleted when the System is deleted. Such a Network may still be inserted in the networks System collection as an associated component, while having its parent attribute referring to the CEP as originally set. Alternatively, the Network could be made a child component of the System by setting its parent attribute to the System’s networks collection Resource.

Note:

A Resource component of a System may in turn use some other Resources that are not component of this System, e.g., a Machine in a System can use a Volume that is neither a component of the Machine, nor a component of the System.

Table 9 describes the System attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>string</td>
<td>The operational state of the System. Allowed values are: (See 5.14.1.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>CREATING:</strong> The System is in the process of being created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>STARTING/STARTED/STOPPING/STOPPED/PAUSING/PAUSED/SUSPENDING/SUSPENDED:</strong> The System shall be in one of these states if all the Machines referenced by the System are in that state. See clause 5.14.1 for the list of available actions based on the state of a Machine. Such transitional states may just indicate that all Machines in a System are undergoing the same operation (e.g., “start”), without the System being actually operated on (e.g., no “start” done at System level). An actual operation on a System may be traced by querying the “job” entity.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>MIXED:</strong> The System shall be in this state if either no Machines are referenced by this System or Machines referenced by this System are in varying states. Such</td>
</tr>
</tbody>
</table>
varying states are likely to occur when an operation is in progress on a System, resulting in transitions of its Machine states toward a new common state (e.g., STOPPED, STARTED) but at a different pace, or sequentially one after the other. **DELETING**: The System is in the process of being deleted. **ERROR**: The Provider has detected an error in the System. The operations that result in transitions to the above defined states are defined in clause 5.13.1.2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>URI</td>
<td>Description</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>systems</td>
<td>collection [System]</td>
<td>A list of references to nested Systems that are components of this System.</td>
</tr>
<tr>
<td>machines</td>
<td>collection [Machine]</td>
<td>A list of references to Machines that are components of this System.</td>
</tr>
<tr>
<td>credentials</td>
<td>collection [Credential]</td>
<td>A list of references to Credentials that are components of this System.</td>
</tr>
<tr>
<td>volumes</td>
<td>collection [Volume]</td>
<td>A list of references Volumes that are components of this System.</td>
</tr>
<tr>
<td>networks</td>
<td>collection [Network]</td>
<td>A list of references to Network that are components of this System.</td>
</tr>
<tr>
<td>networkServices</td>
<td>collection [Network Service]</td>
<td>A reference to the NetworkServiceCollection that are components of this System.</td>
</tr>
<tr>
<td>services</td>
<td>Collection [SystemService]</td>
<td>A list of references to SystemService Resources that represent services supported by this System.</td>
</tr>
<tr>
<td>meters</td>
<td>collection [Meter]</td>
<td>A list of references to Meters monitored for this System, with component semantics. Note that these Meters are for the System and not for any individual component in the System.</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
<td>A reference to the EventLog of this System. Note that this EventLog is for the System and not for any individual component in the System.</td>
</tr>
</tbody>
</table>

When implementing or using System, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 9 as well as in the tables describing embedded Resources or related Collections.

**5.13.1.1 Attributes of type Collection**

The following clause describes the Collection Resources components of Systems.

**5.13.1.1.1 systems Collection**

The Resource type for each item of this Collection is "System". There is no accessory attribute for the items in this Collection, therefore, it is a basic System Collection, the serialization of which follows the rules in 5.5.12. See the SystemCollection Resource clause.
5.13.1.2 machines Collection

The Resource type for each item of this Collection is "Machine". There is no accessory attribute for the items in this Collection, therefore, it is a basic Machine Collection (serialized as described in 5.5.12). See the MachineCollection Resource clause.

5.13.1.3 credentials Collection

The Resource type for each item of this Collection is "Credential". There is no accessory attribute for the items in this Collection, therefore, it is a basic Credential Collection (serialized as described in 5.5.12). See the CredentialCollection Resource clause.

5.13.1.4 volumes Collection

The Resource type for each item of this Collection is "Volume". There is no accessory attribute for the items in this Collection, therefore, it is a basic Volume Collection (serialized as described in 5.5.12). See the VolumeCollection Resource clause.

5.13.1.5 networks Collection

The Resource type for each item of this Collection is "Network". There is no accessory attribute for the items in this Collection, therefore, it is a basic NetworkCollection Resource as described in clause 5.16.2.

5.13.1.6 networkServices Collection

The Resource type for each item of this Collection is "NetworkService". There is no accessory attribute for the items in this Collection, therefore, it is a basic NetworkServiceCollection as described in clause 5.16.18.

5.13.1.7 meters Collection

The Resource type for each item of this Collection is "Meter" as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore it is a basic Meter Collection (serialized as described in 5.5.12). See the MeterCollection Resource clause.

5.13.1.2 Operations

The System Resource supports the Read, Update, and Delete operations. Create is supported through the SystemCollection Resource.

The following custom operations are also defined:

start/stop/restart/pause/suspend

/link@rel: http://schemas.dmtf.org/cimi/2/action/xxx

Where "xxx" is one of "start", "stop", "restart", "pause", or "suspend".

This operation shall recursively perform the requested operation on each component of the System (Machine or sub-System). Note that not all Machines need to be in the same state for this operation to be available and the impact of this operation varies depending on the component's current state; see clause 5.14.1.2 for more details about performing operations on Machines. If the operation fails for a Machine, that Machine shall not be affected by the operation.

export

/link@rel: http://schemas.dmtf.org/cimi/2/action/export
This operation shall export a System along with all Resources component of or used by this System. If an export package exists at that URI, it is updated with the values of the System and any component management Resources. Otherwise, a new export package is created at that URI with a Media Type as specified by the "format" parameter. Other formats may be used if supported, but are not specified by this standard.

Input parameters:

1) "format" - type: string – optional. Indicates the Media Type of the exported data. If not present, the default value shall be "application/ovf."

2) "destination" - type: URI – optional. Indicates the location to where the exported data is placed. If not present, the HTTP response Location header shall contain the URL to the exported data. Based on the specific protocol specified within the URI, the Consumer might need to provide additional information (such as credentials) in the "properties" field. In the case of HTTP, a PUT shall be used to place the data at the specified location.

Output parameters: None.

HTTP protocol

To export a System, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/export" URI of the System where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/export",
  "format": string, ?
  "destination": string, ?
  "properties": { string: string, + } ?
  ...
}
```

XML media type: application/xml

XML serialization

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action>http://schemas.dmtf.org/cimi/2/action/export</action>
  <format> xs:string </format> ?
  <destination> xs:anyURI </destination> ?
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties>
  <xs:any>*
</Action>
```
5.13.2 SystemCollection Resource

A SystemCollection Resource represents a Collection of System Resources and follows the Collection pattern defined in clause 5.5.12.

5.13.2.1 Operations

NOTE The "add" operation requires that a SystemTemplate be used (see 4.2.1.1).

Resources created during the process of creating a System shall be components of the System (see 5.13.1). For example, a componentDescriptor that references a MachineTemplate, and within that MachineTemplate is a reference to a VolumeTemplate, results in a reference to the new Machine being added to the System.machines attribute and a reference to the new Volume being added to the System.volumes attribute. However, if this MachineTemplate refers to an existing Volume, this Volume shall not be added to the top-level System attributes.

The following custom operations are also defined:

import

/link@rel:http://schemas.dmtf.org/cimi/2/action/import

This operation shall import a System. Not only is a System created, but Machines, Volumes, and Networks and possibly recursive Systems and their components may also be created corresponding to imported descriptor entries. More detail about this process is in ANNEX A.

1) Input parameters: "source" - type: URI - mandatory

2) Indicates the location from which the imported data is retrieved. Based on the specific protocol specified within the URI, the Consumer might need to provide additional information (such as credentials) in the "properties" field.

Output parameters: None.

HTTP protocol

To import a System, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/import" URI of the System Collection where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
   "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
   "action": "http://schemas.dmtf.org/cimi/2/action/import",
   "source": string, ?
   "properties": { string: string, + } ?,
   ...
}
```

XML media type: application/xml

XML serialization

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
   <action> http://schemas.dmtf.org/cimi/2/action/import </action>
   <source> xs:anyURI </source> ?
</Action>
```
A SystemService Resource represents some management service for all or a subset of the Resources in a System. A SystemService Resource can define diverse types of management services and holds:

- a) Topology information about the service: a list of the Resources concerned by this management service, e.g., lists of Machines and Volumes subject to disaster recovery policy.
- b) Service-specific parameters: configuration data for the service itself.

System components may be listed under more than one SystemService Resources. For example, a Machine may be under a recovery service, while also participating into an autoscaling service.

Some examples of common services are:

- HighReliability service
- DisasterRecovery service
- Backup service
- Autoscaling service

<table>
<thead>
<tr>
<th>Name</th>
<th>SystemService</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemService">http://schemas.dmtf.org/cimi/2/SystemService</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceType</td>
<td>URI</td>
<td>Unique URI identifying this particular service. It shall be of the form: <a href="http://schemas.dmtf.org/cimi/2/SystemService/">http://schemas.dmtf.org/cimi/2/SystemService/</a>&lt;servicename&gt; where &lt;servicename&gt; is the end of the path, possibly a subpath.</td>
</tr>
<tr>
<td>machines</td>
<td>Collection[ Machine]</td>
<td>A reference to the list of references to Machines that are managed under this SystemService. The Resource item type may be a variant of Machine in case accessory attributes are added to the collection. This Resource items in this Collection are not child components of the SystemService Resource: deleting the SystemService shall not cause the deletion of the referred Machines.</td>
</tr>
<tr>
<td>volumes</td>
<td>Collection[ Volume]</td>
<td>A reference to the list of references to Volumes that are managed under this SystemService. The Resource item type may be a variant of Volume in case accessory attributes are added to the collection. This Resource items in this Collection are not child components of the SystemService Resource: deleting the SystemService shall not cause the deletion of the referred Volumes.</td>
</tr>
<tr>
<td>systems</td>
<td>collection [System]</td>
<td>A reference to the list of references to Systems or sub-Systems that are managed under this SystemService. The Resource item type may be a variant of System in case accessory attributes are added to the collection. This Resource items in this Collection are not child components of the SystemService Resource: deleting the SystemService shall not cause the deletion of the referred Systems.</td>
</tr>
</tbody>
</table>
5.13.3.1 HighReliability service Resource

This service allows for a System to recover from the failures of its Machines; the service intervenes when the Machine stops working - typically the System does not receive the Machine heartbeat anymore. This service protects from hardware and software failures, i.e., the failure of the hardware node executing the machine, or the case of a software process causing a segment violation that stops the OS services.

Table 11 – SystemService attributes for HighReliability service

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>SystemService</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemService">http://schemas.dmtf.org/cimi/2/SystemService</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceType</td>
<td>URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemService/highreliability/active">http://schemas.dmtf.org/cimi/2/SystemService/highreliability/active</a> or <a href="http://schemas.dmtf.org/cimi/2/SystemService/highreliability/passive">http://schemas.dmtf.org/cimi/2/SystemService/highreliability/passive</a></td>
</tr>
<tr>
<td>machines</td>
<td>Collection [Recoverable Machine]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>A reference to the collection of Machines in the System that are managed under this SystemService, meaning these benefit from recovery service. Adding a Machine reference to this collection means that the Machine becomes managed under this SystemService.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- If the serviceType is ending with &quot;/highreliability/active&quot;, each one of the listed Machines has a backup Machine. In case of failure, the backup Machine (referred to by the recoverableMachine collection item) shall take over.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- If the serviceType is ending with &quot;/highreliability/passive&quot;, each one of the listed Machines has an up-to-date MachineImage. In case of failure the backup Machine is created from the MachineImage and shall replace the failed Machine.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This Resource items in this Collection are not components of the SystemService Resource: deleting the SystemService does not cause the deletion of the referred Machines.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The details of the SystemService behavior (e.g., failover detection, etc.) depends on the Provider’s implementation, and can be controlled by additional parameters in the next attribute.</td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
<td>name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>networkServices</td>
<td>collection [Network Service]</td>
<td>A reference to the NetworkServiceCollection within the System that support this SystemService.</td>
</tr>
</tbody>
</table>
| heartbeat     | Integer  | Heartbeat frequency, in term of milliseconds between an heartbeat and the next.
DSP0263  Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

<table>
<thead>
<tr>
<th>Name</th>
<th>SystemService</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemService">http://schemas.dmtf.org/cimi/2/SystemService</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>replicationType</td>
<td>String</td>
<td>The kind of Machine replication status (it does not refer to the Volume Resource) allowable values are: synchronous, asynchronous, none, (same Machine, but not status alignment in order to allow the recovery in case just the status could cause failure) onlyAtClusterCreation</td>
</tr>
<tr>
<td>RPO</td>
<td>Integer</td>
<td>Recovery Point Objective (duration in minutes) in case of asynchronous replica of the disks.</td>
</tr>
</tbody>
</table>

### 5.13.3.1.1 RecoverableMachine Collection

The referred Resource type for each item of this Collection is “Machine”. However because there are accessory attributes, this is not a basic but an enhanced Machine Collection. The accessory attribute is defined in Table 12:

**Table 12 – RecoverableMachine accessory attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>RecoverableMachine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>backupmachine</td>
<td>Ref</td>
<td>An additional reference to the backup Machine in the same System, that supports the Machine referenced by this collection item.</td>
</tr>
</tbody>
</table>

### 5.13.3.1.2 Operations

The HighReliability SystemService Resource supports the Read, Update, and Delete operations. Create is supported through the SystemService Collection Resource.

Adding a machine to the collection (see the addRM operation) implies that a backup Machine shall be created and the backupmachine attribute shall be assigned to this copy (even if it is not an running Machine, but only a “passive” copy ready to be executed in case of failure). The way the backup copy is created depends on the Provider implementation, it is expected that an image of the recoverable machine is taken and from this image a new machine is created.

If the Consumer also gives the backup machine reference as input parameter, it is assumed that the backup machine is that referenced machine and no new backup machines shall be created.

A backup machine may also be added as part of the list of recoverable machines (i.e., in the “machines” collection of the System service). This amounts to defining a daisy-chain of two (or more) backup machines for the original (primary) recoverable machine subject to the system service.

The following custom operations are also defined on this SystemService Resource:

#### forceSync

/link@rel: http://schemas.dmtf.org/cimi/2/action/forceSync

This operation shall synchronize the state of a node onto its backup node, regardless of the scheduled synchronization time as dictated by the recovery policies.
The result of this operation depends on the Provider implementation and on the status of the backup Machine; typically it has effect when the backup Machine is obtained by an image copy of the recoverable Machine.

Input parameters: “node” (primary node) type: ref - mandatory.

Output parameters: None.

**swapBackup**

/link@rel: http://schemas.dmtf.org/cimi/2/action/swapBackup

This operation shall swap a Machine and its backup Machine – i.e., replace the Machine with its backup and vice versa.

Some Providers can choose to not make available this operation, not allowing the Consumer to choose which backup node turn in primary one.

Input parameters: "node" - type: ref - mandatory

A reference to the Machine to be replaced by its backup.

Output parameters: None.

**addRM**

/link@rel: http://schemas.dmtf.org/cimi/2/action/addRM

This operation adds a recoverable Machine (or RM) to the collection of recoverable Machines under this service. It adds the reference of the Machine to the machines collection of recoverable Machines, and optionally a reference to the backup Machine (accessory attribute “backupmachine”).

Input parameters: "node" (Machine to be added to the service) - type: ref – mandatory, “backup” (Machine to be used as backup) - type: ref – optional.

Output parameters: None.

**removeRM**

/link@rel: http://schemas.dmtf.org/cimi/2/action/removeRM

This operation removes a recoverable Machine (or RM) from the collection of recoverable Machines under this service. It removes the reference of the Machine from the machines collection of recoverable Machines, and discards the backup Machine.

Input parameters: "node" (Machine to be removed from the service) - type: ref – mandatory.

Output parameters: None.

### 5.13.3.2 DisasterRecovery service Resource

This service allows for a System to recover from a data center failure — by maintaining a remote, up-to-date image of the System.

Unlike the HighReliability service, which enables to define advanced recovery techniques for different error typologies, the DisasterRecovery service intervenes in the specific case of a data center failure and only implements the mechanism to re-start crashed resources on a remote data center.
On a data center failure occurrence, where other advanced approaches fail, this service guarantees
resources’ restoration, although some service-downtime will occur, i.e., there should be no expectation
from the customers that transition from one data center to another is “transparent”.

Typically the DisasterRecovery can be offered by default for every Machine, though some Providers
could activate it as an additional feature to be explicitly requested by the Consumer, or more often could
allow the consumer to choose the location of the remote datacenter; in such cases it is possible to define
a DisasterRecovery service Resource.

The attributes for the DisasterRecovery system service service are:

<table>
<thead>
<tr>
<th>Name</th>
<th>SystemService</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemService">http://schemas.dmtf.org/cimi/2/SystemService</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>serviceType</td>
<td>URI</td>
</tr>
<tr>
<td>machines</td>
<td>Collection[Machine]</td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
</tr>
<tr>
<td></td>
<td>name</td>
</tr>
<tr>
<td></td>
<td>backupData Center</td>
</tr>
<tr>
<td></td>
<td>backupCEP</td>
</tr>
<tr>
<td></td>
<td>network Services</td>
</tr>
</tbody>
</table>

5.13.3.2.1 Operations

The DisasterRecovery SystemService Resource supports the Read, Update, and Delete operations. Create is supported through the SystemService Collection Resource.

addRM

/link@rel: http://schemas.dmtf.org/cimi/2/action/addRM

This operation adds a recoverable Machine (or RM) to the collection of recoverable Machines under this service. It adds the reference of the Machine to the machines collection of recoverable Machines.

Input parameters: “node” (Machine to be added to the service) - type: ref – mandatory.
Output parameters: None.

removeRM

`/link@rel: http://schemas.dmtf.org/cimi/2/action/removeRM`

This operation removes a recoverable Machine (or RM) from the collection of recoverable Machines under this service. It removes the reference of the Machine from the `machines` collection of recoverable Machines.

Input parameters: "node" (Machine to be removed from the service) - type: ref – mandatory,

Output parameters: None.

### 5.13.4 SystemTemplate Resource

The SystemTemplate Resource contains the set of individual descriptors that are necessary to create or associate the components of a System. In practice, the Provider interprets the set of component descriptors as a set of creation (or association) operations to be executed in an order compatible with the dependencies (e.g., attachments or references between components) that are expressed between these components.

A SystemTemplate may include symbolic component references in the descriptors, used to express links between components of the resulting System. A component reference uses the "name" of the target (referred) component. For example, `<volume href="#newVolume"/>` would reference a Volume named "newVolume." The reference name – #newVolume – is replaced by the actual Resource URL in the instantiated System.

Table 14 describes the SystemTemplate attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>SystemTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemTemplate">http://schemas.dmtf.org/cimi/2/SystemTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| component Descriptors | `component Descriptor[]` | The list of component descriptors describing the components of a System instance realized from this SystemTemplate. For each component descriptor, the corresponding component is either created when a System instance is created (i.e., a child component), or simply associated with the system if it already exists.  

- In case of a child component: The component descriptor refers to a Template (either by reference or by value), and may also provide additional metadata (name, description, properties). The creation order of components is not specified in SystemTemplate; in particular the order of the component descriptors in this array is not meaningful in terms of creation order.  

- In case of an existing Resource to be added as an associated component of the System: The component descriptor refers directly to the existing Resource. |

<table>
<thead>
<tr>
<th>Name</th>
<th>componentDescriptor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Type</td>
</tr>
<tr>
<td>name</td>
<td><code>string</code></td>
</tr>
<tr>
<td>Name</td>
<td>SystemTemplate</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemTemplate">http://schemas.dmtf.org/cimi/2/SystemTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>string</td>
<td>The value of the “description” attribute that is associated with a System component created from this component descriptor.</td>
</tr>
<tr>
<td>properties</td>
<td>map</td>
<td>The key-value pairs that is associated with a System component created from this component descriptor.</td>
</tr>
<tr>
<td>type</td>
<td>URI</td>
<td>The TypeURI of the component to be created from this component descriptor, e.g., for a Machine: <a href="http://schemas.dmtf.org/cimi/2/Machine">http://schemas.dmtf.org/cimi/2/Machine</a></td>
</tr>
</tbody>
</table>
| <component Resource> | <any> | The exact name of this attribute varies depending on the type of Resource being created or added. This attribute shall contain one of these options:  
  - A Template that is provided inline. Such an embedded Template may contain component references, each one of which shall resolve to the URI of a component with same name once created from this SystemTemplate. In such a case, the attribute name is same as the Template type name, with the first letter in lowercase (e.g., machineTemplate).  
  - A reference to an externally defined Template. Some attribute name/value pairs may be added inside the componentTemplate element to override similar attributes in the referred Template (as described in 4.2.1.1). This example shows how component references can be added to an external Template. The attribute name is same as the Template type name, with the first letter in lowercase (e.g., machineTemplate).  
  
Example (JSON):  
```json  
"machineTemplate": {  
  "href":  
  "http://example.com/machineTemplates/72000",  
  "credential": { "href": "#MyCredential"  
  }  
}  
```

Note: The “credential” attribute in this example assumes that there is another componentDescriptor item named “MyCredential” of type “Credential” in the SystemTemplate. It shall set or override similar attribute in the referred MachineTemplate if instantiating the Machine component.  
  - A reference to an existing Resource to become associated component of the System. The attribute name is same as the Resource type name, with the first letter in lowercase (e.g., machine).
<table>
<thead>
<tr>
<th>Name</th>
<th>SystemTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/SystemTemplate">http://schemas.dmtf.org/cimi/2/SystemTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>quantity</td>
<td>integer</td>
<td>The number of component instances to be created from this component descriptor, if a template. By default, this number is equal to 1. If the value is 2 or more, the actual name assigned to each instance is the &quot;name&quot; value concatenated with a sequential number (e.g., if name=&quot;mymachine&quot;, and quantity=3, the names are: mymachine1, mymachine2, mymachine3.)</td>
</tr>
<tr>
<td>serviceDescriptors</td>
<td>serviceDescriptor[]</td>
<td>The list of service descriptors for the services to be supported by a System instance realized from this SystemTemplate. For each service descriptor, the corresponding SystemService is created when a System instance is created. The names of the System components subject to the service are listed using the symbolic component reference notation previously described (&quot;#&lt;name&gt;&quot;).</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Data</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The value of the &quot;name&quot; attribute that is associated with a SystemService instance created from this service descriptor.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The value of the &quot;description&quot; attribute that is associated with a SystemService instance created from this service descriptor.</td>
</tr>
<tr>
<td>properties</td>
<td>map</td>
<td>The key-value pairs that is associated with a SystemService instance created from this service descriptor.</td>
</tr>
<tr>
<td>serviceType</td>
<td>URI</td>
<td>The serviceType of the service to be created from this service descriptor, e.g., for a SystemService of type &quot;DisasterRecovery&quot;: <a href="http://schemas.dmtf.org/cimi/2/SystemService/disasterrecovery">http://schemas.dmtf.org/cimi/2/SystemService/disasterrecovery</a></td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
<td>This is where additional service-specific attributes are listed (see clause 5.13.6).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>MeterTemplate[]</th>
</tr>
</thead>
</table>

| Meter Templates | MeterTemplate[] | A list of references to MeterTemplates that shall be used to create and connect a set of new Meters to the new System. |

Note that the attributes of the MeterTemplate may be specified rather than a reference to an existing MeterTemplate Resource.

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

| eventLog Template | ref | A reference to an EventLogTemplate that shall be used to create and connect a new EventLog to the new System. |

Note that the attributes of the EventLogTemplate may be specified rather than a reference to an existing EventLogTemplate Resource.

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

| import Image | URI |

If the Template is the result of an import – e.g., of an OVF package - this attribute should be used. If present, it shall reference the import source (e.g., OVF package) used to create this Template.
When implementing or using SystemTemplate, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 14 as well as in the tables describing embedded Resources or related Collections.

### 5.13.4.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the SystemTemplateCollection Resource.

The following custom operations are also defined:

**export**

/link@rel: http://schemas.dmtf.org/cimi/2/action/export

This operation shall export a SystemTemplate along with all its component Resources as well as the used Resources that are listed in its top-level Collections. If an export package exists at that URI, it is updated with the values of the SystemTemplate and any component management Resources. Otherwise a new export package is created at that URI with a Media Type as specified by the "format" parameter. Other formats may be used if supported, but are not specified by this standard.

**Input parameters:**

1) "format" - type: string - optional. Indicates the Media Type of the exported data. If not present, the default value shall be "application/ovf."

2) "destination" - type: URI – optional. Indicates the location to where the exported data is placed. If not present, the HTTP response Location header shall contain the URL to the exported data. Based on the specific protocol specified within the URI, the Consumer might need to provide additional information (such as credentials) in the "properties" field. In the case of HTTP, a PUT shall be used to place the data at the specified location.

**Output parameters:** None.

**HTTP protocol**

To export a SystemTemplate, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/export" URI of the SystemTemplate where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/export",
    "format": string, ?
    "destination": string, ?
    "properties": { string: string, + } ?
}```
5.13.5 SystemTemplateCollection Resource

A SystemTemplateCollection Resource represents the Collection of SystemTemplate Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.13.5.1 Operations

The following custom operations are defined:

import

/link@rel: http://schemas.dmtf.org/cimi/2/action/import

This operation shall import a SystemTemplate. Not only is a SystemTemplate created, but MachineTemplates, VolumeTemplates, and NetworkTemplates and possibly recursive SystemTemplates and their components may also be created, corresponding to imported descriptor entries. More detail about this process is in ANNEX A.

Input parameters:
1) "source" - type: URI - mandatory.
2) Indicates the location from which the imported data is retrieved. Based on the specific protocol specified within the URI, the Consumer might need to provide additional information (such as credentials) in the "properties" field.

Output parameters: None.

HTTP protocol

To import a SystemTemplate, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/import" URI of the SystemTemplateCollection where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/import",
}
```
5.13.6 Service-specific Descriptor attributes

This clause defines the additional attributes specific to each service type that need be added to a serviceDescriptor for this service type in the SystemTemplate.

5.13.6.1 Parameters for the HighReliability service type

Service type: http://schemas.dmtf.org/cimi/2/SystemService/highreliability

Table 15 – Additional parameters for HighReliability service

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>machines</td>
<td>String[]</td>
<td>Symbolic references to the Machine components in the System that are subject to the service. Uses the symbolic component reference notation previously described (“#&lt;name&gt;”).</td>
</tr>
<tr>
<td>network</td>
<td>string</td>
<td>Symbolic reference to the Network Resource in the System that enables this service. The Network shall provide the necessary connections between Machines to support this Service.</td>
</tr>
<tr>
<td>heartbeat</td>
<td>Integer</td>
<td>Heartbeat frequency, in term of milliseecs between an heartbeat and the next.</td>
</tr>
<tr>
<td>replicationType</td>
<td>String</td>
<td>The kind of disk replication data (it does not refer to the Volume Resource) allowable values are: synchronous, asynchronous, none, onlyAtClusterCreation</td>
</tr>
<tr>
<td>RPO</td>
<td>Integer</td>
<td>Recovery Point Objective (duration in minutes) in case of asynchronous replica of the disks.</td>
</tr>
</tbody>
</table>

5.14 Machine Resources and relationships

5.14.1 Machine

An instantiated compute Resource that encapsulates both CPU and Memory. Table 16 describes the Machine attributes.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Description</th>
</tr>
</thead>
</table>
| state    | http://schemas.dmtf.org/cimi/2/Machine      | The operational state of the Machine. Allowed values are:  
|          |                                              | CREATING: The Machine is in the process of being created.  
|          |                                              | STARTING: The Machine is in the process of being started.  
|          |                                              | STARTED: The Machine is available and ready for use.  
|          |                                              | STOPPING: The Machine is in the process of being stopped.  
|          |                                              | STOPPED: This value is the virtual equivalent of powering off a physical Machine. There is no saved CPU or memory state. Clause 5.14.1.2 defines the initial state of a Machine.  
|          |                                              | PAUSING: The Machine is in the process of being PAUSED.  
|          |                                              | PAUSED: In this state the Machine and its virtual resources remain instantiated and resources remain allocated, similar to the "STARTED" state, but the Machine and its virtual resources are not enabled to perform tasks. This is equivalent to a "stand-by" state.  
|          |                                              | SUSPENDING: The Machine is in the process of being suspended.  
|          |                                              | SUSPENDED: In this state the Machine and its virtual resources are stored on nonvolatile storage. The Machine and its resources are not enabled to perform tasks.  
|          |                                              | CAPTURE: If the Machine is undergoing the "capture" operation its state may be set to "CAPTURE". If some operations that were accepted by the Machine before the capture are no longer available during the capture, the Machine shall be in the CAPTURE state.  
|          |                                              | RESTORING: The Machine is in the process of being restored from a MachineImage.  
|          |                                              | DELETING: The Machine is in the process of being deleted.  
|          |                                              | ERROR: The Provider has detected an error in the Machine.  
|          |                                              | FAILED: The Machine is not operational due to some error condition and in accordance to the Provider's policies it is considered failed. This state calls for a recovery procedure, if any.  
<p>|          |                                              | The operations that result in transitions to the above defined states are defined in clause 5.14.1.2.                                                                                                                                                                                                                                                                                                                                                             |
| cpu      | integer                                      | The amount of CPU that this Machine has.                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| memory   | integer                                      | The size of the memory (RAM) in kibibytes allocated to this Machine. If this value is increased, it implies that the Machine is allocated more RAM, and vice versa if the value is decreased.                                                                                                                                                                                                                                                                                                      |
| disks    | collection [Disk]                            | A reference to the list of disks (local storage) that are part of the Machine. Adding an element to this list creates a disk. The Disk Resources are components of the Machine. Note: The Disk Resource type is defined in clause 5.14.1.1.1.                                                                                                                                                                                                                           |
| cpuArch  | string                                       | The CPU architecture that is supported by Machines created by using this configuration. Allowed values are: 68000, Alpha, ARM, Itanium, MIPS, PA_RISC, POWER, PowerPC, x86, x86_64, z/Architecture, SPARC. Providers may define additional values.                                                                                                                                                                                                                                              |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Machine</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Machine">http://schemas.dmtf.org/cimi/2/Machine</a></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>cpuSpeed</td>
<td>integer</td>
<td>The approximate CPU speed of this Machine - in megahertz.</td>
</tr>
</tbody>
</table>
| volumes              | collection [located Volume]        | A reference to the list of references to Volumes that are connected 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.  
Note: This Collection has the semantics of usage of the Volumes by the Machine (deleting the Machine does not cause the deletion of the referred Volumes). It is defined in clause 5.14.1.1.2. |
| latestSnapshot       | ref                                | A reference to the SNAPSHOT representing the latest state captured for this Machine (either the most recent Snapshot or the last Snapshot reverted to).  
Constraints:  
Provider: support optional; mutable  
Consumer: support optional; read-only |
| snapshots            | collection [Machine Image]         | A reference to the list of references to the MachineImages of type SNAPSHOT taken of this Machine. This Collection has the semantics of usage of SNAPSHOT MachineImages by the Machine (The deletion of the Machine does not cause the deletion of the referred Snapshots.) |
| meters               | collection [Meter]                 | A reference to the list of Meters monitored for this Machine.                                                                                                                                                |
| eventLog             | ref                                | A reference to the EventLog of this Machine.                                                                                                                                                                 |

When implementing or using Machine, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 16, as well as in the tables describing embedded Resources or related Collections.

**5.14.1.1 Collections**

The following clause describes the Collection Resources components of Machines.

**5.14.1.1.1 Disk Collection**

The Resource type for each item of this Collection is "Disk", defined in Table 17:

**Table 17 – Disk attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Disk</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Disk">http://schemas.dmtf.org/cimi/2/Disk</a></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>capacity</td>
<td>integer</td>
<td>The initial capacity, in kilobytes, of the disk.</td>
</tr>
<tr>
<td>Name</td>
<td>Disk</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Disk">http://schemas.dmtf.org/cimi/2/Disk</a></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>initialLocation</td>
<td>string</td>
<td>Operating System-specific location (path) in its namespace where this disk first appears. After deployment, Consumers may consider moving the location of this Disk. Support of this attribute indicates that the Provider can report this information back to the Consumer.</td>
</tr>
</tbody>
</table>

5.14.1.1.2 volumes Collection

The referred Resource type for each item of this Collection is “Volume”. However because there is an accessory attribute (initialLocation), this is not a basic but an enhanced Volume Collection. The name “locatedVolume” is used to define the type of each Collection item. The accessory attribute is defined in Table 18:

<table>
<thead>
<tr>
<th>Name</th>
<th>locatedVolume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/locatedVolume">http://schemas.dmtf.org/cimi/2/locatedVolume</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>initialLocation</td>
<td>string</td>
</tr>
</tbody>
</table>

The resourceURI attribute value for the Collection of locatedVolume items is:

http://schemas.dmtf.org/cimi/2/locatedVolumeCollection.

5.14.1.1.3 interfaces Collection

The Resource type for each item of this Collection is “NetworkInterface”, defined in clause 5.16.13. The Collection is a basic NetworkInterfaceCollection as described in clause 5.16.14.

5.14.1.1.4 snapshots Collection

The Resource type for each item of this Collection is “MachineImage”. It is a basic MachineImage Collection. Its serialization is described in the MachineImageCollection Resource clause.

5.14.1.1.5 meters Collection

The Resource type for each item of this Collection is “Meter” as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic Meter Collection (serialized as described in 5.5.12). See the MeterCollection Resource clause.

5.14.1.2 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the MachineCollection Resource.

The following custom operations are also defined:

start

/link@rel: http://schemas.dmtf.org/cimi/2/action/start
This operation shall start a Machine.

Input parameters: None.
Output parameters: None.

During the processing of this operation, the Machine shall be in the “STARTING” state.

Upon successful completion of this operation, the Machine shall be in the “STARTED” state.

If a Machine is in the “STOPPED” state, starting it shall be the virtual equivalent of powering on a physical machine. There is no restored CPU or Memory state, so the guest OS typically performs boot or installation tasks.

If the Machine was in the “SUSPENDED” or “PAUSED” state, starting it shall have the effect of resuming it.

**HTTP protocol**

To start a Machine, a POST is sent to the “http://schemas.dmtf.org/cimi/2/action/start” URI of the Machine where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/start",
    "properties": { string: string, + } ?

    ...
}
```

**XML media type:** application/xml

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/start </action>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
    <xs:any>*
</Action>
```

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

Stop

```
/link@rel: http://schemas.dmtf.org/cimi/2/action/stop
```

This operation shall stop a Machine.
Input parameters:

1) "force" - type: boolean - optional.
2) A flag to indicate whether the Provider shall simulate a power off condition (force=true) or shall simulate a shutdown operation that allows applications to save their state and the file system to be made consistent (force=false). Inclusion of this parameter by Consumers is optional and if not specified, the Provider may choose either mechanism. Providers are encouraged to advertise this choice by way of the MachineStopForceDefault capability.

Output parameters: None.

During the processing of this operation, the Machine shall be in the "STOPPING" state.

Upon successful completion of this operation, the Machine shall be in the "STOPPED" state. Stopping a Machine with force=true shall be the virtual equivalent of powering off a physical machine. There is no saved CPU or Memory state. Stopping a Machine with force=false shall result in a machine with consistent file systems.

A Consumer may re-issue a stop operation if the state is STOPPING, perhaps with force=true, but Providers shall not issue a force=true stop operation on their own.

HTTP protocol

To stop a Machine, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/stop" URI of the Machine where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/stop",
    "force": boolean, ?
    "properties": { string: string, + } ?
    ...
}
```

XML media type: application/xml

XML serialization:

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">

    <action> http://schemas.dmtf.org/cimi/2/action/stop </action>

    <force> xs:boolean </force> ?

    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>

    <xs:any>*

</Action>
```

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

/link@rel: http://schemas.dmtf.org/cimi/2/action/restart

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

Input parameters:

1) "force" - type: boolean - optional.
2) A flag to indicate whether the Provider shall simulate a power off condition (force=true) or shall
simulate a shutdown operation that allows applications to save their state and the file system to
be made consistent (force=false). Inclusion of this parameter by Consumers is optional and if
not specified, the Provider may choose either mechanism. Providers are encouraged to
advertise this choice by way of the MachineStopForceDefault capability.

Output parameters: None.

During the processing of this operation, the Machine shall be in the "STOPPING" or "STARTING" states,
as appropriate depending on its initial state.

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

HTTP protocol

To restart a Machine, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/restart" URI of the
Machine where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/restart",
    "force": boolean, ?
    "properties": { string: string, + } ?
    ...
}
```

**XML media type:** application/xml

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/restart </action>
    <force> xs:boolean </force> ?
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
    <xs:any>*
</Action>
```
Upon successful processing of the request, the HTTP response body may be empty.

**pause**

/link@rel: http://schemas.dmtf.org/cimi/2/action/pause

This operation shall pause a Machine.

Input parameters: None.

Output parameters: None.

During the processing of this operation, the Machine shall be in the "PAUSING" state.

Upon successful completion of this operation, the Machine shall be in the "PAUSED" state. Pausing a Machine shall keep the Machine and its resources instantiated, but the Machine shall not be available to perform any tasks. The current state of the CPU and Memory shall be retained in volatile memory.

**HTTP protocol**

To pause a Machine, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/pause" URI of the Machine where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/pause",
    "properties": { "string": string, + } ?
...
}
```

**XML media type:** application/xml

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/pause </action>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
    <xs:any>*
</Action>
```

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

**suspend**

/link@rel: http://schemas.dmtf.org/cimi/2/action/suspend

This operation shall suspend a Machine.

Input parameters: None.

Output parameters: None.
During the processing of this operation, the Machine shall be in the "SUSPENDING" state.

Upon successful completion of this operation, the Machine shall be in the "SUSPENDED" state.

Suspending a Machine shall keep the Machine and its resources instantiated, but the Machine shall not be available to perform any tasks. The current state of the CPU and Memory shall be retained in non-volatile memory.

HTTP protocol

To suspend a Machine, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/suspend" URI of the Machine where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/suspend",
  "properties": { string: string, + } ? ...
}
```

XML media type: application/xml

XML serialization:

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/suspend </action>
  <properties>
    <property key="xs:string"> xs:string </property>*
  </properties>
  <xs:any>*
</Action>
```

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

capture

/link@rel: http://schemas.dmtf.org/cimi/2/action/capture

This operation shall create a new MachineImage from an existing Machine. This operation is defined within the MachineImage Resource; see 5.14.7.1 for more details. Note that while this operation is performed against a MachineImage, its presence in the Machine serialization is used to advertise support for the operation.

Snapshotting a Machine

/link@rel: http://schemas.dmtf.org/cimi/2/action/snapshot

This operation shall create a new SNAPSHOT MachineImage from an existing Machine. This operation is defined within the MachineImage Resource; see 5.14.7.1 for more details. Note that while this operation is performed against a MachineImage, its presence in the Machine serialization is used to advertise support for the operation.
**Restoring a Machine**

`/link@rel: http://schemas.dmtf.org/cimi/2/action/restore`

This operation shall restore a Machine from a previously created MachineImage.

**Input parameters:**

1) "image" - type: URI - mandatory.
2) A reference to the Machine Image.

**Output parameters:** None.

During the processing of this operation, the Machine shall be in the "RESTORING" state.

Upon successful completion of this operation, the Machine shall be in the same state as the state specified in the MachineImage, if specified. See 5.14.1.2 for more details.

Note that Providers can indicate support for restoring from non-KEY MachineImages by way of the Machine "RestoreFromImage" capability. If the RestoreFromImage capability is not supported, and the restore operation is supported, the restore operation can only restore from a SNAPSHOT MachineImage.

**HTTP protocol**

To restore a Machine, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/restore" URI of the Machine where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/restore",
    "image": { "href": string },
    "properties": { string: string, + } ?
    ...
}
```

**XML media type:** application/xml

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/restore </action>
    <image href="xs:anyURI"/>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
    <xs:any>*
</Action>
```

Where the "image" URI is a reference to the MachineImage to be used.

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

/link@rel: http://schemas.dmtf.org/cimi/2/action/connectvolume

This operation shall start a Machine.

Input parameters: Volume reference, initialLocation, Credentials, properties. The properties capture Provider-specific options for the operation,

Output parameters: None.

HTTP protocol

To connect a Volume to a Machine, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/connectvolume" URI of the Machine where the HTTP request body shall be as described below.

**JSON media type: application/json**

**JSON serialization:**

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/connectvolume",
    "volume": { "href": "string" },
    "initialLocation": "string",
    "credentials": { "href": "string" },
    "properties": { string: string, }?
}
```

**XML media type: application/xml**

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/connectvolume</action>
    <volume href="xs:anyURI"/>
    <initialLocation>xs:string</initialLocation>
    <credentials href="xs:anyURI"/>
    <action> http://schemas.dmtf.org/cimi/2/action/connectvolume</action>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
    <xs:any>*
</Action>
```

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

**5.14.2 MachineCollection Resource**

A MachineCollection Resource represents the Collection of Machine Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.
5.14.2.1 Operations

NOTE The "add" operation requires that a MachineTemplate be used (see 4.2.1.1).

Upon successful processing of the "add" operation, unless otherwise specified by way of the MachineTemplate "initialState" attribute, the state of the new Machine shall be the value of the DefaultInitialState capability, if defined. If no DefaultInitialState capability is defined, the default value shall be "STOPPED." The semantics of "initialState" shall be equivalent to the Provider issuing the appropriate actions against the new Machine to move it into that state. Note that this controls the actions of the hypervisor and the state of the resources within the Machine (e.g., the operating system) are also influenced by the data within the MachineImage used to create the new Machine. For example, if a new Machine's initialState is "STARTED" and a SNAPSHOT MachineImage was used to create the new Machine, the Machine would not be "booted" but rather resume executing from the saved state in the MachineImage.

If a Provider is unable to change the state of the new Machine to the appropriate "initialState" (either as specified by the MachineTemplate or as implied by the previous stated rules), the Machine creation shall fail.

If a Provider is unable to create the new Machine due to invalid or inconsistent credentials in the MachineTemplate, the Machine creation process shall fail. If any credentials are included in the MachineTemplate, they shall be part of the new Machine regardless of the type of MachineImage used.

5.14.3 MachineTemplate

A MachineTemplate represents the set of metadata and instructions used in the creation of a Machine. Table 19 describes the MachineTemplate attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/MachineTemplate">http://schemas.dmtf.org/cimi/2/MachineTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>initialState</td>
<td>string</td>
</tr>
<tr>
<td>machineConfig</td>
<td>ref</td>
</tr>
<tr>
<td>machineImage</td>
<td>ref</td>
</tr>
<tr>
<td>credential</td>
<td>ref</td>
</tr>
</tbody>
</table>
### Name: MachineTemplate

<table>
<thead>
<tr>
<th>Type</th>
<th>URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://schemas.dmtf.org/cimi/2/MachineTemplate">http://schemas.dmtf.org/cimi/2/MachineTemplate</a></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Attribute: volumes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volses</td>
<td>volume[]</td>
<td>A list of structures, each containing a reference to an existing Volume and potentially describing aspects of the way that the given Volume is to be connected to the Machine during its creation from this MachineTemplate. Each volume structure has the following attributes:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>initialLocation</td>
<td>string</td>
</tr>
<tr>
<td>credential</td>
<td>ref</td>
</tr>
<tr>
<td>volume</td>
<td>ref</td>
</tr>
</tbody>
</table>

#### Attribute: volumeTemplates

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>volumeTemplates</td>
<td>volumeTemplate[]</td>
<td>A list of structures, each containing a reference to a VolumeTemplate from which a Volume is created and connected to the Machine resulting from this MachineTemplate. Each structure can potentially also include aspects of the way in which each created Volume is connected to the created Machine. Credentials associated with the new Volume are same as the Credentials for this Machine. If the Machine is created as part of a System creation, the Volumes created from these Templates are considered as part of that System without the need for these VolumeTemplates to also be listed in the volumeTemplates attribute of the relevant SystemTemplate. If the same VolumeTemplate reference is listed in both the volumeTemplates attribute of a SystemTemplate and in the volumeTemplates attribute of a MachineTemplate component of that SystemTemplate, this means that multiple, distinct Volume instances are created as part of the overall System creation. Each volumeTemplate structure has the following attributes:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>volumeTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>initialLocation</td>
<td>string</td>
</tr>
<tr>
<td>volumeTemplate</td>
<td>ref</td>
</tr>
</tbody>
</table>
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkInterfaceTemplate[]</th>
</tr>
</thead>
</table>
| **interfaceTemplates** | A list of references to `NetworkInterfaceTemplate[]` that shall be used to create a new set of `NetworkInterfaceTemplate` Resources for the new Machine.  
Note that the attributes of a `NetworkInterfaceTemplate` may be given instead of a reference to an existing `NetworkInterfaceTemplate` Resource. |

<table>
<thead>
<tr>
<th>Name</th>
<th>string</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>userData</strong></td>
<td>A Base64 encoded string whose decoded version is to be injected into Machines created by using this Template. See the discussion of <em>injection of user-defined data</em> below.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>meterTemplates[]</th>
</tr>
</thead>
</table>
| **meterTemplates** | A list of references to `MeterTemplate` that shall be used to create and connect a set of new `MeterTemplate` Resources to the new Machine.  
Note that the attributes of the `MeterTemplate` may be specified rather than a reference to an existing `MeterTemplate` Resource. |

<table>
<thead>
<tr>
<th>Name</th>
<th>ref</th>
</tr>
</thead>
</table>
| **eventLogTemplate** | A reference to an `EventLogTemplate` that shall be used to create and connect a new `EventLogTemplate` Resource to the new Machine.  
Note that the attributes of the `EventLogTemplate` may be specified rather than a reference to an existing `EventLogTemplate` Resource. |

<table>
<thead>
<tr>
<th>Name</th>
<th>ResourceMetadata</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>genResourceMetadata</strong></td>
<td>A reference to a <code>ResourceMetadata</code> that shall be associated with every Machine generated from this Template.</td>
</tr>
</tbody>
</table>

When implementing or using `MachineTemplate`, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 19, as well as in the tables describing embedded Resources or related Collections.

**Injection of user-defined data**

To simplify the customization of individual Machines, it is possible to pass arbitrary data into the new Machine by using the `userData` parameter. The value of this parameter shall be the Base64-encoded payload. The Provider shall arrange for this data to be available from inside the Machine by using one of the following methods:

- **Metadata server**: The data can be retrieved from within the instance by using an HTTP GET request to `http://169.254.169.254/cimi/latest/user-data`.

- **Disk**: The Machine has access to a Disk with an ISO 9660 file system on it. The data can be found in a file at `<location>/cimi/user-data`.

- **Image modification**: The Provider modifies the root file system of the machine image just before launching the Machine. In UNIX-like operating systems, the data can be found in the file `/var/lib/cimi/user-data`.

It is strongly recommended that Providers implement a metadata server, or, failing that, injection by way of Disk, as image modification may not work for every operating system in use. The Provider shall indicate which of these three methods is supported with the Machine 'UserData' capability in the `ResourceMetadata` for Machines. The value for this feature shall be one of metadata, disk, or imgmod, corresponding to the three methods listed above.

The Provider shall preserve this data across restarts of the Machine. The data is the Base64-decoded version of the data that was passed into the `MachineCreate` request.
5.14.3.1 Operations
This Resource supports the Read, Update, and Delete operations. Create is supported through the MachineTemplateCollection Resource.

5.14.4 MachineTemplateCollection Resource
A MachineTemplateCollection Resource represents the Collection of MachineTemplate Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.14.4.1 Operations
This Resource supports the Read and Update operations. Creation of new MachineTemplate Resources is supported by way of a POST to the "add" operation’s URI as described in clause 4.2.1.1.

5.14.5 MachineConfiguration Resource
The MachineConfiguration Resource represents the set of configuration values that define the (virtual) hardware resources of a to-be-realized Machine Instance. MachineConfigurations are created by Providers and may, at the Providers discretion, be created by Consumers.

Table 20 describes the MachineConfiguration attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>MachineConfiguration</th>
<th>Type URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:cpu</td>
<td>integer</td>
<td>The amount of CPU that a Machine realized from this configuration.</td>
</tr>
<tr>
<td>Name:memory</td>
<td>integer</td>
<td>The amount of RAM, in kibibytes, that a Machine realized from this configuration.</td>
</tr>
<tr>
<td>Name:disks</td>
<td>disk[]</td>
<td>A list of structures, each containing the attributes defining the disks to be created for the Machine instantiated with this MachineConfiguration Resource. The disks are local storage to the Machine. Each disks attribute has the following subattributes:</td>
</tr>
<tr>
<td>Name:capacity</td>
<td>integer</td>
<td>The initial capacity, in kilobytes, of the disk described by this attribute.</td>
</tr>
<tr>
<td>Name:format</td>
<td>string</td>
<td>The format/type of this disk (e.g., ext4, NTFS).</td>
</tr>
<tr>
<td>Name:initialLocation</td>
<td>string</td>
<td>An Operating System-specific location (path) in its namespace where this Disk first appears. After creation of a Machine, Consumers may change the location of this Disk.</td>
</tr>
<tr>
<td>Name:cpuArch</td>
<td>string</td>
<td>The CPU architecture that is supported by Machines created by using this configuration. Allowed values are: 68000, Alpha, ARM, Itanium, MIPS, PA_RISC, POWER, PowerPC, x86, x86_64, z/Architecture, SPARC. Providers may define additional values.</td>
</tr>
<tr>
<td>Name:cpuSpeed</td>
<td>integer</td>
<td>The approximate CPU speed of this Machine in megahertz.</td>
</tr>
</tbody>
</table>
NOTE The disk attributes "format" does not appear on Machine Resources because after the Machine is created, the user of the Machine is able modify this attribute of a disk, possibly without the Provider's knowledge. Therefore these attributes might not be an aspect of the Machine that the Provider can reliably manage.

5.14.5.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the MachineConfigurationCollection Resource.

5.14.6 MachineConfigurationCollection Resource

A MachineConfigurationCollection Resource represents the Collection of MachineConfiguration Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.14.6.1 Operations

This Resource supports the Read and Update operations. Creation of new MachineConfiguration Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

5.14.7 MachineImage Resource

This Resource represents the information necessary for hardware virtualized Resources to create a Machine Instance; it contains configuration data such as startup instructions, including possible combinations of the following items, depending on the "type" of MachineImage created:

- The software image (i.e., a copy of an installed Machine), that is to be instantiated on the disk and other virtual resources. The image can be a snapshot that consists of disk images plus memory and other resource state information.
- 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 to install new components not included in the original disk image.

Table 21 describes the MachineImage attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>MachineImage</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/MachineImage">http://schemas.dmtf.org/cimi/2/MachineImage</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>The operational state of the MachineImage. Allowed values are: CREATING: The MachineImage is in the process of being created. AVAILABLE: The MachineImage is available and ready for use. Unless otherwise specified, the MachineImage shall initially be in this state after successful creation. DELETING: The MachineImage is in the process of being deleted. ERROR: The Provider has detected an error in the MachineImage. The operations that result in transitions to the above defined states are defined in clause 5.14.7.1</td>
</tr>
</tbody>
</table>
**Name** | MachineImage  
---|---
**Type URI** | http://schemas.dmtf.org/cimi/2/MachineImage  

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| type | string | The type of MachineImage that is represented by this Resource. This specification defines the following values:  
**IMAGE**: This type represents the persisted data of a stopped Machine. Unlike "snapshots", it does not contain any runtime information. If this value is used, the "relatedImage" attribute shall not be present.  
**SNAPSHOT**: This type represents the persisted data of a Machine. If the Machine was not in a stopped state when this Image was created, it also contains runtime information. If this value is used, the "relatedImage" attribute shall reference the most recently created (or reverted to) snapshot Image for that Machine, which allows for easy discovery of the "previous" snapshot. The "relatedImage" attribute shall not be set by Consumers.  
**PARTIAL_SNAPSHOT**: This type follows the same semantics as the "SNAPSHOT" MachineImage except that it contains just the changes (deltas) made to the Machine based on the referenced "relatedImage" MachineImage rather than a complete representation of the Machine.  
If a MachineImage is deleted, the following semantics shall apply:  
- Any "SNAPSHOT" MachineImages that have a "relatedImage" value that references the deleted MachineImage shall have that value changed to the "relatedImage" attribute of the delete MachineImage.  
- Any "PARTIAL_SNAPSHOT" MachineImages that have a "relatedImage" value that references the deleted MachineImage shall also be deleted. This detail applies recursively to any subsequent "PARTIAL_SNAPSHOT" MachineImages as well. |
| imageLocation | URI | A reference to the location of the binary data that makes up this image. |
| relatedImage | ref | A reference to another MachineImage Resource that is related to this one. The specific meaning of this value varies depending on the type of MachineImage. |

### 5.14.7.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the MachineImageCollection Resource.

If creating a new MachineImage, the representation of the new MachineImage may include a reference in the "imageLocation" attribute. Providers shall inspect this reference (most likely by way of an HTTP HEAD) to determine if any special processing is required. This specification defines the following additional steps that Providers shall take depending on the type of Resource being referenced:

- If the "imageLocation" is a reference to a Machine, the Provider shall create a new MachineImage based on the Machine being referenced. The machine is captured or snapshotted, depending on whether the request was sent to the "http://schemas.dmtf.org/cimi/2/action/capture" or the "http://schemas.dmtf.org/cimi/2/action/snapshot" URI of the Machine. However the resulting resource, although linked to the Machine from which it was originated, shall be a MachineImage for all purposes and can be used for creating new machines.

- If creating a SNAPSHOT, and upon completion of the create operation, the MachineImage's "imageLocation" attribute shall not reference the Machine (as the Machine might change over time), but instead it shall reference (or contain the data of) the static representation of the Machine. Additionally,
the referenced Machine's MachineSnapshotCollection shall be updated to include a reference to this newly created SNAPSHOT MachineImage Resource. If the Machine is unable to accept operations at any point while it is being captured to create the MachineImage, the Machine shall be in the CAPTURING state.

5.14.8 MachineImageCollection Resource

A MachineImageCollection Resource represents the Collection of MachineImage Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.14.8.1 Operations

This Resource supports the Read and Update operations. Creation of new MachineImage Resources is supported by way of a POST to the “add” operation's URI as described in clause 4.2.1.1, where the request body and the way it is processed are described in clause 5.14.7.1.

5.14.9 Credential Resource

A Credential Resource contains the information required to create the initial administrative superuser of a newly created Machine or to represent the credentials needed to perform some operation. 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 Resource with additional attributes to meet their requirements.

For example, a Provider might extend this Resource 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 22 describes the Credential attributes.

Table 22 – Credential attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><a href="http://schemas.dmtf.org/cimi/2/Credential">http://schemas.dmtf.org/cimi/2/Credential</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameters</td>
<td>map</td>
<td>A list of attributes that are specific to this Provider.</td>
</tr>
</tbody>
</table>

Some common extension attributes that Providers might use include:

Table 23 – UserName/Password attributes

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

Table 24 – Public key attributes

<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>

When implementing or using Credential, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in table 22, as well as in the table describing related Collections.
5.14.9.1 Operations
This Resource supports the Read, Update, and Delete operations. Create is supported through the CredentialCollection Resource.

5.14.10 CredentialCollection Resource
A CredentialCollection Resource represents the Collection of Credential Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.14.10.1 Operations
NOTE The "add" operation requires that a CredentialTemplate be used (see 4.2.1.1).

5.14.11 CredentialTemplate Resource
This Resource captures the configuration values for realizing a Credential Resource. A CredentialTemplate may be used to create multiple Credentials. Table 25 describes the CredentialTemplate attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>CredentialTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/CredentialTemplate">http://schemas.dmtf.org/cimi/2/CredentialTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
</tr>
</tbody>
</table>

When implementing or using CredentialTemplate, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 25 as well as in the table describing related Collections.

5.14.11.1 Operations
This Resource supports the Read, Update, and Delete operations. Create is supported through the CredentialTemplateCollection Resource.

5.14.12 CredentialTemplateCollection Resource
A CredentialTemplateCollection Resource represents the Collection of CredentialTemplate Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.14.12.1 Operations
This Resource supports the Read and Update operations. Creation of new CredentialTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

5.15 Volume Resources and relationships

5.15.1 Volume
A Volume represents storage at either the block or the file-system level. Volumes can be connected to Machines. Once connected, Volumes can be accessed by processes on that Machine. Table 26 describes the Volume attributes.
Table 26 – Volume attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Volume">http://schemas.dmtf.org/cimi/2/Volume</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| state     | string   | The operational state of the Volume. Allowed values are:  
CREATING: The Volume is in the process of being created.  
AVAILABLE: The Volume is available and ready for use. Unless otherwise specified, the Volume shall be in this state initially after successful creation.  
CAPTURING: The Volume is in the process of being captured (snapshotted) into a new VolumeImage.  
RESTORING: The Volume is in the process of being restored.  
DELETING: The Volume is in the process of being deleted.  
ERROR: The Provider has detected an error in the Volume. The operations that result in transitions to the above defined states are defined in clause 5.15.1.2 |
| type      | URI      | A URI that indicates the type of Volume to be created. This specification defines the following URI:  
http://schemas.dmtf.org/cimi/2/mapped: Indicates a Volume that shall be used for shared storage that might be available to multiple Machines, but which does not require an explicit mount operation from within the guest operating system.  
Additional values may be defined. If certain types of Volumes require additional data, it is expected that this Resource is extended. For example, a "sharedFileSystem" type might require additional networking information and credentials to be specified. |
| capacity  | integer  | The maximum size, if limited, of the Volume in kilobytes. If this value is increased, the Volume can contain more data. Decreasing this value may require evaluations. |
| bootable  | boolean  | This property indicates whether this Volume is bootable. |
| images    | collection [Volume Image] | A reference to the list of references to VolumeImages that represent snapshots taken from the Volume.  
Note: This Collection has the semantics of usage of VolumeImages by the Volume (deleting the Volume does not cause the deletion of the referred VolumeImages). |
| meters    | collection [Meter] | A reference to the list of Meters monitored for this Volume. |
| eventLog  | ref      | A reference to the EventLog of this Volume. |

When implementing or using Volume, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 26 as well as in the tables describing embedded Resources or related Collections.

5.15.1.1 Collections

The following clauses describe the Collection Resources owned by Volumes.

5.15.1.1.1 images Collection

The Resource type for each item of this Collection is "VolumeImage". There is no accessory attribute for the items in this Collection, therefore, it is a basic VolumeImage Collection (serialized as described in 5.5.12).
See the VolumeImageCollection Resource clause.

NOTE Previous versions of this specification included an "add" operation on this Resource. It is now deprecated in favor of creating a new VolumeImage with the imageLocation attribute pointing to the Volume to be captured.

5.15.1.2 meters Collection

The Resource type for each item of this Collection is "Meter" as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic Meter Collection (serialized as described in 5.5.12).

See the MeterCollection Resource clause.

5.15.1.2 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the VolumeCollection Resource.

In addition also the following custom operations are supported.

snapshot

/link@rel: http://schemas.dmtf.org/cimi/2/action/snapshot

This operation shall create a new VolumeImage from an existing Volume. This operation is defined within the VolumeImage Resource; see 5.15.7.1 for more details. Note that while this operation is performed against a VolumeImage, its presence in the Volume serialization is used to advertise support for the operation.

If the Volume is unable to accept operations at any point while it is creating the VolumeImage, the Volume shall be in the CAPTURING state.

restore

/link@rel: http://schemas.dmtf.org/cimi/2/action/restore

This operation shall restore a Volume from a previously created VolumeImage.

Input parameters:

1) "image" - type: ref - mandatory.
2) A reference to the Volume Image.

Output parameters: None.

During the processing of this operation, the Volume shall be in the "RESTORING" state.

Upon successful completion of this operation, the Volume shall again be in the state "AVAILABLE".

HTTP protocol

To restore a Volume, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/restore" URI of the Volume where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/restore",
}
```
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

XML media type: application/xml

XML serialization

```
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/restore </action>
  <image href="xs:anyURI"/>
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties>
  <xs:any>*
</Action>
```

Where the "image" ref content is a reference to the VolumeImage to be used.

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

5.15.2 VolumeCollection Resource

A VolumeCollection Resource represents the Collection of Volumes within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.15.2.1 Operations

NOTE The "add" operation requires that a VolumeTemplate be used (see 4.2.1.1).

5.15.3 VolumeTemplate Resource

This Resource captures the configuration values for realizing a Volume. A VolumeTemplate may be used to create multiple Volumes. Table 27 describes the VolumeTemplate attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/VolumeTemplate">http://schemas.dmtf.org/cimi/2/VolumeTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>volumeConfig</td>
<td>ref</td>
</tr>
<tr>
<td>volumeImage</td>
<td>ref</td>
</tr>
<tr>
<td>meterTemplates</td>
<td>Meter Templates[]</td>
</tr>
<tr>
<td>Name</td>
<td>VolumeTemplate</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/VolumeTemplate">http://schemas.dmtf.org/cimi/2/VolumeTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventLog Template</td>
<td>ref</td>
<td>A reference to an EventLogTemplate that shall be used to create and connect a new EventLog to the new Volume. Note that the attributes of the EventLogTemplate may be specified rather than a reference to an existing EventLogTemplate Resource.</td>
</tr>
<tr>
<td>genResource Metadata</td>
<td>ref</td>
<td>A reference to a ResourceMetadata that shall be associated with every Volume generated from this Template.</td>
</tr>
</tbody>
</table>

When implementing or using `VolumeTemplate`, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 27 as well as in the tables describing embedded Resources or related Collections.

### 5.15.3.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the `VolumeTemplateCollection Resource`.

### 5.15.4 VolumeTemplateCollection Resource

A `VolumeTemplateCollection` Resource represents the Collection of `VolumeTemplate` Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

### 5.15.4.1 Operations

This Resource supports the Read and Update operations. Creation of new `VolumeTemplate` Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

### 5.15.5 VolumeConfiguration Resource

The `VolumeConfiguration` Resource represents the set of configuration values needed to create a Volume with certain characteristics. `VolumeConfigurations` are created by Providers and may, at the Providers discretion, be created by Consumers.

Table 28 describes the `VolumeConfiguration` attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeConfiguration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/VolumeConfiguration">http://schemas.dmtf.org/cimi/2/VolumeConfiguration</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>URI</td>
<td>A URI that indicates the type of <code>Volume</code> to be created. This specification defines the following URI: <a href="http://schemas.dmtf.org/cimi/1/mapped">http://schemas.dmtf.org/cimi/1/mapped</a>, which indicates a <code>Volume</code> that shall be used for shared storage that might be available to multiple Machines, but which does not require an explicit mount operation from within the guest operating system. Additional values may be defined. If certain types of <code>Volumes</code> require additional data, it is expected that this Resource is extended.</td>
</tr>
<tr>
<td>Name</td>
<td>VolumeConfiguration</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/VolumeConfiguration">http://schemas.dmtf.org/cimi/2/VolumeConfiguration</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>format</td>
<td>string</td>
<td>The format of the file system that is placed on Volumes created from this configuration. This attribute is only meaningful for VolumeConfigurations that describe block devices. This attribute is optional; the absence of this attribute indicates that Volumes created from this configuration are not formatted with a file system. Example values: &quot;ext4,&quot; &quot;ntfs.&quot;</td>
</tr>
<tr>
<td>capacity</td>
<td>integer</td>
<td>The default size in kilobytes, if limited, of the Volume created from this VolumeConfiguration.</td>
</tr>
</tbody>
</table>

### 5.15.5.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the VolumeConfigurationCollection Resource.

### 5.15.6 VolumeConfigurationCollection Resource

A VolumeConfigurationCollection Resource represents the Collection of VolumeConfiguration Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

### 5.15.6.1 Operations

This Resource supports the Read and Update operations. Creation of new VolumeImage Resources is supported by way of a POST to the "add" operations' URI as described in clause 4.2.1.1.

### 5.15.7 VolumeImage Resource

This Resource represents an image that could be placed on a preloaded volume. Table 29 describes the VolumeImage attributes.

#### Table 29 – VolumeImage attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>VolumeImage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/VolumeImage">http://schemas.dmtf.org/cimi/2/VolumeImage</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>The operational state of the VolumeImage. Allowed values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CREATING: The VolumeImage is in the process of being created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AVAILABLE: The VolumeImage is available and ready for use. Unless otherwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td>specified, the VolumeImage shall initially be in this state after successful</td>
</tr>
<tr>
<td></td>
<td></td>
<td>creation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DELETING: The VolumeImage is in the process of being deleted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ERROR: The Provider has detected an error in the VolumeImage. The operations</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that result in transitions to the above defined states are defined in clause</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5.15.7.1.</td>
</tr>
<tr>
<td>imageLocation</td>
<td>URI</td>
<td>A reference to the location of the binary data that makes up this image.</td>
</tr>
<tr>
<td>bootable</td>
<td>boolean</td>
<td>This property indicates whether Volumes created from this VolumeImage are bootable.</td>
</tr>
</tbody>
</table>
5.15.7.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the VolumeImageCollection Resource.

5.15.8 VolumeImageCollection Resource

A VolumeImageCollection Resource represents the Collection of VolumeImage Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.15.8.1 Operations

This Resource supports the Read and Update operations. Creation of new VolumeImage Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

During the creation of a new VolumeImage Resource, if the "imageLocation" attribute refers to an existing Volume, this operation shall be interpreted as a request to create a snapshot of the Volume.

Once completed, the "imageLocation" attribute of the new VolumeImage Resource shall not refer to the original Volume; instead it shall refer to a static copy of the Volume. Additionally, the referenced Volume's VolumeImageCollection shall be updated to include a reference to this newly created snapshot VolumeImage Resource. During this process, the Provider may put the Volume into a "CAPTURING" state if necessary.

5.16 Network Resources and relationships

A Network is a logical construct that allows communication between defined Endpoints within a Segment. Each Segment uses a single, fixed, protocol to communicate and access is provided by associating an Endpoint with an Interface.

Only Endpoints within a Segment can communicate implicitly. All other communication must be explicitly enabled using Network Services.

- Each Network has one or more Segments.
- Each Segment supports communication using a single protocol.
- Each Segment may have one or more addressable Endpoints.
- Each Endpoint is associated with a single Segment.
- Each Endpoint may be associated with a single Interface.
- An Interface can be associated with more than one Endpoint.
- A Network may contain subordinate Networks to form hierarchical structures (similar to Systems).
- One or more Services may be associated with a Network to provide additional functionality.

5.16.1 Network

Table 30 describes the Network Resource attributes.
Table 30 – Network attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>string</td>
<td>The operational state of the Network. Allowed values are:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CREATING: The Network is in the process of being created.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STARTING: The Network is in the process of being started.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STARTED: The Network is available and ready for use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOPPING: The Network is in the process of being stopped.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOPPED: The Network is stopped and not available for use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DELETING: The Network is in the process of being deleted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ERROR: The Provider has detected an error in the Network.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The operations that result in transitions to the above defined states are</td>
</tr>
<tr>
<td></td>
<td></td>
<td>defined in clause 5.16.2.1. Clause 5.16.2.1 defines the initial state of a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Network.</td>
</tr>
<tr>
<td>services</td>
<td>collection [Network Service]</td>
<td>A reference to a Collection of Services that may be applied to this Network.</td>
</tr>
<tr>
<td>subnetworks</td>
<td>collection [Network]</td>
<td>A reference to a Collection of subordinate Networks contained within this</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Network.</td>
</tr>
<tr>
<td>meters</td>
<td>collection [Meter]</td>
<td>A reference to the list of Meters monitored for this Network.</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
<td>A reference to the EventLog of this Network.</td>
</tr>
</tbody>
</table>

The Provider should supply at least one Network Resource in the CEP Networks Collection to represent communication channels that are external to the Consumers cloud. Typically this would be a connection to the Internet. As an alternative the Provider may supply a NetworkTemplate Resource by which such external Networks can be created when required.

When implementing or using Network Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 30 as well as in the tables describing embedded Resources or related Collections. Both Consumer and Provider shall serialize this Resource as described below. The following pseudo-schemas (see notation in 1.3) describe the serialization of the Resource in both JSON and XML.

5.16.1.1 Collections

The following clauses describe the Collection Resources that are components of Networks.

5.16.1.1.1 segments Collection

The Resource type for each item of this Collection is “ProtocolSegment”. There is no accessory attribute for the items in this Collection, therefore, it is a basic ProtocolSegmentCollection, as described in 5.16.6.
5.16.1.1.2 services Collection

The Resource type for each item of this Collection is "NetworkService". There is no accessory attribute for the items in this Collection, therefore, it is a basic NetworkServiceCollection, as described in 5.16.18.

5.16.1.1.3 subnetworks Collection

The Resource type for each item of this Collection is "Network". There is no accessory attribute for the items in this Collection, therefore, it is a basic NetworkCollection, as described in 5.16.2.

5.16.1.1.4 meters Collection

The Resource type for each item of this Collection is "Meter" as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic MeterCollection as described in 5.5.12.

See the MeterCollection Resource clause.

5.16.1.2 Operations

Network Resources support the Read, Update, and Delete operations. Create is supported through the NetworkCollection Resource, as described in 5.16.2.

The following custom operations are also defined:

start

/link@rel: http://schemas.dmtf.org/cimi/2/action/start

This operation shall recursively start and enable all the components within a Network.

Input parameters: None.

Output parameters: None.

During the processing of this operation, the Network shall be in the "STARTING" state.

Upon successful completion of this operation, the Network shall be in the "STARTED" state.

HTTP protocol

To start a Network, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/start" URI of the Network where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
"resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
"action": "http://schemas.dmtf.org/cimi/2/action/start",
"properties": { string: string, + } ?
...
}
```
Upon successful processing of the request, the HTTP response body may be empty.

/\link@rel: http://schemas.dmtf.org/cimi/2/action/stop

This operation shall recursively stop and disable all components of a Network.

Input parameters: None.

Output parameters: None.

During the processing of this operation, the Network shall be in the "STOPPING" state.

Upon successful completion of this operation, the Network shall be in the "STOPPED" state.

HTTP protocol

To stop a Network, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/stop" URI of the Network where the HTTP request body shall be as described below.

JSON media type: application/json

```json
{ "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
 "action": "http://schemas.dmtf.org/cimi/2/action/stop",
 "properties": { string: string, + } ?,
 ...
}
```

XML media type: application/xml

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/stop </action>
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties> ?
  <xs:any>*
</Action>
```
Upon successful processing of the request, the HTTP response body may be empty.

### 5.16.2 NetworkCollection Resource

A NetworkCollection Resource represents the Collection of Networks and follows the Collection pattern that is defined in clause 5.5.12.

#### 5.16.2.1 Operations

**NOTE** The "add" operation requires that a NetworkTemplate be used (see 5.16.3).

Upon successful processing of the "add" operation, unless otherwise specified by way of the NetworkTemplate "initialState" attribute, the state of the new Network shall be the value of the DefaultInitialState capability of the Network Resource's ResourceMetadata, if defined. If no DefaultInitialState capability is defined, the default value shall be "STOPPED." The semantics of "initialState" shall be equivalent to the Provider issuing the appropriate actions against the new Network to move it into that state.

If a Provider is unable to change the state of the new Network to the appropriate "initialState" (either as specified by the NetworkTemplate or as implied by the previous stated rules), the Network creation shall fail.

### 5.16.3 NetworkTemplate Resource

The NetworkTemplate is a set of configuration values for realizing a Network. An instance of NetworkTemplate may be used to create multiple Networks. Table 31 describes the NetworkTemplate attributes.

#### Table 31 – NetworkTemplate attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>NetworkTemplate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/NetworkTemplate">http://schemas.dmtf.org/cimi/2/NetworkTemplate</a></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>initialState</td>
<td>string</td>
<td>Sets the initial state of a Network created using this Template.</td>
<td>string</td>
<td>The allowed values are restricted to the non-transient states specified for the state attribute of the Network Resource, described in Table 30. Providers should advertise the list of available values via the Network ResourceMetadata initialStates Capability.</td>
</tr>
<tr>
<td>segments</td>
<td>Protocol Segment[]</td>
<td>A list of references to existing ProtocolSegment Resources to be inserted into the &quot;segments&quot; collection of the Network Resource created using this Template.</td>
<td>Protocol Segment[]</td>
<td></td>
</tr>
<tr>
<td>segmentTemplates</td>
<td>Protocol Segment Template[]</td>
<td>A list of references to ProtocolSegmentTemplates, from each of which a ProtocolSegment Resource is created and its reference inserted into the &quot;segments&quot; collection of the Network Resource created using this NetworkTemplate.</td>
<td>Protocol Segment Template[]</td>
<td></td>
</tr>
<tr>
<td>services</td>
<td>Network Service[]</td>
<td>A list of references to NetworkService Resources to be added to the &quot;services&quot; collection of the Network Resource created using this Template.</td>
<td>Network Service[]</td>
<td></td>
</tr>
<tr>
<td>serviceTemplates</td>
<td>Network Service Template[]</td>
<td>A list of references to NetworkServiceTemplates, from each of which a NetworkService Resource is created and its reference inserted into the &quot;services&quot; collection of the Network Resource created using this Template.</td>
<td>Network Service Template[]</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>NetworkTemplate</td>
<td>Type URI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------</td>
<td>----------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>NetworkTemplate</td>
<td><a href="http://schemas.dmtf.org/cimi/2/NetworkTemplate">http://schemas.dmtf.org/cimi/2/NetworkTemplate</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>subnetworks</td>
<td>Network[]</td>
<td>A list of references to Network Resources to be added to the subnetworks collection of the Network created from this NetworkTemplate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>subnetworkTemplates</td>
<td>Network Template[]</td>
<td>A list of references to NetworkTemplates, from each of which a Network Resource is created and added to the subnetworks collection of the Network created using this NetworkTemplate.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>meterTemplates</td>
<td>Meter Template[]</td>
<td>A list of references to MeterTemplates that shall be used to create and connect a set of new Meters to the new Network. Note that the attributes of the MeterTemplate may be specified rather than a reference to an existing MeterTemplate Resource.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>eventLogTemplate</td>
<td>ref</td>
<td>A reference to an EventLogTemplate that shall be used to create and connect a new EventLog to the new Network. Note that the attributes of the EventLogTemplate may be specified rather than a reference to an existing EventLogTemplate Resource.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When implementing or using NetworkTemplate, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 31 as well as in the tables describing embedded Resources or related Collections.

### 5.16.3.1 Operations

The NetworkTemplate Resource supports the Read, Update and Delete operations. Create is supported through the NetworkTemplateCollection Resource.

### 5.16.4 NetworkTemplateCollection Resource

A NetworkTemplateCollection Resource represents the Collection of NetworkTemplates within a Provider and follows the Collection pattern defined in clause 5.5.12.

### 5.16.4.1 Operations

The NetworkTemplateCollection Resource supports the Read and Update operations. Creation of new NetworkTemplate Resources is supported by way of a POST to the “add” operation’s URI as described in clause 4.2.1.1.

### 5.16.5 Segments

A Segment is an individual channel within a Network that utilizes a single communication protocol. Segments are ProtocolSegment Resources, the attributes of which are described in Table 32.
## Table 32 – ProtocolSegment attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProtocolSegment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ProtocolSegment">http://schemas.dmtf.org/cimi/2/ProtocolSegment</a></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
<td>The operational state of the Segment. Allowed values are: CREATING: The Segment is in the process of being created. STARTED: The Segment is available (enabled) and ready for use. STOPPED: The Segment is stopped (disabled) and not available for use. DELETING: The Segment is in the process of being deleted. ERROR: The Provider has detected an error in the Segment. The operations that result in transitions to the above defined states are defined in clause 5.16.5.3. Clause 5.16.6.1 defines the initial state of a Segment.</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The official name of the protocol supported by this segment. Allowed values are: Ethernet: As defined by IEEE 802.3. IPv4: Internet Protocol version 4, as defined in RFC 791. IPv6: Internet Protocol Version 6 as defined in RFC 2460.</td>
</tr>
<tr>
<td>noDefault Routing</td>
<td>boolean</td>
<td>If set to TRUE the default communication between Endpoints within the Segment is disabled. Communication between Endpoints in this case must be performed by a Service. The default value is FALSE, which enables communication between endpoints.</td>
</tr>
<tr>
<td>endpoints</td>
<td>collection [Protocol Endpoint]</td>
<td>A reference to a list of references to Endpoints associated with this Segment.</td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
<td>A polymorphic attribute the contents of which depend on the specific network protocol. As examples this would include &quot;netmask&quot; for IPv4 and &quot;bandwidth&quot; for &quot;Ethernet&quot;. See tables 33-35 for details of the data to be included.</td>
</tr>
<tr>
<td>meters</td>
<td>collection [Meter]</td>
<td>A reference to the list of Meters monitored for this Segment.</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
<td>A reference to the EventLog of this Segment.</td>
</tr>
</tbody>
</table>

When implementing or using ProtocolSegment Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 32 as well as in the tables describing embedded Resources or related Collections.

### 5.16.5.1 Protocol-specific parameters

Each Segment may require additional data that is specific to a communication protocol. This additional data is specified using the parameters attribute of the ProtocolSegment. This specification defines the following key-value pairs that must be supplied for the indicated protocols:
### Table 33 - IPv6 ProtocolSegment parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>IPv6ProtocolParameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
<td><strong>Value Type</strong></td>
</tr>
<tr>
<td>prefixLength</td>
<td>integer</td>
</tr>
<tr>
<td>subnetAddress</td>
<td>string</td>
</tr>
</tbody>
</table>

### Table 34 - IPv4 ProtocolSegment parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>IPv4ProtocolParameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
<td><strong>Value Type</strong></td>
</tr>
<tr>
<td>netmask</td>
<td>string</td>
</tr>
<tr>
<td>subnetAddress</td>
<td>string</td>
</tr>
</tbody>
</table>

### Table 35 – Ethernet ProtocolSegment parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>EthernetProtocolParameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key</strong></td>
<td><strong>Value Type</strong></td>
</tr>
<tr>
<td>speed</td>
<td>integer</td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
</tr>
</tbody>
</table>

Note that Providers may support additional key-value pairs for the parameter attribute to extend the existing protocols. Consumers are not required to process any additional key-value pairs but must return them to the Provider in the serialization of ProtocolSegments.

### 5.16.5.2 Collections

The following clauses describe the Collection Resources that are components of ProtocolSegments.

#### 5.16.5.2.1 endpoints Collection

The Resource type for each item of this Collection is a “ProtocolEndpoint” as defined in clause 5.16.9. There is no accessory attribute for the items in this Collection therefore, it is a basic ProtocolEndpointCollection Resource, serialized as described in 5.16.10.

#### 5.16.5.2.2 meters Collection

The Resource type for each item of this Collection is “Meter” as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic Meter Collection (serialized as described in 5.5.12).

### 5.16.5.3 Operations

The ProtocolSegment Resource supports the Read, Update, and Delete operations. Create is supported through the ProtocolSegmentCollection Resource.
Deleting a ProtocolSegment shall remove that Segment from the global (Cloud Entry Point) ProtocolSegmentCollection and also all references to the Segment in Collections of other Resources (e.g., from corresponding Network segments Collection).

The following custom operations are also defined:

start
/link@rel: http://schemas.dmtf.org/cimi/2/action/start

This operation shall start a ProtocolSegment.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the ProtocolSegment shall be in the "STARTED" state.

HTTP protocol

To start a ProtocolSegment, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/start" URI of the ProtocolSegment where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/start",
    "properties": { "string": string, + } ?
...
}
```

XML media type: application/xml

XML serialization

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/start </action>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties>
</Action>
```

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

stop
/link@rel: http://schemas.dmtf.org/cimi/2/action/stop

This operation shall stop a ProtocolSegment.

Input parameters: None.

Output parameters: None.
Upon successful completion of this operation, the ProtocolSegment shall be in the "STOPPED" state.

**HTTP protocol**

To stop a ProtocolSegment, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/stop" URI of the ProtocolSegment where the HTTP request body shall be as described below.

**JSON media type: application/json**

**JSON serialization:**

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/stop",
  "properties": { string: string, + } ?,
  ...
}
```

**XML media type: application/xml**

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/stop </action>
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties> ?
  <xs:any>*
</Action>
```

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

### 5.16.6 ProtocolSegmentCollection Resource

A ProtocolSegmentCollection Resource represents the Collection of ProtocolSegments within a Provider and follows the Collection pattern defined in clause 5.5.12.

#### 5.16.6.1 Operations

**NOTE** The "add" operation requires that a ProtocolSegmentTemplate be used (see clause 5.16.7).

Upon successful processing of the "add" operation, unless otherwise specified by the ProtocolSegmentTemplate "initialState" attribute, the state of the new ProtocolSegment shall be the value of the DefaultInitialState capability of the ProtocolSegment Resource's ResourceMetadata, if defined. If no DefaultInitialState capability is defined, the default value shall be "STOPPED." The semantics of "initialState" shall be equivalent to the Provider issuing the appropriate actions against the new ProtocolSegment to move it into that state.

If a Provider is unable to change the state of the new ProtocolSegment to the appropriate "initialState" (either as specified by the ProtocolSegmentTemplate or as implied by the previous stated rules), the ProtocolSegment creation shall fail.
5.16.7 ProtocolSegmentTemplate Resource

The ProtocolSegmentTemplate is a set of configuration values for realizing a ProtocolSegment. A ProtocolSegmentTemplate may be used to create multiple ProtocolSegments. Table 36 describes the ProtocolSegmentTemplate attributes.

Table 36 – ProtocolSegmentTemplate attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>ProtocolSegmentTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ProtocolSegmentTemplate">http://schemas.dmtf.org/cimi/2/ProtocolSegmentTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>network</td>
<td>ref</td>
</tr>
<tr>
<td>initialState</td>
<td>string</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
</tr>
<tr>
<td>noDefaultRouting</td>
<td>boolean</td>
</tr>
<tr>
<td>endpoints</td>
<td>Protocol Endpoint[]</td>
</tr>
<tr>
<td>endpointTemplates</td>
<td>Protocol Endpoint Template[]</td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
</tr>
<tr>
<td>meterTemplates</td>
<td>meterTemplates[]</td>
</tr>
<tr>
<td>eventLogTemplate</td>
<td>ref</td>
</tr>
</tbody>
</table>
When implementing or using ProtocolSegmentTemplate Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 36 as well as in the tables describing embedded Resources or related Collections.

5.16.7.1 Collections

The ProtocolSegmentTemplate Resource has no attributes of type Collection.

5.16.7.2 Operations

The ProtocolSegmentTemplate Resource supports the Read, Update, and Delete operations. Create is supported through the ProtocolSegmentTemplateCollection Resource.

5.16.8 ProtocolSegmentTemplateCollection Resource

A ProtocolSegmentTemplateCollection Resource represents the Collection of ProtocolSegmentTemplates within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.16.8.1 Operations

The ProtocolSegmentTemplateCollection Resource supports the Read and Update operations. Creation of new ProtocolSegmentTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

5.16.9 Endpoints

An Endpoint is an addressable element within a protocol that is a source, destination, or source and destination for communication. Endpoints are ProtocolEndpoint Resources, the attributes of which are described in Table 37.

Table 37 – ProtocolEndpoint attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>ProtocolSegment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ProtocolEndpoint">http://schemas.dmtf.org/cimi/2/ProtocolEndpoint</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>The operational state of the Endpoint. Allowable values are: CREATING: The Endpoint is in the process of being created. ENABLED: The Endpoint is available and ready for use. DISABLED: The Endpoint is not available for use. DELETING: The Endpoint is in the process of being deleted. ERROR: The Provider has detected an error in the Endpoint. The operations that result in transitions to the above defined states are defined in clause 5.16.9.3. Clause 5.16.10.1 defines the initial state of an Endpoint.</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The official name of the protocol supported by this segment. This attribute is intended as a convenience only and if specified, its value must be identical to the value of the protocol attribute of the Segment with which the Endpoint is associated. Possible values are those specified in the ProtocolSegment Resource described in clause 5.16.5.</td>
</tr>
<tr>
<td>address</td>
<td>string</td>
<td>The address assigned to this Endpoint in the format required by the supported protocol.</td>
</tr>
<tr>
<td>Name</td>
<td>ProtocolSegment</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ProtocolEndpoint">http://schemas.dmtf.org/cimi/2/ProtocolEndpoint</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>origin</td>
<td>string</td>
<td>A string representing how protocol specific data is assigned to this Endpoint. Allowable values are: [ STATIC</td>
</tr>
<tr>
<td>interface</td>
<td>Network Interface</td>
<td>A reference to the Interface that is used to connect to the Network using this Endpoint.</td>
</tr>
<tr>
<td>parameters</td>
<td>map</td>
<td>A polymorphic attribute the contents of which depend on the specific network protocol. As examples this would include &quot;netmask&quot; for IPv4 and &quot;bandwidth&quot; for &quot;Ethernet&quot;. See tables 38-40 for details of the data to be included.</td>
</tr>
<tr>
<td>meters</td>
<td>collection [Meter]</td>
<td>A reference to the list of Meters monitored for this Endpoint.</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
<td>A reference to the EventLog of this Endpoint.</td>
</tr>
</tbody>
</table>

When implementing or using ProtocolEndpoint, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 37 as well as in the tables describing embedded Resources or related Collections.

5.16.9.1 Protocol specific parameters

Each Endpoint may require additional data that is specific to the communication protocol supported. This additional data is specified using the parameters attribute of a ProtocolEndpoint. This specification defines the following key-value pairs that provide supplemental information for Endpoints of specific protocol types:

### Table 38 – IPv6 ProtocolEndpoint parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>IPv6ProtocolEndpointParameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>Value Type</td>
</tr>
<tr>
<td>addressType</td>
<td>string</td>
</tr>
<tr>
<td>prefixLength</td>
<td>integer</td>
</tr>
</tbody>
</table>

### Table 39 – IPv4 ProtocolEndpoint parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>IPv4ProtocolEndpointParameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key</td>
<td>Value Type</td>
</tr>
<tr>
<td>hostname</td>
<td>string</td>
</tr>
</tbody>
</table>
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

### Table 40 – Ethernet ProtocolEndpoint parameters

<table>
<thead>
<tr>
<th>Key</th>
<th>Value Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aliases</td>
<td>string[]</td>
<td>Other unicast addresses that may be used to communicate with the Endpoint.</td>
</tr>
<tr>
<td>groupAddresses</td>
<td>string[]</td>
<td>Multicast addresses to which the Endpoint listens.</td>
</tr>
</tbody>
</table>

Note that Providers may support additional key-value pairs for the `parameter` attribute to extend the existing protocols. Consumers are not required to process any additional key-value pairs, but must return them to the Provider in the serialization of ProtocolEndpoints.

### 5.16.9.2 Collections

The following clauses describe the Collection Resources that are components of ProtocolEndpoints.

#### 5.16.9.2.1 meters Collection

The Resource type for each item of this Collection is "Meter" as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic Meter Collection (serialized as described in 5.5.12).

### 5.16.9.3 Operations

The ProtocolEndpoints Resource supports the Read, Update, and Delete operations. Create is supported through the ProtocolEndpointCollection Resource.

Deleting a ProtocolEndpoint shall remove that Endpoint from the global (Cloud Entry Point) ProtocolEndpointCollection. Additionally, references to the Endpoint in ProtocolEndpointCollections of all other Resources (e.g., ProtocolSegments, NetworkServices) must be removed.

The following custom operations are also defined:

**enable**

[link@rel: http://schemas.dmtf.org/cimi/2/action/enable](#)

This operation shall enable a ProtocolEndpoint.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the ProtocolEndpoint shall be in the "ENABLED" state.

**HTTP protocol**

To enable a ProtocolEndpoint, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/enable" URI of the ProtocolEndpoint where the HTTP request body shall be as described below.
Upon successful processing of the request, the HTTP response body may be empty.

disable

/link@rel: http://schemas.dmtf.org/cimi/2/action/disable

This operation shall disable a ProtocolEndpoint.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the ProtocolEndpoint shall be in the "DISABLED" state.

HTTP protocol

To stop a ProtocolEndpoint, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/disable" URI of the ProtocolEndpoint where the HTTP request body shall be as described below.

**JSON media type: application/json**

**JSON serialization:**

```json
{ "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
 "action": "http://schemas.dmtf.org/cimi/2/action/disable",
 "properties": [ string: string, + ] ?
 ...
 }
```

**XML media type: application/xml**

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
 <action> http://schemas.dmtf.org/cimi/2/action/disable </action>
 <properties>
   <property key="xs:string"> xs:string </property> *
 </properties> ?
 <xs:any>*</Action>
```

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

disable

/link@rel: http://schemas.dmtf.org/cimi/2/action/disable

This operation shall disable a ProtocolEndpoint.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the ProtocolEndpoint shall be in the "DISABLED" state.

HTTP protocol

To stop a ProtocolEndpoint, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/disable" URI of the ProtocolEndpoint where the HTTP request body shall be as described below.

**JSON media type: application/json**

**JSON serialization:**

```json
{ "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
 "action": "http://schemas.dmtf.org/cimi/2/action/disable",
 "properties": [ string: string, + ] ?
 ...
 }
```
Upon successful processing of the request, the HTTP response body may be empty.

5.16.10 ProtocolEndpointCollection Resource

A ProtocolEndpointCollection Resource represents the Collection of ProtocolEndpoints within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.16.10.1 Operations

NOTE The "add" operation requires that a ProtocolEndpointTemplate be used (see clause 5.16.11).

Upon successful processing of the "add" operation, unless otherwise specified by the ProtocolEndpointTemplate "initialState" attribute, the state of the new ProtocolEndpoint shall be the value of the DefaultInitialState capability of the ProtocolEndpoint Resource's ResourceMetadata, if defined. If no DefaultInitialState capability is defined, the default value shall be "DISABLED." The semantics of "initialState" shall be equivalent to the Provider issuing the appropriate actions against the new ProtocolEndpoint to move it into that state.

If a Provider is unable to change the state of the new ProtocolEndpoint to the appropriate "initialState" (either as specified by the ProtocolEndpointTemplate or as implied by the previous stated rules), the ProtocolEndpoint creation shall fail.

5.16.11 ProtocolEndpointTemplate Resource

The ProtocolEndpointTemplate is a set of configuration values for realizing a ProtocolEndpoint. A ProtocolEndpointTemplate may be used to create multiple ProtocolEndpoints. Table 41 describes the ProtocolEndpointTemplate attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>ProtocolEndpointTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ProtocolEndpointTemplate">http://schemas.dmtf.org/cimi/2/ProtocolEndpointTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>initialState</td>
<td>string</td>
</tr>
<tr>
<td>Name</td>
<td>ProtocolEndpointTemplate</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/ProtocolEndpointTemplate">http://schemas.dmtf.org/cimi/2/ProtocolEndpointTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| address             | string  | If the origin attribute value is “STATIC”, this attribute contains the address to be assigned to this Endpoint in the format required by the supported protocol.  
If the origin attribute value is “DYNAMIC”, this attribute must not be supplied by the Template. |
| origin              | string  | A string representing how protocol specific data is assigned to this Endpoint. Allowable values are: [ STATIC | DYNAMIC ]  
If the value of this attribute is “STATIC”, all protocol specific data for the Endpoint must be supplied by this Template.  
If the value of this attribute is “DYNAMIC”, the protocol specific data for this Endpoint is allocated by other mechanisms and must not be supplied by this Template. |
| interface           | Network Interface | A reference to a NetworkInterface Resource with which this new Endpoint is associated.                                                      |
| parameters          | map     | A polymorphic attribute the contents of which depend on the specific protocol supported. The allowed key-value pairs are as specified in clause 5.16.9.  
Whether this data is required to be supplied by this Template is determined by the value of the “origin” attribute described above. |
| meterTemplates      | MeterTemplate[] | A list of references to MeterTemplates that shall be used to create and connect a set of new Meters to the new ProtocolEndpoint.  
Note that the attributes of the MeterTemplate may be specified rather than a reference to an existing MeterTemplate Resource. |
| eventLogTemplate    | ref     | A reference to an EventLogTemplate that shall be used to create and connect a new EventLog to the new ProtocolEndpoint.  
Note that the attributes of the EventLogTemplate may be specified rather than a reference to an existing EventLogTemplate Resource. |

When implementing or using ProtocolEndpointTemplate Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 41 as well as in the tables describing embedded Resources or related Collections.

5.16.11.1 Collections

The ProtocolEndpointTemplate Resource has no attributes of type Collection.

5.16.11.2 Operations

The ProtocolEndpointTemplate Resource supports the Read, Update, and Delete operations.

Create is supported through the ProtocolEndpointTemplateCollection Resource.

5.16.12 ProtocolEndpointTemplateCollection Resource

A ProtocolEndpointTemplateCollection Resource represents the Collection of ProtocolEndpointTemplates within a Provider and follows the Collection pattern defined in clause 5.5.12.
5.16.12.1 Operations

The ProtocolEndpointTemplateCollection Resource supports the Read and Update operations.
Creation of new ProtocolEndpointTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

5.16.13 Interfaces

An Interface provides a connection to a Network by associating Endpoints with Machines. The model is basically that of a virtual Network Interface Card (vNIC) that can support multiple communication protocols at multiple levels. Interfaces are NetworkInterface Resources, the attributes of which are described in Table 42.

Table 42 – NetworkInterface attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkInterface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/NetworkInterface">http://schemas.dmtf.org/cimi/2/NetworkInterface</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>endpoints</td>
<td>collection [Protocol Endpoint]</td>
</tr>
<tr>
<td>speed</td>
<td>integer</td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
</tr>
<tr>
<td>meters</td>
<td>collection [Meter]</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
</tr>
</tbody>
</table>

When implementing or using NetworkInterface, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 42 as well as in the tables describing embedded Resources or related Collections.

5.16.13.1 Collections

The following clauses describe the Collection Resources that are components of NetworkInterfaces.
5.16.13.1.1 meters Collection

The Resource type for each item of this Collection is "Meter" as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic Meter Collection (serialized as described in 5.5.12).

5.16.13.2 Operations

The NetworkInterfaces Resource supports the Read, Update, and Delete operations. Create is supported through the NetworkInterfaceCollection Resource.

Deleting a NetworkInterface shall remove that Endpoint from the global (Cloud Entry Point) NetworkInterfaceCollection. Additionally, references to the Endpoint in NetworkInterfaceCollections of all other Resources (e.g., ProtocolEndpoints, NetworkServices) must be removed.

The following custom operations are also defined:

enable

/link@rel: http://schemas.dmtf.org/cimi/2/action/enable

This operation shall enable a NetworkInterface.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the NetworkInterface shall be in the "ENABLED" state.

HTTP protocol

To enable a NetworkInterface, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/enable" URI of the NetworkInterface where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/enable",
  "properties": {
    "string": string,
    //...  
  }
}
```
Upon successful processing of the request, the HTTP response body may be empty.

This operation shall disable a NetworkInterface.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the NetworkInterface shall be in the "DISABLED" state.

**HTTP protocol**

To stop a NetworkInterface, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/disable" URI of the NetworkInterface where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/disable",
  "properties": {
    "string": string,
  }
}
```

**XML media type:** application/xml

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/disable </action>
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties> ?
</Action>
```
Upon successful processing of the request, the HTTP response body may be empty.

### 5.16.14 NetworkInterfaceCollection Resource

A `NetworkInterfaceCollection` Resource represents the Collection of `NetworkInterfaces` within a Provider and follows the Collection pattern defined in clause 5.5.12.

#### 5.16.14.1 Operations

**NOTE** The "add" operation requires that a `NetworkInterfaceTemplate` be used (see clause 5.16.15).

Upon successful processing of the "add" operation, unless otherwise specified by the `NetworkInterfaceTemplate"initialState" attribute, the state of the new `NetworkInterface` shall be the value of the DefaultInitialState capability of the `NetworkInterface` Resource's `ResourceMetadata`, if defined. If no DefaultInitialState capability is defined, the default value shall be "DISABLED." The semantics of "initialState" shall be equivalent to the Provider issuing the appropriate actions against the new `NetworkInterface` to move it into that state.

If a Provider is unable to change the state of the new `NetworkInterface` to the appropriate "initialState" (either as specified by the `NetworkInterfaceTemplate` or as implied by the previous stated rules), the `NetworkInterface` creation shall fail.

### 5.16.15 NetworkInterfaceTemplate Resource

The `NetworkInterfaceTemplate` is a set of configuration values for realizing a `NetworkInterface`. A `NetworkInterfaceTemplate` may be used to create multiple `NetworkInterfaces`. Table 43 describes the `NetworkInterfaceTemplate` attributes.

#### Table 43 – `NetworkInterfaceTemplate` attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td><code>NetworkInterfaceTemplate</code></td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><code>http://schemas.dmtf.org/cimi/2/NetworkInterfaceTemplate</code></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td><strong>Type</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><code>initialState</code></td>
<td><code>string</code></td>
<td>Sets the initial state of the Endpoint created using this Template. The allowed values are restricted to the nontransient states specified for the <code>state</code> attribute of the <code>NetworkInterface</code> Resource, described in 5.16.13. Providers should advertise the list of available values via the <code>NetworkInterface</code> Resource <code>ResourceMetadata initialStates Capability</code>.</td>
</tr>
<tr>
<td><code>endpoints</code></td>
<td><code>collection</code></td>
<td>A reference to a list of references to <code>ProtocolEndpoints</code> this Interface supports. Note: This Collection represents an association between the Interface and a list of Endpoints in one or more Segments.</td>
</tr>
<tr>
<td><code>speed</code></td>
<td><code>integer</code></td>
<td>The initial bandwidth of the Interface in Bits per Second.</td>
</tr>
<tr>
<td><code>mtu</code></td>
<td><code>integer</code></td>
<td>The size in bytes of the initial maximum transmission unit (MTU) that can be supported by this Interface.</td>
</tr>
<tr>
<td><code>meterTemplates</code></td>
<td><code>meterTemplates []</code></td>
<td>A list of references to <code>MeterTemplates</code> that shall be used to create and connect a set of new <code>Meters</code> to the new <code>NetworkInterface</code>. Note that the attributes of the <code>MeterTemplate</code> may be specified rather than a reference to an existing <code>MeterTemplate</code> Resource.</td>
</tr>
<tr>
<td>Name</td>
<td>NetworkInterfaceTemplate</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/NetworkInterfaceTemplate">http://schemas.dmtf.org/cimi/2/NetworkInterfaceTemplate</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventLogTemplate</td>
<td>ref</td>
<td>A reference to an EventLogTemplate that shall be used to create and connect a new EventLog to the new NetworkInterface. Note that the attributes of the EventLogTemplate may be specified rather than a reference to an existing EventLogTemplate Resource.</td>
</tr>
</tbody>
</table>

When implementing or using NetworkInterfaceTemplate Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 43 as well as in the tables describing embedded Resources or related Collections.

### 5.16.15 Collections

The following clauses describe Collection Resources that are components of NetworkInterfaceTemplates.

#### 5.16.15.1 endpoints Collection

The Resource type for each item of this Collection is "ProtocolEndpoint" as defined in clause 5.16.9. There is no accessory attribute for the items in this Collection, therefore, it is a basic ProtocolEndpointCollection (serialized as described in 5.16.10).

#### 5.16.15.2 Operations

The NetworkInterfaceTemplate Resource supports the Read, Update, and Delete operations. Create is supported through the NetworkInterfaceTemplateCollection Resource.

### 5.16.16 NetworkInterfaceTemplateCollection Resource

A NetworkInterfaceTemplateCollection Resource represents the Collection of NetworkInterfaceTemplates within a Provider and follows the Collection pattern defined in clause 5.5.12.

#### 5.16.16.1 Operations

The NetworkInterfaceTemplateCollection Resource supports the Read and Update operations. Creation of new NetworkInterfaceTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

#### 5.16.17 Services

Services provide all additional functionality within Networks beyond basic routing within a single Segment. Services can be applied to individual Segments or Endpoints, collections of Segments or Endpoints, or combinations of these elements. The actual function provided by a Service is determined and configured by policies. Services are NetworkService Resources, the attributes of which are described in Table 44.
Table 44 – NetworkService attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkService</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/NetworkService">http://schemas.dmtf.org/cimi/2/NetworkService</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>string</td>
</tr>
<tr>
<td>endpoints</td>
<td>collection</td>
</tr>
<tr>
<td>segments</td>
<td>collection</td>
</tr>
<tr>
<td>policies</td>
<td>map</td>
</tr>
<tr>
<td>meters</td>
<td>collection</td>
</tr>
<tr>
<td>eventLog</td>
<td>ref</td>
</tr>
</tbody>
</table>

When implementing or using NetworkService Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 44 as well as in the tables describing embedded Resources or related Collections.

5.16.17.1 Collections

The following clauses describe the Collection Resources that are components of NetworkServices.

5.16.17.1.1 endpoints Collection

The Resource type for each item of this Collection is a “ProtocolEndpoint” as defined in clause 5.16.9. There is no accessory attribute for the items in this Collection, therefore, it is a basic ProtocolEndpointCollection Resource, serialized as described in 5.16.10.

5.16.17.1.2 segments Collection

The Resource type for each item of this Collection is a “ProtocolSegment” as defined in clause 5.16.55.16.9. There is no accessory attribute for the items in this Collection, therefore, it is a basic ProtocolSegmentCollection Resource, serialized as described in 5.16.6.
5.16.17.1.3 meters Collection

The Resource type for each item of this Collection is "Meter" as defined in clause 5.17.3. There is no accessory attribute for the items in this Collection, therefore, it is a basic Meter Collection (serialized as described in 5.5.12).

5.16.17.2 Operations

The NetworkService Resource supports the Read, Update, and Delete operations. Create is supported through the NetworkServiceCollection Resource.

Deleting a NetworkService shall remove that Service from the global (Cloud Entry Point) NetworkServiceCollection and also all references to the Service in Collections of other Resources (e.g., from corresponding Network services Collections).

The following custom operations are also defined:

start

/link@rel: http://schemas.dmtf.org/cimi/2/action/start

This operation shall start a NetworkService.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the NetworkService shall be in the "STARTED" state.

HTTP protocol

To start a NetworkService, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/start" URI of the NetworkService where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/start",
    "properties": { string: string, + } ?
    ...
}
```

XML media type: application/xml

XML serialization

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/start </action>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties> ?
    <xs:any>*
</Action>
```
Upon successful processing of the request, the HTTP response body may be empty.

```
stop
/link@rel: http://schemas.dmtf.org/cimi/2/action/stop
```

This operation shall stop a NetworkService.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the NetworkService shall be in the "STOPPED" state.

**HTTP protocol**

To stop a NetworkService, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/stop" URI of the NetworkService where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
  "action": "http://schemas.dmtf.org/cimi/2/action/stop",
  "properties": { string: string, + } ?
  ...
}
```

**XML media type:** application/xml

**XML serialization**

```
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/stop </action>
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties> ?
  <xs:any>*</n
</Action>
```

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

### 5.16.18 NetworkServiceCollection Resource

A NetworkServiceCollection Resource represents the Collection of NetworkServices within a Provider and follows the Collection pattern defined in clause 5.5.12. This Resource shall be serialized as follows:

**JSON serialization:**

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/2/NetworkServiceCollection",
  "id": string,
  "count": number,
  "services": [
```
{ "resourceURI": "http://schemas.dmtf.org/cimi/2/NetworkService",
   "id": string,
   ... remaining NetworkService attributes ...
 }, +
}, ?
"operations": [ [ "rel": "add", "href": string ] ? ]
... 
}

XML serialization:

```xml
<Collection
   resourceURI="http://schemas.dmtf.org/cimi/2/NetworkServiceCollection"
   xmlns="http://schemas.dmtf.org/cimi/2">
   <id> xs:anyURI </id>
   <count> xs:integer </count>
   <services>
      <NetworkService>
         <id> xs:anyURI </id>
         ... remaining NetworkService attributes ...
         </NetworkService> *
      </services>
      <operations>
         <operation rel="add" href="xs:anyURI"/> ?
      </operations>
      <xs:any>*
</Collection>
```

### 5.16.18.1 Operations

**NOTE** The "add" operation requires that a NetworkServiceTemplate be used (see clause 5.16.19).

Upon successful processing of the "add" operation, unless otherwise specified by the
NetworkServiceTemplate "initialState" attribute, the state of the new NetworkService shall be the
value of the DefaultInitialState capability of the NetworkService Resource's ResourceMetadata, if
defined. If no DefaultInitialState capability is defined, the default value shall be "STOPPED." The
semantics of "initialState" shall be equivalent to the Provider issuing the appropriate actions against the
new NetworkService to move it into that state.

If a Provider is unable to change the state of the new NetworkService to the appropriate "initialState"
(either as specified by the NetworkServiceTemplate or as implied by the previous stated rules), the
NetworkService creation shall fail.

### 5.16.19 NetworkServiceTemplate Resource

The NetworkServiceTemplate is a set of configuration values for realizing a NetworkService. A
NetworkServiceTemplate may be used to create multiple NetworkServices. Table 45 describes
the NetworkServiceTemplate attributes.
Table 45 – NetworkServiceTemplate attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>NetworkServiceTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/NetworkServiceTemplate">http://schemas.dmtf.org/cimi/2/NetworkServiceTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>network</td>
<td>ref</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>initialState</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>endpoints</td>
<td>Protocol Endpoint[]}</td>
</tr>
<tr>
<td>segments</td>
<td>Protocol Segment[]}</td>
</tr>
<tr>
<td>policies</td>
<td>map</td>
</tr>
<tr>
<td>meterTemplates</td>
<td>meterTemplates []</td>
</tr>
<tr>
<td>eventLogTemplate</td>
<td>ref</td>
</tr>
</tbody>
</table>

When implementing or using NetworkServiceTemplate Resources, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 45 as well as in the tables describing embedded Resources or related Collections.

5.16.19.1 Collections

The NetworkServiceTemplate Resource has no attributes of type Collection.

5.16.19.2 Operations

The NetworkServiceTemplate Resource supports the Read, Update, and Delete operations. Create is supported through the NetworkServiceTemplateCollection Resource.
5.16.20 NetworkServiceTemplateCollection Resource

A NetworkServiceTemplateCollection Resource represents the Collection of NetworkServiceTemplates within a Provider and follows the Collection pattern defined in clause 5.5.12. Operations

The NetworkServiceTemplateCollection Resource supports the Read and Update operations. Creation of new NetworkServiceTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

5.17 Monitoring Resources and relationships

5.17.1 Job Resource

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

If a Provider supports exposing Job Resources to Consumers, each request from a Consumer that the Provider responds to with a 202 status code, shall result in a Job Resource being created and an absolute URI reference to that Job Resource shall be made available to the requesting Consumer. Providers may create additional Job Resources for Provider-initiated operations if the Provider chooses to expose these Jobs to Consumers.

If a Job is not completed successfully (e.g., it is in the FAILED or STOPPED state), this specification does not place any requirements on the Provider to ensure that the affected Resources are left in certain states. Based on the environmental conditions at that time, the Provider might choose to "undo" any impact of the operation; simply halt processing; attempt some kind of "cleanup" action; or choose to do something else. However, Providers shall list all Resources impacted by the Job in the "affectedResources" attribute, thus allowing Consumers an opportunity to examine the state of each Resource themselves. In cases where a Resource has been deleted, references to that Resource shall not appear in the "affectedResources" attribute.

The Job Resource allows for nesting of Jobs. The determination of when a single operation is converted into multiple nested Jobs is out of scope of this specification. However, if there are nested Jobs, the topmost Job Resource shall report the overall status of all Jobs and shall only be in a "SUCCESS" state if all nested Jobs are also in "SUCCESS" state. If nested Jobs are created, there is no requirement for the topmost Job Resource to reference all affected Resources in its "affectedResources" attribute. The Consumer needs to traverse the entire set of nested Jobs to determine the complete list of Resources impacted by the Jobs.

Table 46 describes the Job attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Job</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Job">http://schemas.dmtf.org/cimi/2/Job</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
</tr>
<tr>
<td>Name</td>
<td>Job</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Job">http://schemas.dmtf.org/cimi/2/Job</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>targetResource</td>
<td>ref</td>
</tr>
<tr>
<td>affectedResources</td>
<td>ref[]</td>
</tr>
<tr>
<td>action</td>
<td>URI</td>
</tr>
<tr>
<td>returnCode</td>
<td>integer</td>
</tr>
<tr>
<td>progress</td>
<td>integer</td>
</tr>
<tr>
<td>statusMessage</td>
<td>string</td>
</tr>
<tr>
<td>timeOfStatusChange</td>
<td>dateTime</td>
</tr>
<tr>
<td>parentJob</td>
<td>ref</td>
</tr>
<tr>
<td>nestedJobs</td>
<td>ref[]</td>
</tr>
</tbody>
</table>

When implementing or using Job, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 46 as well as in the tables describing referred Resources or related Collections.

5.17.1.1 Operations Resource

This Resource supports the Read, Update, and Delete operations. Deleting a Job that is in the "RUNNING" state shall be the equivalent of first stopping the Job and then deleting it. A request to delete a running Job that does not support the "stop" action shall fail.

The following custom operations are also defined:

stop

/link@rel: http://schemas.dmtf.org/cimi/2/action/stop

This operation shall stop a Job.
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

3783  Input parameters: None.
3784  Output parameters: None.

3785  During the processing of this operation, the Job shall be in the "STOPPING" state.
3786  Upon successful completion of this operation, the Job shall be in the "STOPPED" state.

3787  **HTTP protocol**
3788  To stop a Job, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/stop" URI of the Job where the HTTP request body shall be as described below.
3789  **JSON media type: application/json**
3790  **JSON serialization:**
3791  ```json
3792  {
3793  "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
3794  "action": "http://schemas.dmtf.org/cimi/2/action/stop",
3795  "properties": { string: string, + } ?
3796  ...
3797  }
3798  ```
3799  **XML media type: application/xml**
3800  **XML serialization**
3801  ```xml
3802  <Action xmlns="http://schemas.dmtf.org/cimi/2">
3803  <action> http://schemas.dmtf.org/cimi/2/action/stop </action>
3804  <properties>
3805  <property key="xs:string"> xs:string </property> *
3806  </properties> ?
3807  <xs:any>*
3808  </Action>
3809  ```

3810  Upon successful processing of the request, the HTTP response body may be empty.

3811  **5.17.2 JobCollection Resource**
3812  A JobCollection Resource represents the Collection of Jobs within a Provider and follows the Collection pattern defined in clause 5.5.12.

3813  **5.17.3 Meter Resource**
3814  This Resource represents an available Meter of some property associated to a given Resource.
3815  If a Meter's "targetResource" is deleted all Meters associated with that Resource shall also be deleted.
3816  In other words, deleting a Resource-specific MetersCollection (e.g., a Machine's MetersCollection) shall also result in the deletion of the Meters referenced from that Collection.
3817  Table 47 describes the Meter attributes.
Table 47 – Meter attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Meter</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Meter">http://schemas.dmtf.org/cimi/2/Meter</a></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>targetResource</td>
<td>ref</td>
<td>A reference to the Resource to which the Meter is related.</td>
</tr>
<tr>
<td>aspect</td>
<td>URI</td>
<td>A unique identifier representing the aspect of the Resource being metered.</td>
</tr>
<tr>
<td>units</td>
<td>string</td>
<td>The name of the used units, e.g., kilobits per second, CPU usage percentage, etc.</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>integer</td>
<td>The time between consecutive samples in seconds.</td>
</tr>
<tr>
<td>timeScope</td>
<td>string</td>
<td>The time scope to which this meter’s value applies. 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 Meter whose purpose is to provide the daily average CPU usage.</td>
</tr>
<tr>
<td>intervalDuration</td>
<td>duration</td>
<td>The interval duration when the timeScope is set to &quot;Interval&quot;. Possible values: hourly, daily, weekly, monthly, or yearly.</td>
</tr>
<tr>
<td>isContinuous</td>
<td>boolean</td>
<td>This value indicates whether the Meter value is continuous or scalar. Performance Meters are an example of a linear metric.</td>
</tr>
<tr>
<td>samples</td>
<td>collection [Sample]</td>
<td>A reference to the list of taken samples.</td>
</tr>
<tr>
<td>minValue</td>
<td>string</td>
<td>The expected minimal measure value.</td>
</tr>
<tr>
<td>maxValue</td>
<td>string</td>
<td>The expected maximum measure value.</td>
</tr>
<tr>
<td>stopTime</td>
<td>dateTime</td>
<td>The time from which the meter stops tracking samples.</td>
</tr>
<tr>
<td>expiresTime</td>
<td>dateTime</td>
<td>The time from which the Meter is not monitored anymore. It implies the deletion of the Meter after this time. Note that a Meter might be deleted before this time if the Resource being metered is deleted.</td>
</tr>
</tbody>
</table>

When implementing or using Meter, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 47 as well as in the tables describing related Collections.

5.17.3.1 Collections

The following clauses describe the Collection resources that are components of Meters.

5.17.3.1.1 SampleCollection Resource

The Resource type for each item of this Collection is "Sample", defined in Table 48:

Table 48 – Sample attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Sample">http://schemas.dmtf.org/cimi/2/Sample</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>timestamp</td>
<td>dateTime</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
</tr>
</tbody>
</table>
When implementing or using Sample, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 48 as well as in the tables describing related Collections.

5.17.3.2 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported via the MeterCollection Resource. The deletion of a Meter shall remove the Meter from the targetResource's "meter" attribute.

The following custom operations are also defined:

start

/link@rel: http://schemas.dmtf.org/cimi/2/action/start

This operation shall start a Meter.

Input parameters: None.

Output parameters: None.

Upon successful completion of this operation, the Meter shall start recording samples related to its associated Resource.

HTTP protocol

To start a Meter, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/start" URI of the Meter where the HTTP request body shall be as described below.

JSON media type: application/json

JSON serialization:

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
    "action": "http://schemas.dmtf.org/cimi/2/action/start",
    "properties": { "string": string, + } ?
...
}
```

XML media type: application/xml

XML serialization

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
    <action> http://schemas.dmtf.org/cimi/2/action/start </action>
    <properties>
        <property key="xs:string"> xs:string </property> *
    </properties> ?
    <xs:any>*
</Action>
```

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

stop

/link@rel: http://schemas.dmtf.org/cimi/2/action/stop
This operation shall stop a Meter.

Input parameters: None.
Output parameters: None.

Upon successful completion of this operation, the Meter shall no longer be recording samples related to its associated Resource.

HTTP protocol

To stop a Meter, a POST is sent to the "http://schemas.dmtf.org/cimi/2/action/stop" URI of the Meter where the HTTP request body shall be as described below.

**JSON media type:** application/json

**JSON serialization:**

```json
{
"resourceURI": "http://schemas.dmtf.org/cimi/2/Action",
"action": "http://schemas.dmtf.org/cimi/2/action/stop",
"properties": [ string: string, + ]
...}
```

**XML media type:** application/xml

**XML serialization**

```xml
<Action xmlns="http://schemas.dmtf.org/cimi/2">
  <action> http://schemas.dmtf.org/cimi/2/action/stop </action>
  <properties>
    <property key="xs:string"> xs:string </property> *
  </properties> ?
  <xs:any>*
</Action>
```

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

### 5.17.4 MeterCollection Resource

A MeterCollection Resource represents the Collection of Meters within a Provider and follows the Collection pattern defined in clause 5.5.12.

### 5.17.4.1 Operations

**NOTE** The "add" operation requires that a MeterTemplate be used (see 4.2.1.1).

If Meters are created through the global (Cloud Entry Point) MeterCollection's "add" operation, they shall be added automatically to the corresponding targetResource's "Meters" Collection Resource as well.

### 5.17.5 MeterTemplate Resource

A MeterTemplate represents the information needed to create a new Meter. Table 49 describes the MeterTemplate attributes.
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

Table 49 – MeterTemplate attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>MeterTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/MeterTemplate">http://schemas.dmtf.org/cimi/2/MeterTemplate</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>targetResource</td>
<td>ref</td>
</tr>
<tr>
<td>meterConfig</td>
<td>ref</td>
</tr>
</tbody>
</table>

When implementing or using MeterTemplate, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 49 as well as in the tables describing referred Resources or related Collections.

5.17.6 MeterTemplateCollection Resource

A MeterTemplateCollection Resource represents the Collection of MeterTemplate Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.17.6.1 Operations

This Resource supports the Read and Update operations. Creation of new MeterTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

5.17.7 MeterConfiguration Resource

A MeterConfiguration represents the definition of a Meter. Table 50 describes the MeterConfiguration attributes.

Table 50 – MeterConfiguration attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>MeterConfiguration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/MeterConfiguration">http://schemas.dmtf.org/cimi/2/MeterConfiguration</a></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
</tr>
<tr>
<td>associatedResources</td>
<td>URI[ ]</td>
</tr>
<tr>
<td>aspect</td>
<td>URI</td>
</tr>
<tr>
<td>units</td>
<td>string</td>
</tr>
<tr>
<td>sampleInterval</td>
<td>integer</td>
</tr>
</tbody>
</table>
### Table 51 – aspect URIs

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpu</td>
<td>The percentage CPU usage of the Resource. Typically associated with CloudEntryPoint, System, and Machine Resources. For Resources that group other Resources (e.g., CloudEntryPoint or System Resources), this aspect provides the aggregated percentage usage of the CPU.</td>
</tr>
<tr>
<td>memory</td>
<td>The amount of memory being used by the Resource. Typically associated with CloudEntryPoint, System, and Machine Resources. For Resources that group other Resources (e.g., CloudEntryPoint or System Resources), this aspect provides the aggregated usage of the memory.</td>
</tr>
<tr>
<td>disk</td>
<td>The amount of disk being used by the Resource. Typically associated with CloudEntryPoint, System, Machine, and Volume Resources. For Resources that group other Resources (e.g., CloudEntryPoint or System Resources), this aspect provides the aggregated disk usage.</td>
</tr>
<tr>
<td>bandwidth</td>
<td>The amount of network traffic. Typically associated with CloudEntryPoint, System, and Network Resources. For CloudEntryPoint and System Resources, this aspect provides the aggregated bandwidth of all the networks under them.</td>
</tr>
<tr>
<td>inputBandwidth</td>
<td>The amount of input bandwidth used by the Resource. Typically associated with Machine, NetworkPort, and Volume Resources. For Machine Resources, this aspect provides the aggregated input bandwidth usage of all its network interfaces.</td>
</tr>
<tr>
<td>outputBandwidth</td>
<td>The amount of output bandwidth used by the Resource. Typically associated with Machine, NetworkPort, and Volume Resources. For Machine Resources, this aspect provides the aggregated output bandwidth usage of all its network interfaces.</td>
</tr>
</tbody>
</table>

### 5.17.7.1 Operations

This Resource supports the Read, Update, and Delete operations. Create is supported through the MeterConfigurationCollection Resource.
5.17.8 EventLog Resource

A Resource that represents a registry of Events.

If an EventLog’s "targetResource" is deleted, the EventLog associated with that Resource may also be deleted. In other words, deleting a Resource (e.g., a Machine) may also result in the deletion of the EventLog referenced from that Resource. This behavior is denoted by the EventLog "Linked" capability.

If an EventLog is deleted, all of its Events shall also be deleted.

Table 52 describes the EventLog attributes.

5.17.9 MeterConfigurationCollection Resource

A MeterConfigurationCollection Resource represents the Collection of MeterConfigurations within a Provider and follows the Collection pattern defined in clause 5.5.12.

5.17.9.1 Operations

This Resource supports the Read and Update operations. Creation of new MeterConfiguration Resources is supported by way of a POST to the "add" operation’s URI as described in clause 4.2.1.1.

Table 52 – EventLog attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>EventLog</td>
<td><a href="http://schemas.dmtf.org/cimi/2/EventLog">http://schemas.dmtf.org/cimi/2/EventLog</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetResource</td>
<td>ref</td>
<td>A reference to the Resource to which the Events are related.</td>
</tr>
<tr>
<td>Events</td>
<td>collection {Event}</td>
<td>A reference to the list of occurred Events.</td>
</tr>
<tr>
<td>Persistence</td>
<td>string</td>
<td>A value that indicates the persistence of the Events within the EventLog, for instance, daily, weekly, monthly, or yearly. Events that exceed the persistence duration may be deleted.</td>
</tr>
<tr>
<td>Name</td>
<td>EventLog</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/EventLog">http://schemas.dmtf.org/cimi/2/EventLog</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>&lt;unnamed structure&gt;</td>
<td>A summary of all the events present in the EventLog when the read operation is performed, grouped by severity. Each summary attribute is an (unnamed) structure that has the following subattributes:</td>
</tr>
</tbody>
</table>

<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>

When implementing or using EventLog, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 52 as well as in the tables describing embedded Resources or related Collections.

### 5.17.9.2 Collections

The following clauses describe the Collection Resources EventLogs.

#### 5.17.9.2.1 events Collection

The Resource type for each item of this Collection is "Event" as defined in clause 5.17.13.

### 5.17.9.3 Operations

This Resource supports the Read, Update, and Delete operations.

#### 5.17.10 EventLogCollection Resource

An EventLogCollection Resource represents the Collection of EventLogs within a Provider and follows the Collection pattern defined in clause 5.5.12.

#### 5.17.11 EventLogTemplate Resource

An EventLogTemplate represents the information needed to create a new EventLog. Table 53 describes the EventLogTemplate attributes.

### Table 53 – EventLogTemplate attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>EventLogTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/EventLogTemplate">http://schemas.dmtf.org/cimi/2/EventLogTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetResource</td>
<td>ref</td>
<td>A reference to the Resource to which the EventLog shall be connected.</td>
</tr>
</tbody>
</table>
Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol

<table>
<thead>
<tr>
<th>Name</th>
<th>EventLogTemplate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/EventLogTemplate">http://schemas.dmtf.org/cimi/2/EventLogTemplate</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>persistence</td>
<td>string</td>
<td>A value that indicates the persistence of the Events in the new EventLog, for instance, daily, weekly, monthly, or yearly. Events that exceed the persistence duration may be deleted.</td>
</tr>
</tbody>
</table>

When implementing or using EventLogTemplate, Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 53 as well as in the tables describing referred Resources or related Collections.

### 5.17.12 EventLogTemplateCollection Resource

An EventLogTemplateCollection Resource represents the Collection of EventLogTemplate Resources within a Provider and follows the Collection pattern defined in clause 5.5.12.

### 5.17.12.1 Operations

This Resource supports the Read and Update operations. Creation of new EventLogTemplate Resources is supported by way of a POST to the "add" operation's URI as described in clause 4.2.1.1.

### 5.17.13 Event Resource

A Resource that represents the occurrence of an event within the managed infrastructure. Some examples of Event are:

- 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 information that the Provider is able to track within its infrastructure and that can constitute useful information for the Consumer. Possible examples include, but are not limited to, errors and inconveniences that occur in the (virtual) resources assigned to Consumers; Provider-initiated actions, such as maintenance tasks; etc.

Table 54 describes the Event attributes.

<table>
<thead>
<tr>
<th>Name</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type URI</td>
<td><a href="http://schemas.dmtf.org/cimi/2/Event">http://schemas.dmtf.org/cimi/2/Event</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timestamp</td>
<td>date Time</td>
<td>The time of occurrence of the actual Event. NOTE: This attribute should not be confused with the time of creation of the Event Resource instance, which is captured in the common &quot;created&quot; attribute.</td>
</tr>
</tbody>
</table>

Table 54 – Event attributes
<table>
<thead>
<tr>
<th>Name</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type URI</strong></td>
<td><a href="http://schemas.dmtf.org/cimi/2/Event">http://schemas.dmtf.org/cimi/2/Event</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>URI</td>
<td>A URI that uniquely identifies the type of the Event. If the &quot;content&quot; attribute is present, this URI determines the actual data structure used for this content, e.g., to which schema it is associated.</td>
</tr>
</tbody>
</table>
| content   | any  | A polymorphic attribute that represents detailed event data, the type of which varies with the Event "type." Typically, a data structure; for example:  
In the case of a monitoring event, the content shall hold the target Resource ID and type, measured attribute(s), and status value(s).  
In the case of an audit event conforming to the CADF model, the content shall hold the detailed event structure that complies with CADF event schema.  
In the case of a CIM Indication, the content shall hold the structure and attributes defined for such events. |
| outcome   | string | A string value that characterizes the general significance of the Event. A core set is defined that may be used regardless of the Event type. For each Event type, the definition of a core outcome value may be refined in the context of this type, provided it does not conflict with the general meaning of the outcome given below.  
Core outcomes are:  
**Pending:** The Event is about an action or process that is still ongoing.  
**Unknown:** The Event is about a request or action that is not known by the Provider.  
**Status:** The Event reports on the state or status of a Resource.  
**Success:** The Event reports on a successful outcome of some action or process.  
**Warning:** The Event reports on a situation that requires attention or remedial action.  
**Failure:** The Event reports on a failed outcome of some action or process.  
This set of core outcome values may be extended to accommodate possible outcomes of a specific Event type. In this case, the extended set of values shall apply to all Events of this type. |
| severity  | string | A value indicating the Event severity. Possible values are:  
**critical**  
**high**  
**medium**  
**low**  
The meaning of the severity level may vary depending on the Event "type." If such an attribute is not relevant to a particular type of Event, it should be omitted. |
| contact   | string | A reference to a contact point or processing point to handle the Event. The actual type of this content (e.g., email address, phone number of helpdesk or staff, message queue, URL...) is dependent on, and determined by the Event "type." This attribute is mutable as it may be determined after Event creation by the Provider. |

---

3970 | NOTE There exists a legacy of several Event models that have been standardized or designed for various domains relevant to IT. The objective in CIMI is not to elect one particular Event model, but to select as top-level attributes the most immediately relevant data useful for Event processing in a cloud environment. Additional Event data may still be represented in the variable content attribute that allows for mapping other Event models into a CIMI Event. |
When implementing or using Event Providers and Consumers shall adhere to the syntax and semantics of its attributes as described in Table 54.

Table 55 describes the "type" URIs that are defined or acknowledged by this specification. Additional types may be added by a Provider, for example to characterize external events mapped into CIMI Events. It is recommended that these URIs be dereferencable such that Consumers can discover a more detailed description of the type. Event types defined by this specification share the same base URI: http://schemas.dmtf.org/cimi/2/event/. For brevity, if the "Event Type" column in the table only shows a relative URI (e.g., "state"), it shall be appended to the end of this base URI.

Table 55 – type URIs

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Events of this type report state information about CIMI run-time resources such as instances of Machines, Systems, Networks, and Volumes. This information includes reports on any change in the &quot;state&quot; of these Resources. The content element associated with this Event type has the following structure:</td>
</tr>
<tr>
<td></td>
<td>Data</td>
</tr>
<tr>
<td></td>
<td>resName</td>
</tr>
<tr>
<td></td>
<td>resource</td>
</tr>
<tr>
<td></td>
<td>resType</td>
</tr>
<tr>
<td></td>
<td>state</td>
</tr>
<tr>
<td></td>
<td>previous</td>
</tr>
<tr>
<td>alarm</td>
<td>Events of this type report errors or alarms occurring during management operations of cloud resources. This information includes failures to provision resources, failures to fulfill requests to the CIMI interface, and any critical situation that needs be addressed in a timely manner. The content element associated with this event type has the following structure:</td>
</tr>
<tr>
<td></td>
<td>Data</td>
</tr>
<tr>
<td></td>
<td>resName</td>
</tr>
<tr>
<td></td>
<td>resource</td>
</tr>
<tr>
<td></td>
<td>restype</td>
</tr>
<tr>
<td></td>
<td>code</td>
</tr>
<tr>
<td></td>
<td>detail</td>
</tr>
</tbody>
</table>
### Event Type | Description
--- | ---
**model** | **Events** of this type report changes in the CIMI resource model, which includes creation, modification, and destruction of Resource instances; and updates to metadata (Resource extensions, capabilities and constraints, etc.).
The **content** element associated with this event type has the following structure:

<table>
<thead>
<tr>
<th>Data</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resName</td>
<td>string</td>
<td>The name of the main model Resource affected by the modification.</td>
</tr>
<tr>
<td>resource</td>
<td>ref</td>
<td>The reference to the main model Resource affected by the modification.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Note: This reference may become invalid because the event might outlive the Resource.)</td>
</tr>
<tr>
<td>resType</td>
<td>URI</td>
<td>URI denoting this Resource type (same as the type URI associated with the Resource type for this Resource).</td>
</tr>
<tr>
<td>change</td>
<td>string</td>
<td>The kind of modification reported (create/update/delete).</td>
</tr>
<tr>
<td>detail</td>
<td>string</td>
<td>The detailed information associated with the change, typically the data for an update or creation, as used in a request.</td>
</tr>
</tbody>
</table>

**access** | **Events** of this type keep track of all requests to access some Resource of a CIMI provider.
The **content** element associated with this event type has the following structure:

<table>
<thead>
<tr>
<th>Data</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation</td>
<td>string</td>
<td>The method or name of the operation intended for this access (for the HTTP protocol, the HTTP method for the request).</td>
</tr>
<tr>
<td>resource</td>
<td>ref</td>
<td>The reference of the Resource supporting the operation (for the HTTP protocol, the Resource URI or the URI associated with the operation). (Note: This reference may become invalid because the event might outlive the Resource.)</td>
</tr>
<tr>
<td>detail</td>
<td>string</td>
<td>The detailed information associated with the change, typically the data for an update or creation, as used in a request.</td>
</tr>
<tr>
<td>initiator</td>
<td>string</td>
<td>The details identifying the request initiator, in case that information can be associated with the request.</td>
</tr>
</tbody>
</table>

http://schemas.dmtf.org/cloud/audit/1.0/ | Events of this type represent events that have audit significance, as defined by CADF (...). This type can be subdivided further by extending the URI path (e.g., http://schemas.dmtf.org/cloud/audit/1.0/event/security, for security audit events). The **content** element associated with this event type has the same structure as the event serialization defined in CADF ([DSP0262]).

### 5.17.13.1 Operations
This resource supports the Read, Update, and Delete operations.
6 Security considerations

There are many security mechanisms that can be used in conjunction with this specification. This specification does not mandate any particular mechanism. Providers shall provide enough information about their security mechanisms so that the Consumer can implement the necessary algorithms to successfully communicate with the Provider.

An implementation may set limits on:

- The length of attribute values it accepts.
- The size of arrays it accepts.
- The size of the request body or the length of request URIs it accepts.

These limits may not all be advertised in the ResourceMetadata, although this specification recommends Providers to do so. A Provider that receives a request that exceeds any of these limits, shall return a response with an appropriate standard HTTP status code.

7 Conformance

This clause describes a minimal set of features that a Cloud Provider must implement to be in conformance with the specification.

This does not preclude a implementing additional features and is not exclusive of other levels of conformance that may be defined outside of this document.

The goal is to specify a basic set of features upon which implementations may rely that provides useful functionality and aids interoperability without making onerous demands on Cloud Provider implementations.

7.1 Minimal conformance clause

A Cloud Provider implementation is in minimal conformance with the specification if it satisfies all of the following requirements:

- It implements the **Machine** Resource specified in clause 5.14 “Machine Resources and relationships”, along with its mandatory (providerMandatory=true) common attributes, and at least the following attributes: cpu, memory, disks, cpuArch, cpuSpeed,

- It implements the **MachineImage** Resource specified in clause 5.14.7 “MachineImage Resource”, along with its mandatory (providerMandatory=true) common attributes, and at least the following attributes: imageLocation,

- It implements the **MachineConfiguration** Resource specified in clause 5.14.5 “MachineConfiguration Resource” along with its mandatory (providerMandatory=true) common attributes, and at least the following attributes: cpu, memory, disks, cpuArch, cpuSpeed, in addition to mandatory common attributes,

- It implements **ResourceMetadata** ResourceMetadata specified in clause 5.11 “Resource Metadata”, with at least the attributes: typeURI, name, attributes, and all the fields in the attribute data type except for consumerMandatory. The minimal support required for ResourceMetadata is only for discovery via the CEP. No access is required from any other Resource i.e., no ResourceMetadata reference is required in any other Resource.

- It supports the creation of Machine Resources with template data passed by value, as specified in clause 4.2.1.1 “Creating a new Resource”, i.e., is able to process a Machine creation request
where the Machine template is passed by value. No support for the MachineTemplate Resource is required.


- It implements the **CEP** Resource as specified in clause 5.12 “Cloud Entry Point”, with the following collection attributes: resourceMetadata, machines, machineImages, machineConfigs.

- For all the above Resources, it provides at least read-only access to their attributes, and at least the create and delete operations.

- It handles requests and generates responses according to protocol requirements as specified in clause 4.2 “Protocol operations”.

- It handles content serialization in requests and serializes content in generated messages as specified in clause 5.5 “Data types and their serialization”.

This annex defines how elements of an OVF descriptor are mapped to CIMI resources and their attributes. This definition allows the import of an OVF package to create multiple CIMI resources. This is done by specifying a reference to an OVF package in the import operation of a SystemCollection or SystemTemplateCollection (the Media Type at that URI shall be "application/ovf"). Refer to DSP0243 for more information about OVF.

Support for OVF import and export is optional for a Provider and it is an implementation choice as to how many of the attributes in the OVF package are exposed through CIMI resources. A Provider may support the import of OVF package for only Systems, only SystemTemplates or both. Support for the actual import and export of an OVF package is handled by a hypervisor under the management of the CIMI implementation, and thus the CIMI resources that are created reflect what the hypervisor did upon import and form a “View” into the results.

The import of an OVF package can be reflected in the creation of Templates that can be later used to create Systems, Machines and other component Resources. The import of an OVF package can also be used to directly create Systems, Machines, and other component Resources, bypassing the step of creating Templates.

Clause 5.13.5 details how to import an OVF file to create a SystemTemplate (and component Resources). The SystemTemplate thus created contains a reference to a MachineTemplate for every VirtualSystem that is defined in the OVF descriptor VirtualSystemCollection. Note that CIMI currently allows Systems of Systems, so for each VirtualSystemCollection encountered in a nested set of collections, a separate SystemTemplate is created within the parent SystemTemplate with MachineTemplates for each of the contained VirtualSystems in that VirtualSystemCollection.

The values of the attributes for the MachineTemplate are taken from the VirtualHardwareSection of the VirtualSystem description (required in OVF). If more than one VirtualHardwareSection is used for a given VirtualSystem (allowed in OVF), the result is implementation dependent, but the implementation might choose a MachineTemplate from an existing (perhaps static) set that best matches a VirtualHardwareSection. Items in the VirtualHardwareSection are mapped to CIMI MachineConfiguration properties and the corresponding MachineConfiguration Resource is created and linked to from the created MachineTemplate for that VirtualSystem.

The CIMI VolumeTemplates are created according to the DiskSection of an OVF descriptor and can be shared among more than one VirtualSystem (CIMI MachineTemplates) defined in an OVF package. In addition, a new CIMI MachineImage Resource may be created from the DiskSection if an ovf:fileRef for the virtual disk content is specified.

The CIMI NetworkTemplates are created according to the NetworkSection of an OVF descriptor along with the Connection elements in the VirtualHardwareSection elements that refer to these named networks.

Clause 5.13.2.1 details how to import an OVF file to create a System (and component Resources). The System thus created contains a reference to a Machine for every VirtualSystem that is defined in an OVF descriptor VirtualSystemCollection. Note that CIMI currently allows Systems of Systems, so for each VirtualSystemCollection encountered in a nested set of collections, a separate System is created within the parent System with Machines for each of the contained VirtualSystems in that VirtualSystemCollection.
The values of the attributes for the Machine are taken from the VirtualHardwareSection of the VirtualSystem description (required in OVF). If more than one VirtualHardwareSection is used for a given VirtualSystem (allowed in OVF), the result is implementation dependent. Items in the VirtualHardwareSection are mapped to CIMI MachineConfiguration properties and the corresponding MachineConfiguration Resource is created and linked to from the created Machine for that VirtualSystem.

The CIMI Volumes are created according to the DiskSection of an OVF descriptor and can be shared among more than one VirtualSystem (CIMI Machines) defined in an OVF package. In addition, a new CIMI MachineImage Resource may be created from the DiskSection if an ovf:fileRef attribute for the virtual disk content is specified.

The CIMI Networks are created according to the NetworkSection of an OVF descriptor along with the Connection elements in the VirtualHardwareSection that refer to these named networks.
ANNEX B
(normative)
XML Schema

The XML Schema for the XML serialization of the CIMI model can be found at:

http://schemas.dmtf.org/cimi/2/dsp8009_1.0.xsd

The schema provided does not intend to reflect every single modeling constraint and requirement specified in the model. This schema is designed to apply more broadly to any model-related serialized material found in Consumer requests as well as in Provider responses, and is intended to provide a preliminary, nonexhaustive syntactic check on these. In particular, future updates of this specification may intermix new XML elements into the Resources using the current CIMI namespace to Resources. The schema that is provided is just a starting point for those who would find it useful and it might need to be modified based on specific application's needs.
ANNEX C
(normative)
Change log

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0.0</td>
<td>2012-08-28</td>
<td>DMTF Standard</td>
</tr>
<tr>
<td>1.0.1</td>
<td>2012-09-12</td>
<td>DMTF Standard</td>
</tr>
<tr>
<td>1.1.0</td>
<td>2013-10-22</td>
<td>DMTF Standard</td>
</tr>
<tr>
<td>2.0.0</td>
<td>2016-07-27</td>
<td>DMTF Standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Resolve multiple Mantis issues</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Improve the usability of the network template</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Apply various editorial comments</td>
</tr>
</tbody>
</table>
Bibliography

DMTF Standard: *Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol* specification V1.0 (DSP0263)

[http://dmtf.org/sites/default/files/standards/documents/DSP0263_1.0.0.pdf](http://dmtf.org/sites/default/files/standards/documents/DSP0263_1.0.0.pdf)

DMTF Standard: *Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol* specification V1.1 (DSP0263)

[https://members.dmtf.org/apps/org/workgroup/cmwg/download.php/73648/DSP0263_1.1.0b_RC2.pdf](https://members.dmtf.org/apps/org/workgroup/cmwg/download.php/73648/DSP0263_1.1.0b_RC2.pdf)