Document Number: DSP-IS0201

Date: 2011-04-15

Version: 1.0.0

CIM Operations Over RESTful Services

Document Type: Specification

Document Status: DMTF Informational Specification

Document Language: en-US

IMPORTANT: This specification is not a standard. It is an exploratory, informational document developed in order to obtain industry feedback. It does not reflect the views of the DMTF or all of its members. It is possible that future standards may or may not consider this work product to be an input in whole or in part.

CONTENTS

Foreword 8
Introduction 9
Document conventions 9
Typographical conventions 9
ABNF usage conventions 9
Experimental material 9
1 Scope 11
2 Normative references 11
3 Terms and definitions 12
4 Symbols and abbreviated terms 15
5 Concepts 17
5.1 REST architectural style supported by CIM-RS 17
6 Conformance 18
7 REST resources and their resource identifiers 19
7.1 Structure of WBEM resource identifiers 19
7.1.1 General structure 19
7.1.2 Percent-encoding 20
7.1.3 Scheme 22
7.1.4 Authority 22
7.1.5 Client knowledge about URI structure 22
7.2 REST resources and their URIs 22
7.2.1 Namespace collection resource 23
7.2.2 Namespace resource 23
7.2.3 Class collection resource 23
7.2.4 Class resource 23
7.2.5 Class associator collection resource 24
7.2.6 Class reference collection resource 24
7.2.7 Class method invocation resource 24
7.2.8 Instance collection resource 24
7.2.9 Instance resource 24
7.2.10 Instance associator collection resource 25
7.2.11 Instance reference collection resource 25
7.2.12 Instance method invocation resource 25
7.2.13 Qualifier type collection resource 25
7.2.14 Qualifier type resource 25
7.2.15 Instance query resource 25
7.3 Query parameters in URIs 26
7.3.1 acn (associated class name) 26
7.3.2 arn (associated role name) 27
7.3.3 c (class) 27
7.3.4 dd (delete dependents) 28
7.3.5 esbp (exclude subclass properties) 28
7.3.6 fql (filter query language) 29
7.3.7 fqs (filter query string) 29
7.3.8 ico (include class origin) 30
7.3.9 iie (include inherited elements) 31
7.3.10 ip (included properties) 31
7.3.11 iq (include qualifiers) 32
7.3.12 isbc (include subclasses) 32
7.3.13 n (namespace) 33
7.3.14 pr (paged retrieval) 33

7.3.15 q (qualifier) 34
7.3.16 rcn (referencing class name) 35
7.3.17 spc (superclass) 35
7.3.18 srn (source role name) 36
8 Operations 36
8.1 Overview on REST resources and HTTP methods 36
8.2 Common behaviors for all operations 37
8.2.1 Content negotiation 37
8.2.2 Links 38
8.2.3 Verifying the basis of resource modifications 39
8.2.4 Caching of responses 39
8.2.5 Success and failure 40
8.2.6 Error messages 40
8.2.7 Consistency model 40
8.2.8 Common operation parameters for all operations 40
8.3 Optional behaviors for CIM-RS protocol 40
8.3.1 Entity tagging feature 40
8.3.2 Paged retrieval feature 40
8.4 Protocol payload elements 42
8.4.1 NamespaceCollection payload element 42
8.4.2 Namespace payload element 43
8.4.3 ClassCollection payload element 43
8.4.4 Class payload element 44
8.4.5 InstanceCollection payload element 44
8.4.6 Instance payload element 45
8.4.7 QualifierTypeCollection payload element 46
8.4.8 QualifierType payload element 46
8.4.9 MethodInvocationRequest payload element 47
8.4.10 MethodInvocationResponse payload element 47
8.4.11 InstanceModificationRequest payload element 48
8.4.12 InstanceQueryRequest payload element 48
8.4.13 ErrorResponse payload element 49
8.5 WBEM server and listener operations 49
8.5.1 OPTIONS 49
8.6 Namespace collection operations 50
8.6.1 GET namespace collection 50
8.7 Namespace operations 52
8.7.1 GET namespace 52
8.8 Class collection operations 53
8.8.1 GET class collection 53
8.8.2 POST class collection 56
8.9 Class operations 57
8.9.1 GET class 57
8.9.2 DELETE class 58
8.9.3 PUT class 59
8.10 Class associator collection operations 59
8.10.1 GET class associator collection 59
8.11 Class reference collection operations 61
8.11.1 GET class reference collection 61
8.12 Class method operations 63
8.12.1 POST class method 63
8.13 Instance collection operations 64
8.13.1 GET instance collection 64
8.13.2 POST instance collection 66
8.14 Instance operations 67
8.14.1 GET instance 67

8.14.2 DELETE instance ... 68
8.14.3 PUT instance ... 69
8.14.4 PATCH instance ... 70
8.15 Instance associator collection operations ... 72
8.15.1 GET instance associator collection ... 72
8.16 Instance reference collection operations ... 74
8.16.1 GET instance reference collection ... 74
8.17 Instance method operations ... 76
8.17.1 POST instance method ... 76
8.18 Qualifier type collection operations ... 77
8.18.1 GET qualifier type collection ... 77
8.18.2 POST qualifier type collection ... 78
8.19 Qualifier type operations ... 79
8.19.1 GET qualifier type ... 79
8.19.2 DELETE qualifier type ... 80
8.19.3 PUT qualifier type ... 81
8.20 Instance query operations ... 82
8.20.1 POST instance query ... 82
8.21 Functionality without specific operations ... 83
8.21.1 Subscription and other indication related functions ... 83
8.22 Potential future operations ... 83
8.22.1 Indication delivery ... 83
9 Usage of HTTP and HTTPS ... 84
9.1 HTTP and HTTPS version requirements ... 84
9.2 Authentication requirements ... 84
9.2.1 Operating without authentication ... 84
9.2.2 HTTP basic authentication ... 84
9.2.3 HTTP digest authentication ... 85
9.2.4 Other authentication mechanisms ... 85
9.3 Message encryption ... 85
9.4 HTTP header fields ... 85
9.4.1 Accept ... 85
9.4.2 Content-Type ... 86
9.4.3 Etag ... 86
9.4.4 If-Match ... 87
9.4.5 CIMRS-Content-Types ... 87
9.4.6 CIMRS-Entity-Tagging-Feature ... 87
9.4.7 CIMRS-Paged-Retrieval-Feature ... 88
9.4.8 CIMRS-Filter-Query-Languages ... 88
9.4.9 CIMRS-Instance-Query-Languages ... 88
10 Payload representation ... 89
10.1 Media type ... 89
10.2 Payload element representations ... 89
ANNEX A (informative) Known payload representations ... 91
ANNEX B (informative) Examples for structure of resource URIs ... 92
B.1 Example using CIM object types as URI segments ... 92
B.1.1 Namespace collection resource ... 92
B.1.2 Namespace resource ... 92
B.1.3 Class collection resource ... 93
B.1.4 Class resource ... 93
B.1.5 Class associator collection resource ... 93
B.1.6 Class reference collection resource ... 94
B.1.7 Class method invocation resource ... 94
B.1.8 Instance collection resource ... 94
B.1.9 Instance resource ... 94

B.1.10 Instance associator collection resource 96
B.1.11 Instance reference collection resource 96
B.1.12 Instance method invocation resource 96
B.1.13 Qualifier type collection resource 97
B.1.14 Qualifier type resource 97
B.1.15 Instance query resource 97
B.2 Example using WBEM URIs 98
ANNEX C (informative) Change log 99
Bibliography 100

Tables

Table 1 – REST resources and HTTP methods 36
Table 2 – Links included in payload elements 38
Table 3 – Operations supporting paged retrieval 41
Table 4 – Additional links for paged retrieval 42
Table 5 – Properties of NamespaceCollection payload element 42
Table 6 – Links of NamespaceCollection payload element 42
Table 7 – Properties of Namespace payload element 43
Table 8 – Links of Namespace payload element 43
Table 9 – Properties of ClassCollection payload element 43
Table 10 – Links of ClassCollection payload element 43
Table 11 – Properties of Class payload payload element 44
Table 12 – Links of Class payload element 44
Table 13 – Properties of InstanceCollection payload element 45
Table 14 – Links of InstanceCollection payload element 45
Table 15 – Properties of Instance payload element 45
Table 16 – Links of Instance payload element 46
Table 17 – Properties of QualifierTypeCollection payload element 46
Table 18 – Links of QualifierTypeCollection payload element 46
Table 19 – Properties of QualifierType payload element 47
Table 20 – Links of QualifierType payload element 47
Table 21 – Properties of MethodInvocationRequest payload element 47
Table 22 – Properties of MethodInvocationResponse payload element 48
Table 23 – Properties of InstanceModificationRequest payload element 48
Table 24 – Properties of PropertyValue generic type 48
Table 25 – Properties of InstanceQueryRequest payload element 49
Table 26 – Properties of ErrorResponse payload element 49
Table 27 – Response header fields for OPTIONS to WBEM server 50
Table 28 – Response header fields for OPTIONS to WBEM listener 50
Table 29 – Query parameters for GET namespace collection 51
Table 30 – Query parameters for GET namespace 53
Table 31 – Query parameters for GET class collection 54
Table 32 – Query parameters for GET class 57
Table 33 – Query parameters for DELETE class 58
Table 34 – Query parameters for GET class associator collection 60
Table 35 – Query parameters for GET class reference collection 62

Table 36 – Query parameters for GET instance collection 65
Table 37 – Query parameters for GET instance 67
Table 38 – Query parameters for GET instance associator collection 72
Table 39 – Query parameters for GET instance reference collection 74
Table 40 – Query parameters for GET qualifier type collection 77
Table 41 – Known CIM-RS payload representations 91

Foreword

The *CIM Operations Over RESTful Services* (DSP-IS0201) informational specification was prepared by the DMTF CIM-RS Incubator.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. For information about the DMTF, see http://www.dmtf.org.

Acknowledgments

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

- Andreas Maier, IBM (editor)
- Cornelia Davis, EMC
- George Ericson, EMC
- Johannes Holzer, IBM
- Robert Kieninger, IBM
- Larry Lamers, VMware
- Marvin Waschke, Computer Associates

Introduction

The information in this document should be sufficient for a WBEM server, WBEM client, and WBEM listener to unambiguously identify the protocol interactions that shall be performed to implement the CIM-RS protocol.

NOTE: This version of this document describes the WBEM listener and WBEM server in their roles related to indication delivery, but it does not yet describe the indication delivery as a protocol interaction.

The target audience for this specification is implementers who are writing WBEM servers, WBEM clients, and WBEM listeners supporting the CIM-RS protocol.

Document conventions

Typographical conventions

The following typographical conventions are used in this document:

- Document titles are marked in *italics*.
- Important terms that are used for the first time are marked in *italics*.
- Terms include a link to the term definition in the "Terms and definitions" clause, enabling easy navigation to the term definition.
- ABNF rules are in `monospaced font`.

ABNF usage conventions

Format definitions in this document are specified using ABNF (see RFC5234), with the following deviations:

- Literal strings are to be interpreted as case-sensitive Unicode characters, as opposed to the definition in RFC5234 that interprets literal strings as case-insensitive US-ASCII characters.

The following ABNF rules may be used in any ABNF in this document:

```
WS = *( U+0020 / U+0009 / U+000A )   ; zero or more white space characters
```

Experimental material

Experimental material has yet to receive sufficient review to satisfy the adoption requirements set forth by the DMTF. Experimental material is included in this document as an aid to implementers who are interested in likely future developments. Experimental material may change as implementation experience is gained. It is likely that experimental material will be included in an upcoming revision of the document. Until that time, experimental material is purely informational.

The following typographical convention indicates experimental material:

EXPERIMENTAL

Experimental material appears here.

EXPERIMENTAL

In places where this typographical convention cannot be used (for example, tables or figures), the "EXPERIMENTAL" label is used alone.

CIM Operations Over RESTful Services

1 Scope

This informational specification describes the *CIM Operations Over RESTful Services* protocol (also referred to as the *CIM-RS* protocol), which defines the use of RESTful services as a communications protocol between WBEM client, WBEM server, and WBEM listener.

The semantic of the operations defined for the CIM-RS protocol conforms to the generic operations defined in DSP0223. Because the current version of DSP0223 does not yet cover indications, this version of this document also does not support indications.

The CIM-RS protocol can be used with HTTP and HTTPS.

The CIM-RS protocol supports multiple payload representations. These payload representations are expected to be described in separate documents. Their use within the CIM-RS protocol is determined through HTTP content negotiation.

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 DSP0004, *CIM Infrastructure Specification 2.6,*
http://www.dmtf.org/standards/published_documents/DSP0004_2.6.pdf

DMTF DSP0223, *Generic Operations 1.0,*
http://www.dmtf.org/standards/published_documents/DSP0223_1.0.pdf

DMTF DSP0228, *Message Registry XML Schema 1.1*,
http://schemas.dmtf.org/wbem/messageregistry/1/dsp0228_1.1.xsd

DMTF DSP1033, *Profile Registration Profile 1.0,*
http://www.dmtf.org/standards/published_documents/DSP1033_1.0.pdf

DMTF DSP8016, *WBEM Operations Message Registry 1.0*,
http://schemas.dmtf.org/wbem/messageregistry/1/dsp8016_1.0.xml

ECMA-262, *ECMAScript Language Specification, 5th Edition, December 2009*,
http://www.ecma-international.org/publications/standards/Ecma-262.htm

IETF RFC2616, *Hypertext Transfer Protocol – HTTP/1.1*, June 1999,
http://tools.ietf.org/html/rfc2616

IETF RFC2617, *HTTP Authentication: Basic and Digest Access Authentication*, June 1999,
http://tools.ietf.org/html/rfc2617

IETF RFC2818, *HTTP Over TLS*, May 2000,
http://tools.ietf.org/html/rfc2818

IETF RFC3986, *Uniform Resource Identifier (URI): Generic Syntax*, January 2005,
http://tools.ietf.org/html/rfc3986

IETF RFC4346, *The Transport Layer Security (TLS) Protocol, Version 1.1*, April 2006,
http://tools.ietf.org/html/rfc4346

IETF RFC4648, *The Base16, Base32, and Base64 Data Encodings*, October 2006,
http://tools.ietf.org/html/rfc4648

IETF RFC5005, *Feed Paging and Archiving*, September 2007,
http://tools.ietf.org/html/rfc5005

IETF RFC5234, *Augmented BNF for Syntax Specifications: ABNF*, January 2008,
http://tools.ietf.org/html/rfc5234

IETF RFC5246, *The Transport Layer Security (TLS) Protocol, Version 1.2*, August 2008,
http://tools.ietf.org/html/rfc5246

IETF RFC5789, *PATCH Method for HTTP*, March 2010,
http://tools.ietf.org/html/rfc5789

ISO/IEC 10646:2003, *Information technology -- Universal Multiple-Octet Coded Character Set (UCS)*,
http://standards.iso.org/ittf/PubliclyAvailableStandards/c039921_ISO_IEC_10646_2003(E).zip

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards (2004, 5th edition)*,
http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse

The Unicode Consortium, *The Unicode Standard, Version 5.2.0, Annex #15: Unicode Normalization Forms*,
http://www.unicode.org/reports/tr15/

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 DSP0004 and DSP0223 apply to this document. The following additional terms are used in this document.

3.1
HTTP basic authentication
a simple authentication scheme for use by HTTP and HTTPS that is based on providing credentials in HTTP header fields. It is defined in RFC2617.

3.2
HTTP content negotiation
a method for selecting a representation of content in an HTTP response when there are multiple representations available. It is defined in section 12 of RFC2616. Its use in the CIM-RS protocol is described in 8.2.1.

3.3
HTTP digest authentication
an authentication scheme for use by HTTP and HTTPS that is based on verifying shared secrets that are not exchanged. It is defined in RFC2617.

3.4
HTTP entity body
the payload within an HTTP message, as defined in section 7.2 of RFC2616

3.5
HTTP entity header
an HTTP header field used in both HTTP request and/or HTTP response. Also called "entity header field".

3.6
HTTP header field
a named value used in the header of HTTP messages, as defined in section 4.2 of RFC2616. Also called "HTTP header".

3.7
HTTP message
an interaction between an HTTP client and an HTTP server (in any direction), as defined in section 4 of RFC2616

3.8
HTTP method
the type of interaction stated in an HTTP request, as defined in section 5.1.1 of RFC2616

3.9
HTTP request
an HTTP message sent from an HTTP client to an HTTP server as defined in section 5 of RFC2616. Also called "HTTP request message".

3.10
HTTP request header
an HTTP header field used in an HTTP request. Also called "request header field".

3.11
HTTP response
an HTTP message sent from an HTTP server to an HTTP client, as defined in section 6 of RFC2616. Also called "HTTP response message".

3.12
HTTP response header
an HTTP header field used in an HTTP response. Also called "response header field".

3.13
interop namespace
a namespace containing CIM instances representing specific capabilities of a WBEM server, such as for example, CIM_Namespace instances representing the namespaces of the WBEM server. For details, see DSP1033.

3.14
link
a WBEM resource identifier that appears in an HTTP entity body. For details, see 8.2.2.

3.15
MIME media type
a string identification for media types in Internet protocols, as defined in RFC2045 and RFC2046

3.16
Normalization Form C
a normalization form for UCS characters that avoids the use of combining marks where possible and that allows comparing UCS character strings on a per-code-point basis. It is defined in *The Unicode Standard, Annex #15*.

3.17
REST architectural style
the architectural style described in *Architectural Styles and the Design of Network-based Software Architectures*, chapter 5, and in *REST APIs must be hypertext driven*

3.18
REST resource
any entity that is accessed through and targeted by a RESTful service

3.19
RESTful service
a service accessible using REST concepts

3.20
UCS character
a character from the Universal Character Set defined in ISO/IEC 10646:2003

3.21
UCS code position
a numeric identification for a UCS character in the range of 0x0 to 0x10FFFF, as defined in ISO/IEC 10646:2003. See also DSP0004 for an overview.

3.22
Unicode character
an alternate term for UCS character

3.23
WBEM client
a CIM client (see DSP0004) that supports a WBEM protocol. A WBEM client originates WBEM operations for processing by a WBEM server. For details, see DSP0223.

3.24
WBEM indication
an interaction within a WBEM protocol that is originated on a WBEM server and processed by a WBEM listener. For details, see DSP0223. This release of this document does not cover WBEM indications.

3.25
WBEM listener
a CIM listener (see DSP0004) that supports a WBEM protocol. A WBEM listener processes WBEM indications originated by a WBEM server. For details, see DSP0223.

3.26
WBEM operation
an interaction within a WBEM protocol that is originated by a WBEM client and processed by a WBEM server. For details, see DSP0223.

3.27
WBEM protocol
a communications protocol between WBEM client, WBEM server, and WBEM listener. A WBEM protocol defines how the WBEM operations and WBEM indications work on top of an underlying protocol layer (for example, HTTP, SOAP, or TCP). For details, see DSP0223.

3.28
WBEM resource identifier
a URI that addresses a REST resource in a WBEM server. Some but not all of those REST resources correspond to CIM objects in a WBEM server. Also called "resource identifier" in this document.

3.29
WBEM server
a CIM server (see DSP0004) that supports a WBEM protocol. A WBEM server processes WBEM operations originated by a WBEM client and originates WBEM indications for processing by a WBEM listener. For details, see DSP0223.

4 Symbols and abbreviated terms

The abbreviations defined in DSP0004 and DSP0223 apply to this document. The following additional abbreviations are used in this document.

4.1
ABNF
Augmented Backus-Naur Form, as defined in RFC5234

4.2
CIM
Common Information Model, as defined by DMTF

4.3
CIM-RS
CIM Operations Over RESTful Services; the name of the protocol defined in this document

4.4
CQL
CIM Query Language, as defined in DSP0202

4.5
ECMAScript
a scripting language that is the standard version of what was called JavaScript. It is defined in ECMA-262.

4.6
HTTP
Hyper Text Transfer Protocol. HTTP version 1.1 is defined in RFC2616. Unless otherwise noted, the term HTTP is used in this document to mean both HTTP and HTTPS.

4.7
HTTPS
Hyper Text Transfer Protocol Secure, as defined in RFC2818

4.8
IANA
Internet Assigned Numbers Authority; see http://www.iana.org

4.9
JSON
JavaScript Object Notation, as defined in ECMA-262

4.10
MIME
Multipurpose Internet Mail Extensions, as defined in *IANA MIME Media Types*

4.11
REST
Representational State Transfer, as originally and informally described in *Architectural Styles and the Design of Network-based Software Architectures*

4.12
UCS
Universal Character Set, as defined in ISO/IEC 10646:2003

4.13
URI
Uniform Resource Identifier, as defined in RFC3986

4.14
UTF-8
UCS Transformation Format 8, as defined in ISO/IEC 10646:2003

4.15
WBEM
Web Based Enterprise Management, as defined by DMTF

4.16
XML
eXtensible Markup Language, as defined by W3C

5 Concepts

This clause defines concepts that are the basis for the definition of the CIM-RS protocol.

This document relies on the generic operations defined in DSP0223 for a definition on the semantics of the operations within the CIM-RS protocol. Thus, this document defines a mapping of generic operations to HTTP messages, and of the input and output parameters of generic operations to resource identifiers, HTTP header fields, and HTTP entity bodies.

5.1 REST architectural style supported by CIM-RS

CIM-RS follows most of the principles and constraints of the *REST architectural style* described by Roy Fielding in chapter 5 of *Architectural Styles and the Design of Network-based Software Architectures* and in *REST APIs must be hypertext driven*. CIM-RS deviates from some of these principles and constraints, as described in this clause.

The constraints defined in the REST architectural style are satisfied by CIM-RS as follows:

- **Client-Server:** The participants in CIM-RS have a client-server relationship between WBEM client and WBEM server, and between WBEM server and WBEM listener. This constraint is fully satisfied.
- **Stateless:** Interactions in CIM-RS are self-describing in that the server (that is, the WBEM server in its server role, and the WBEM listener) does not maintain any session state. This constraint is fully satisfied.

  NOTE: Pulled enumeration operations as defined in DSP0223 maintain the enumeration state either on the server side or on the client side. In both approaches, the client needs to hand back and forth an opaque data item called enumeration context, which is the actual enumeration state in case of a client-maintained enumeration state, or a handle to the enumeration state in case of a server-maintained enumeration state. CIM-RS supports both of these approaches. It is possible for a server to remain stateless as far as the enumeration state goes, by implementing the client-based approach. The approach implemented by a server is not visible to a client, because the enumeration context handed back and forth is opaque to the client in both approaches.
- **Cache:** The HTTP methods used by CIM-RS are used as defined in RFC2616 and RFC5789 (for PATCH). As a result, they are cacheable as defined in RFC2616. This constraint is fully satisfied.

  NOTE: RFC2616 defines only the result of HTTP GET methods to be cacheable.
- **Uniform interface:** The resources represented in CIM-RS are CIM namespaces, CIM classes, CIM qualifier types, and CIM instances. CIM-RS defines a uniform interface for creating, deleting, retrieving, replacing, and modifying these resources, based on HTTP methods. The resource identifiers used in that interface are uniformly structured. This constraint is satisfied, with the following deviation:

  CIM methods can be invoked in CIM-RS through the use of HTTP POST. This may be seen as a deviation from the REST architectural style, which suggests that any "method" be represented as a modification of a resource. However, that is not practical in CIM, because significant effort has been put into the definition of CIM method semantics in the CIM Schema and in management profiles, and into existing implementations of these methods.
- **Layered system:** Layering is an inherent part of CIM, because it defines a CIM model of managed objects in a managed environment and thus restricts knowledge of a client to only the modeled representation of the managed environment. CIM-RS fully supports layering in that it represents only the entities modeled in CIM. In addition, CIM-RS does not prevent the use of HTTP intermediaries (for example, caches and proxy servers). This constraint is fully satisfied.
- **Code-On-Demand:** CIM-RS does not provide for sending any code to the client. This constraint is not satisfied.

The representation of REST resources in CIM-RS is as follows:

- For CIM instances, its instance specification, including property values
- For CIM classes, its class specification, including any specified CIM qualifiers
- For CIM qualifier types, its qualifier type specification
- For CIM namespaces, the CIM instances of the CIM_Namespace class representing the namespaces

This approach makes the nature and state of these REST resources available to a client. Because these resource representations are defined at the level of the CIM model, the raw structure of any managed objects represented in the CIM model is not directly visible at the interface. This pattern allows describing REST resources independent of their specific managed environment platform or implementation.

The REST architectural style allows for the representation of static entities such as disk drives, or entities with varying property values over time, such as a metric measuring the amount of available disk space at a specific point in time, or entities that dynamically come into existence or cease to exist, such as a file system mount. CIM-RS represents CIM modeled entities as REST resources, and thus can represent all these types of entities.

In CIM-RS, resources are addressed through resource identifiers that are URIs. More precisely, resource identifiers are used to address the CIM entities represented as REST resources. The REST architectural style recommends that all addressing information for a resource is in the resource identifier (and not, for example, in the HTTP header). In addition, it recommends that resource identifiers are opaque to clients and clients should not be required to understand the structure of resource identifiers or be required to assemble any resource identifiers. CIM-RS follows the recommendations that all addressing information for a resource is in the resource identifier and on opaqueness and non-assembly of the resource identifier.

The REST architectural style promotes late binding between the abstracted resource that is addressed through a resource identifier and the resource representation that is chosen in the interaction between client and server. CIM-RS follows this by supporting multiple types of resource representations that are chosen through HTTP content negotiation. (For details, see 8.2.1.)

CIM-RS supports retrieval of parts of CIM classes or instances. These parts are selected through query parameters in the resource identifier URI. That renders these parts to be separate REST resources (that is, separate from the REST resource representing the entire CIM class or instance), following the principles in the REST architectural style.

The only top-level resources a client needs to know in CIM-RS are the resource identifier URIs of any CIM namespaces the client wants to interact with. From that point on, CIM-RS operations allow retrieval of the resource identifiers of any further REST resources representing CIM classes, instances, and qualifier types, by means of links returned along with previously returned resources.

6 Conformance

This clause defines the criteria for WBEM clients, WBEM servers, and WBEM listeners to implement the CIM-RS protocol conformant to this document.

WBEM clients, WBEM servers, and WBEM listeners implement the CIM-RS protocol conformant to this document only if they satisfy all provisions set out in this document.

The terms WBEM client, WBEM server, and WBEM listener in this document refer to WBEM clients, WBEM servers, and WBEM listeners that are conformant to this document, without explicitly mentioning that.

7 REST resources and their resource identifiers

This clause defines the REST resources used in the CIM-RS protocol and their resource identifiers.

These resource identifiers are called *WBEM resource identifiers* (or short: "resource identifier") in this document and are URIs that address REST resources accessible through a WBEM server. Some of these REST resources represent CIM objects accessible through the WBEM server, and some others are used to give additional context to HTTP methods such as GET.

The following example shows a resource identifier that addresses the CIM_ManagedElement class in the "root/cimv2" namespace accessible through the WBEM server at port 5988 of host "acme.com", using the "http" scheme:

```
http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_ManagedElement
```

7.1 Structure of WBEM resource identifiers

This clause defines the structure of WBEM resource identifiers.

7.1.1 General structure

RFC3986 defines the `URI-reference` ABNF rule as follows:

```
URI-reference = URI / relative-ref

URI           = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

relative-ref  = relative-part [ "?" query ] [ "#" fragment ]

hier-part     = "//" authority path-abempty
              / path-absolute
              / path-rootless
              / path-empty

relative-part = "//" authority path-abempty
              / path-absolute
              / path-noscheme
              / path-empty
```

WBEM resource identifiers shall conform to the following ABNF rule (which conforms to but restricts the `URI-reference` ABNF rule defined in RFC3986):

```
WBEM-resource-identifier = [ scheme ":" ] [ "//" authority ]
                           path-absolute
                           [ "?" query ]
```

Where:

- `scheme` is defined in RFC3986 and shall in addition conform to the definitions in 7.1.3.
- `authority` is defined in RFC3986 and shall in addition conform to the definitions in 7.1.4.
- `path-absolute` is defined in RFC3986 and shall in addition conform to the definitions in 7.1.5.
- `query` is defined in RFC3986 and shall in addition conform to the definitions in 7.3.

7.1.2 Percent-encoding

This clause defines how the percent-encoding rules defined in RFC3986 are applied to WBEM resource identifiers.

RFC3986 defines percent-encoding in its section 2.1 and defines the set of characters to be percent-encoded in URIs as follows:

- Any characters outside the character set allowed for URIs (that is, reserved + unreserved) shall be percent-encoded.
- Any characters in the reserved character set for URIs shall be percent-encoded if data for a URI component would conflict with a reserved character's purpose as a delimiter.
- Any other characters allowed for URIs should not be percent-encoded.

Further, RFC3986 defines the reserved and unreserved character sets for URIs as follows:

```
reserved = gen-delims / sub-delims

gen-delims = ":" / "/"             ; colon (U+003A), slash (U+002F)
           / "?" / "#"             ; quest.mark (U+003F), hash (U+0023)
           / "[" / "]"             ; l.bracket (U+005B), r.bracket(U+005D)
           / "@"                   ; at (U+0040)

sub-delims = "!" / "$"             ; excl.mark (U+0021), dollar (U+0024)
           / "&" / "'"             ; ampersand (U+0026), s.quote (U+0027)
           / "(" / ")"             ; l.paren. (U+0028), r.paren. (U+0029)
           / "*" / "+"             ; asterisk (U+002A), plus (U+002B)
           / "," / ";"             ; comma (U+002C), semicolon (U+003B)
           / "="                   ; equal (U+003D)

unreserved = ALPHA / DIGIT
           / "-" / "."             ; hyphen (U+002D), period (U+002E)
           / "_" / "~"             ; underscore (U+005F), tilde (U+007E)

ALPHA = %x41-5A / %x61-7A          ; A-Z, a-z

DIGIT = %x30-39                    ; 0-9
```

7.1.2.1 What to percent-encode

This clause defines which characters in a WBEM resource identifier can be percent-encoded, consistent with the percent-encoding rules defined in RFC3986.

For CIM element names and CIM namespace names used in a WBEM resource identifier, the following percent-encoding rules apply for producers of WBEM resource identifiers:

```
npe-chars-for-names = ALPHA / DIGIT / "_"
```

- Characters that match the ABNF rule `npe-chars-for-names` should not be percent-encoded.
- Characters that do not match the ABNF rule `npe-chars-for-names` shall be percent-encoded.

For CIM key values used in a WBEM resource identifier, the relevance of the reserved characters with respect to being interpreted as delimiters needs to be defined before the percent-encoding rules defined in RFC3986 can be applied. In order to prepare for potential future extensions, this document defines that all characters in the reserved character set for URIs are considered to be (actual or potential future) delimiters.

NOTE The subset of the reserved characters that is currently used as a delimiter of some sort in WBEM resource identifiers is: `":"`, `"/"`, `"?"`, `"&"`, `","`, `";"`, `"="`.

Thus, for CIM key values used in a WBEM resource identifier, the following percent-encoding rules apply for producers of WBEM resource identifiers:

```
npe-chars-for-keyvalues = ALPHA / DIGIT / "_" / "-" / "." / "~"
```

- Characters that match the ABNF rule `npe-chars-for-keyvalues` should not be percent-encoded.
- Characters that do not match the ABNF rule `npe-chars-for-keyvalues` shall be percent-encoded.

Consumers of WBEM resource identifiers shall support any percent-encoding within the WBEM resource identifier that is permissible according to the rules in this clause.

7.1.2.2 How to percent-encode

RFC3986 defines percent-encoding on the basis of data octets, but it does not define how characters are encoded as data octets. Because CIM element names, CIM namespace names, and CIM key values may contain UCS characters outside of the US-ASCII character set, this document defines the percent-encoding to be used in WBEM resource identifiers as follows.

Any UCS character that is being percent-encoded in WBEM resource identifiers shall be processed by first normalizing the UCS character using Normalization Form C (defined in The Unicode Standard, Annex #15), then encoding it to data octets using UTF-8, and finally percent-encoding the resulting data octets as defined in section 2.1 of RFC3986. The requirement to use a specific Unicode normalization form and a specific Unicode encoding (that is, UTF-8) ensures that the resulting string can be compared octet-wise without having to apply UCS character semantics.

For the purpose of representing key values in WBEM resource identifiers, values of CIM data types other than string shall be represented as string values as defined in DSP0004.

7.1.2.3 Examples

The following examples use the minimally needed percent-encodings:

- The CIM namespace name `"root/cimv2"` becomes `"root%2Fcimv2"` in a resource identifier.
- The CIM class name `"CIM_LogicalDevice"` remains unchanged in a resource identifier.
- The (German) CIM method name `"ÄnderungsRate"` becomes `"%C3%84%0AnderungsRate"` in a resource identifier. (C3 84 0A are the data octets of the UTF-8 encoding of the UCS character U+00C4, which represents `"Ä"` in normalized form.)
- The string typed key value `"a \"brown\" bag\n"` (represented using backslash escape sequences as defined for string literals in CIM MOF) becomes `"a%20%22brown%22%20bag%10"` in a resource identifier.
- The sint8 typed key value -42 becomes `"-42"` in a resource identifier.

7.1.3 Scheme

WBEM clients, WBEM servers, and WBEM listeners shall adhere to the following additional rules regarding the value of the `scheme` component in the `WBEM-resource-identifier` ABNF rule defined in 7.1.1:

- A value of `"http"` shall indicate the use of HTTP as defined in clause 8.21.
- A value of `"https"` shall indicate the use of HTTPS as defined in clause 8.21.
- No other values shall be used.
- The `[ scheme ":" ]` part of `WBEM-resource-identifier` may be omitted only if the scheme is known by other means.

7.1.4 Authority

WBEM clients, WBEM servers, and WBEM listeners shall adhere to the following additional rules regarding the value of the `authority` component in the `WBEM-resource-identifier` ABNF rule defined in 7.1.1:

- The `userinfo` component within `authority` shall not be specified because of security issues with specifying an unencrypted password.
- The `host` component within `authority` shall be the IP (V4 or V6) address of the WBEM server, or a DNS-resolvable host name for that IP address.
- If the `port` component within `authority` is not specified, the port shall be assumed as follows:
  - port number 80 for the scheme `"http"`
  - port number 443 for the scheme `"https"`
- The `[ "//" authority ]` part of `WBEM-resource-identifier` may be omitted only if the authority is known by other means.

7.1.5 Client knowledge about URI structure

This document describes the structure of WBEM resource identifiers as a requirement for the implementation of the server (WBEM server in its server role, or WBEM listener).

A WBEM client should not parse or construct WBEM resource identifiers except for constructing the WBEM resource identifiers of the top-level REST resources (that is, CIM namespaces).

A WBEM server in its client role (that is, when interacting with a WBEM listener) gets the WBEM resource identifiers for interacting with the WBEM listener from listener destinations it knows about. As a result, it does not need to construct any WBEM resource identifiers.

7.2 REST resources and their URIs

This clause defines the REST resources used in the CIM-RS protocol and the URIs addressing these resources by means of path components of WBEM resource identifiers. For details on how these path components are used to construct the URIs, see 7.1.

This clause defines the `path-absolute` rule by using ABNF incremental alternatives (that is, the `=/` construct), based on the initially empty rule:

```
path-absolute = ""     ; initially empty
```

To avoid ambiguities within a WBEM server that provides multiple access facilities through HTTP, all path components defined in this clause begin with "`/cimrs/`".

The path components defined in this clause not only provide for distinguishing the four types of CIM objects defined in DSP0004 (that is, namespace, class, instance, and qualifier type), but in addition encode some portion of the semantics of the operations defined in DSP0223. As a result, a resource identifier using these path components has sufficient operational semantics to be used with the HTTP methods, as detailed in clause 8.

7.2.1 Namespace collection resource

A *namespace collection resource* represents a set of CIM namespace objects in a WBEM server.

The WBEM resource identifier of a namespace collection resource has a defined structure. This allows WBEM clients to use this resource as a starting point (or top-level resource) for interacting with a WBEM server, and to discover further resources from there.

A WBEM resource identifier addressing a namespace collection resource shall conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = NamespaceCollectionPath

NamespaceCollectionPath = "/cimrs/namespaces"
```

Examples:

- `http://acme.com:5988/cimrs/namespaces`

  This resource identifier addresses the collection of all CIM namespaces accessible through the WBEM server at port 5988 of host "acme.com", using the "http" scheme.

7.2.2 Namespace resource

A *namespace resource* represents a CIM namespace object in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a namespace resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of namespace resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.3 Class collection resource

A *class collection resource* represents a set of CIM class objects in a particular CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a class collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of class collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.4 Class resource

A *class resource* represents a CIM class object in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a class resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of class resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.5 Class associator collection resource

A *class associator collection resource* represents a set of CIM class objects that are associated with a particular CIM class object in a CIM namespace in a WBEM server. The associated class objects may not all be in the same CIM namespace or WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a class associator collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of class associator collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.6 Class reference collection resource

A *class reference collection resource* represents the set of CIM association class objects that reference a particular CIM class object in a CIM namespace in a WBEM server. The referencing class objects may not all be in the same CIM namespace or WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a class reference collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of class reference collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.7 Class method invocation resource

A *class method invocation resource* represents an invocation point for static CIM methods that can be invoked on a CIM class, including the CIM class object in a CIM namespace in a WBEM server, on which the methods can be invoked.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a class method resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of class method resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.8 Instance collection resource

An *instance collection resource* represents a set of CIM instance objects of a particular CIM class in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of an instance collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of instance collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.9 Instance resource

An *instance resource* represents a CIM instance object in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of an instance resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of instance resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.10 Instance associator collection resource

An *instance associator collection resource* represents a set of CIM instance objects that are associated with a particular CIM instance object in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of an instance associator collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of instance associator collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.11 Instance reference collection resource

An *instance reference collection resource* represents the set of CIM association instance objects that reference a particular CIM instance object in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of an instance reference collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of instance reference collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.12 Instance method invocation resource

An *instance method invocation resource* represents an invocation point for (static or non-static) CIM methods that can be invoked on a CIM instance, including the CIM instance object in a CIM namespace in a WBEM server on which the methods can be invoked.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of an instance method resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of instance method resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.13 Qualifier type collection resource

A *qualifier type collection resource* represents the set of all CIM qualifier type objects in a particular CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a qualifier type collection resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of qualifier type collection resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.14 Qualifier type resource

A *qualifier type resource* represents a CIM qualifier type object in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of a qualifier type resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of qualifier type resources can be discovered by WBEM clients through links (see 8.2.2).

7.2.15 Instance query resource

An *instance query resource* represents a target for executing queries on CIM instances in a CIM namespace in a WBEM server.

Beyond the requirements defined in 7.1, the structure of the WBEM resource identifier of an instance query resource is implementation-defined (see ANNEX B for examples). The WBEM resource identifiers of instance query resources can be discovered by WBEM clients through links (see 8.2.2).

7.3 Query parameters in URIs

This clause defines the query component of WBEM resource identifiers, and applies in addition to the definition in RFC3986, section 3.4.

The representation of the query component is defined by the following ABNF rules:

```
query = query-parameter *( "&" query-parameter )
```

Where:

- `query-parameter` is a query parameter defined in the subclauses of 7.37.3.

Example:

- `/cimrs/namespaces/myns/classes?spc=CIM_System`

  This resource identifier addresses the list of classes in namespace “myns” whose superclass is CIM_System. (For details on the *spc* query parameter, see 7.3.17.)

Query parameters of URIs are case sensitive, as defined in RFC3986, section 6.2.2.1. As a result, two URIs that differ only in the lexical case of query parameters do not match. The query parameters defined in the subclauses of 7.3 define in some cases that the values of query parameters are to be treated case insensitively. In such cases, two URIs that differ only in the lexical case of query parameters address the same resource, even though the URIs do not match according to the rules defined in in RFC3986. It is recommended that producers of URIs use the lexical case for such element names as specified in the underlying CIM schema, in order to optimize caching based on URIs. For example, if a class is named "CIM_LogicalDevice", its use in the scn query parameter would be "scn=CIM_LogicalDevice".

Query parameters (identified by their name) shall not be repeated in WBEM resource identifiers. If query parameters are repeated in a WBEM resource identifier that is used as the target of an operation, the CIM-RS consumer shall reject that operation.

NOTE: RFC3986 does not detail how the `query` ABNF rule is broken into query parameters, and thus does not address the topic of query parameter repetition.

This clause defines the `query-parameter` rule by using ABNF incremental alternatives (that is, the `=/` construct), based on the initially empty rule:

```
query-parameter = ""     ; initially empty
```

7.3.1 acn (associated class name)

The *acn* query parameter acts as a restricting filter on the set of instances or classes returned by association traversal operations, as defined in the description of the *AssociatedClassName* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ AssociatedClassName-queryparm

AssociatedClassName-queryparm = [ "acn=" AssociatedClassName ]
```

Where:

- `AssociatedClassName` is the name of the associated class on the far end of the association as seen from the source class or instance (including its schema prefix).

An acn query parameter that is not specified in the resource identifier shall correspond to an AssociatedClassName generic operation parameter value of Null.

An acn query parameter that is specified in the resource identifier with a value shall correspond to the equivalent AssociatedClassName generic operation parameter value.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), class names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

`""` (not specified) – Null

`"acn=CIM_Endpoint1"` – associated class name of "CIM_Endpoint1"

7.3.2 arn (associated role name)

The *arn* query parameter acts as a restricting filter on the set of instances or classes returned by association traversal operations, as defined in the description of the *AssociatedRoleName* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ AssociatedRoleName-queryparm

AssociatedRoleName-queryparm = [ "arn=" AssociatedRoleName ]
```

Where:

- `AssociatedRoleName` is the name of the role (that is, reference) of the associated class or instance.

An arn query parameter that is not specified in the resource identifier shall correspond to an AssociatedRoleName generic operation parameter value of Null.

An arn query parameter that is specified in the resource identifier with a value shall correspond to the equivalent AssociatedRoleName generic operation parameter value.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), role (that is, reference) names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

`""` (not specified) – Null

`"arn=Child"` – associated role (that is, reference) name "Child"

7.3.3 c (class)

The *c* query parameter acts as a restricting filter on the set of classes returned by operations to the one that has the class name specified in the c query parameter. It has no directly corresponding generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ Class-queryparm
```

```
Class-queryparm = [ "c=" ClassName ]
```

Where:

- `ClassName` is the name of the class (including its schema prefix).

A c query parameter that is not specified in the resource identifier shall not restrict the returned classes.

A c query parameter that is specified in the resource identifier shall restrict the returned classes to the one that has the class name specified in the c query parameter.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), class names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

`""` (not specified) – no restriction

`"c=CIM_LogicalElement"` – restrict returned classes to "CIM_LogicalElement"

7.3.4 dd (delete dependents)

The *dd* query parameter controls whether dependent classes shall be deleted, as defined in the description of the *DeleteDependents* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ DeleteDependents-queryparm

DeleteDependents-queryparm = [ "dd" [ "=" ( "true" / "false" ) ] ]
```

A dd query parameter that is not specified in the resource identifier shall correspond to a DeleteDependents generic operation parameter value of False.

A dd query parameter that is specified in the resource identifier without a value shall correspond to a DeleteDependents generic operation parameter value of True.

A dd query parameter that is specified in the resource identifier with a value shall correspond to the equivalent DeleteDependents generic operation parameter value.

The value of the dd query parameter, if specified, shall be either "true" or "false".

Examples:

`""` (not specified) – False

`"dd"` – True

`"dd=false"` – False

`"dd=true"` – True

7.3.5 esbp (exclude subclass properties)

The *esbp* query parameter acts as a restricting filter on the set of properties that is included in any returned instances or classes, as defined in the description of the *ExcludeSubclassProperties* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ ExcludeSubclassProperties-queryparm

ExcludeSubclassProperties-queryparm = [ "esbp" [ "=" ("true"/"false") ] ]
```

An esbp query parameter that is not specified in the resource identifier shall correspond to an ExcludeSubclassProperties generic operation parameter value of False (causing subclass properties not to be excluded).

An esbp query parameter that is specified in the resource identifier without a value shall correspond to an ExcludeSubclassProperties generic operation parameter value of True (causing subclass properties to be excluded).

An esbp query parameter that is specified in the resource identifier with a value shall correspond to the equivalent ExcludeSubclassProperties generic operation parameter value.

The value of the esbp query parameter, if specified, shall be either "true" or "false".

Examples:

"" (not specified) – False

`"esbp"` – True

`"esbp=false"` – False

`"esbp=true"` – True

7.3.6 fql (filter query language)

The *fql* query parameter specifies the query language used for the value of the fqs (filter query string) query parameter, as defined in the description of the *FilterQueryLanguage* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ FilterQueryLanguage-queryparm

FilterQueryLanguage-queryparm = [ "fql=" FilterQueryLanguage ];
```

An fql query parameter that is not specified in the resource identifier shall correspond to a FilterQueryLanguage generic operation parameter value of NULL.

An fql query parameter that is specified in the resource identifier with a value shall correspond to the equivalent FilterQueryLanguage generic operation parameter value.

Examples:

"" (not specified) – NULL

`"fql=DMTF%3ACQL"` – query language "DMTF:CQL"

7.3.7 fqs (filter query string)

The *fqs* query parameter specifies the query string of a filter query for operations that return instance collections, in the query language indicated by the fql query parameter.

The filter query acts as an additional restricting filter on the set of instances returned, as defined in the description of the *FilterQueryString* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ FilterQueryString-queryparm

FilterQueryString-queryparm = [ "fqs=" QueryString ]
```

An fqs query parameter that is not specified in the resource identifier shall correspond to a FilterQueryString generic operation parameter value of NULL.

An fqs query parameter that is specified in the resource identifier with a value shall correspond to the equivalent FilterQueryString generic operation parameter value. DSP0223 defines restrictions on the filter query.

Examples:

"" (not specified) – NULL

`"fqs=SELECT%20%2A%20FROM%20CIM_StorageExtent%20WHERE%20Name%3D'abc'"` – CQL query "SELECT * FROM CIM_StorageExtent WHERE Name='abc'"

7.3.8 ico (include class origin)

The *ico* query parameter controls whether class origin information is returned for any elements in returned instances or classes, as defined in the description of the *IncludeClassOrigin* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ IncludeClassOrigin-queryparm

IncludeClassOrigin-queryparm = [ "ico" [ "=" ( "true" / "false" ) ] ]
```

An ico query parameter that is not specified in the resource identifier shall correspond to an IncludeClassOrigin generic operation parameter value of False.

An ico query parameter that is specified in the resource identifier without a value shall correspond to an IncludeClassOrigin generic operation parameter value of True.

An ico query parameter that is specified in the resource identifier with a value shall correspond to the equivalent IncludeClassOrigin generic operation parameter value.

The value of the ico query parameter, if specified, shall be either "true" or "false".

Examples:

"" (not specified) – False

`"ico"` – True

`"ico=false"` – False

`"ico=true"` – True

7.3.9 iie (include inherited elements)

The *iie* query parameter controls whether elements inherited from superclasses are included in any returned classes, as defined in the description of the *IncludeInheritedElements* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ IncludeInheritedElements-queryparm

IncludeInheritedElements-queryparm = [ "iie" [ "=" ("true"/"false") ] ]
```

An iie query parameter that is not specified in the resource identifier shall correspond to an IncludeInheritedElements generic operation parameter value of False (causing inherited elements not to be included).

An iie query parameter that is specified in the resource identifier without a value shall correspond to an IncludeInheritedElements generic operation parameter value of True (causing inherited elements to be included).

An iie query parameter that is specified in the resource identifier with a value shall correspond to the equivalent IncludeInheritedElements generic operation parameter value.

The value of the iie query parameter, if specified, shall be either "true" or "false".

Examples:

`""` (not specified) – False

`"iie"` – True

`"iie=false"` – False

`"iie=true"` – True

7.3.10 ip (included properties)

The *ip* query parameter acts as a restricting filter on the set of properties that is included in any returned instances or classes, as defined in the description of the *IncludedProperties* generic operation parameter in DSP0223.

Its representation in the query component is defined by the following ABNF:

```
query-parameter =/ IncludedProperties-queryparm

IncludedProperties-queryparm = [ "ip=" PropertyNameList ]
```

Where:

- `PropertyNameList` is a comma-separated list of property names. The list may be empty.

An ip query parameter that is not specified in the resource identifier shall correspond to an IncludedProperties generic operation parameter value of Null (causing no restriction on properties).

An ip query parameter that is specified in the resource identifier without a value shall correspond to an IncludedProperties generic operation parameter value of the empty array (causing all properties to be excluded).

An ip query parameter that is specified in the resource identifier with a value shall correspond to the equivalent IncludedProperties generic operation parameter value, where each property name in the list in the query parameter value shall correspond to an array entry in the generic operation parameter value (causing properties not specified to be excluded).

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), property names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

`""` (not specified) – Null

`"ip="` – empty list

`"ip=Prop1,Prop2"` – list of Prop1, Prop2

7.3.11 iq (include qualifiers)

The *iq* query parameter controls whether qualifier information is returned for any elements in returned instances or classes, as defined in the description of the *IncludeQualifiers* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ IncludeQualifiers-queryparm

IncludeQualifiers-queryparm = [ "iq" [ "=" ( "true" / "false" ) ] ]
```

An iq query parameter that is not specified in the resource identifier shall correspond to an IncludeQualifiers generic operation parameter value of False.

An iq query parameter that is specified in the resource identifier without a value shall correspond to an IncludeQualifiers generic operation parameter value of True.

An iq query parameter that is specified in the resource identifier with a value shall correspond to the equivalent IncludeQualifiers generic operation parameter value.

The value of the iq query parameter, if specified, shall be either "true" or "false".

Examples:

`""` (not specified) – False

`"iq"` – True

`"iq=false"` – False

`"iq=true"` – True

7.3.12 isbc (include subclasses)

The *isbc* query parameter controls whether the entire tree of subclasses is included in any returned classes, as defined in the description of the *IncludeSubclasses* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ IncludeSubclasses-queryparm

```

```
IncludeSubclasses-queryparm = [ "isbc" [ "=" ( "true" / "false" ) ] ]
```

An isbc query parameter that is not specified in the resource identifier shall correspond to an IncludeSubclasses generic operation parameter value of False.

An isbc query parameter that is specified in the resource identifier without a value shall correspond to an IncludeSubclasses generic operation parameter value of True.

An isbc query parameter that is specified in the resource identifier with a value shall correspond to the equivalent IncludeSubclasses generic operation parameter value.

The value of the isbc query parameter, if specified, shall be either "true" or "false".

Examples:

"" (not specified) – False

"isbc" – True

"isbc=false" – False

"isbc=true" – True

7.3.13 n (namespace)

The *n* query parameter acts as a restricting filter on the set of namespaces returned by operations to the one that has the namespace name specified in the n query parameter. It has no directly corresponding generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ Namespace-queryparm

Namespace-queryparm = [ "n=" NamespaceName ]
```

Where:

- NamespaceName is the name of the namespace (e.g. "root%2Fcimv2").

An n query parameter that is not specified in the resource identifier shall not restrict the returned namespaces.

An n query parameter that is specified in the resource identifier shall restrict the returned namespaces to the one that has the namespace name specified in the n query parameter.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), namespace names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

"" (not specified) – no restriction

"n=root%2F/cimv2" – restrict returned namespaces to "root/cimv2"

7.3.14 pr (paged retrieval)

The *pr* query parameter controls whether paged retrieval of resources is requested (see 8.3.2).

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ PagedRetrieval-queryparm

PagedRetrieval-queryparm = [ "pr" [ "=" enumerationcontext ] ]
```

A pr query parameter that is not specified in the resource identifier means that no paged retrieval is requested. In other words, the entire resource is requested to be retrieved.

A pr query parameter that is specified in the resource identifier without a value means that paged retrieval of the first subset of the resource is requested. In other words, this opens an enumeration.

A pr query parameter that is specified in the resource identifier with a value means that paged retrieval is requested of the subset of the resource that is identified by the value. The query parameter value shall be the base64url-encoded enumeration context value without any "=" padding characters at the end. The base64url encoding is defined in RFC4648.

Examples:

`""` (not specified) – No paged retrieval is requested

`"pr"` – Paged retrieval of the first subset of the resource is requested

`"pr=YWJjZGVm"` – Paged retrieval of the resource subset identified by enumeration context value "abcdef" is requested

7.3.15 q (qualifier)

The *q* query parameter acts as a restricting filter on the set of qualifier types returned by operations to the one that has the qualifier name specified in the q query parameter. It has no directly corresponding generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ Qualifier-queryparm

Qualifier-queryparm = [ "q=" QualifierName ]
```

Where:

- `QualifierName` is the name of the qualifier type (e.g. "Alias").

A q query parameter that is not specified in the resource identifier shall not restrict the returned qualifier types.

A q query parameter that is specified in the resource identifier shall restrict the returned qualifier types to the one that has the qualifier name specified in the n query parameter.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), qualifier names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

`""` (not specified) – no restriction

`"q=Alias"` – restrict returned qualifier types to "Alias"

7.3.16 rcn (referencing class name)

The *rcn* query parameter acts as a restricting filter on the set of instances or classes returned by association traversal operations, as defined in the description of the *AssociationClassName* generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ ReferencingClassName-queryparm

ReferencingClassName-queryparm = [ "rcn=" ReferencingClassName ]
```

Where:

- `ReferencingClassName` is the name of the association class referencing the source instance or class (including its schema prefix).

An rcn query parameter that is not specified in the resource identifier shall correspond to an AssociationClassName generic operation parameter value of Null.

An rcn query parameter that is specified in the resource identifier with a value shall correspond to the equivalent AssociationClassName generic operation parameter value.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), class names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

`""` (not specified) – Null

`"rcn=CIM_Assoc1"` – referencing class name of "CIM_Assoc1"

7.3.17 spc (superclass)

The *spc* query parameter acts as a restricting filter on the set of classes returned by operations to those that have the superclass specified in the spc query parameter. It has no directly corresponding generic operation parameter in DSP0223.

The representation of this query parameter is defined by the following ABNF:

```
query-parameter =/ Superclass-queryparm

Superclass-queryparm = [ "spc=" SuperClassName ]
```

Where:

- `SuperClassName` is the name of the superclass (including its schema prefix).

An spc query parameter that is not specified in the resource identifier shall not restrict the returned classes.

An spc query parameter that is specified in the resource identifier shall restrict the returned classes to those that have the specified superclass. In other words, the returned classes are restricted to subclasses of the specified class.

NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), class names are to be interpreted case insensitively, as defined in DSP0004.

Examples:

1259 "" (not specified) – no restriction

1260 "spc=CIM_LogicalElement" – restrict returned classes to subclasses of "CIM_LogicalElement"

1261 ### 7.3.18 srn (source role name)

1262 The *srn* query parameter acts as a restricting filter on the set of instances or classes returned by
1263 association traversal operations, as defined in the description of the *SourceRoleName* or *RoleName*
1264 generic operation parameter in DSP0223, depending on the actual operation.

1265 The representation of this query parameter is defined by the following ABNF:

```
1266 query-parameter =/ SourceRoleName-queryparm
1267
1268 SourceRoleName-queryparm = [ "srn=" SourceRoleName ]
```

1269 Where:

1270 - SourceRoleName is the name of the role (that is, reference) of the source class or instance.

1271 An srn query parameter that is not specified in the resource identifier shall correspond to a
1272 SourceRoleName or RoleName generic operation parameter value of Null.

1273 An srn query parameter that is specified in the resource identifier with a value shall correspond to the
1274 equivalent SourceRoleName or RoleName generic operation parameter value.

1275 NOTE: While query parameters of URIs are case sensitive, as defined in RFC3986 (section 6.2.2.1), role (that is,
1276 reference) names are to be interpreted case insensitively, as defined in DSP0004.

1277 Examples:

1278 "" (not specified) – Null

1279 "srn=Parent" – source role (that is, reference) name "Parent"

1280 8 Operations

1281 This clause defines the operations of the CIM-RS protocol.

1282 8.1 Overview on REST resources and HTTP methods

1283 The operations of the CIM-RS protocol are defined as HTTP methods on certain REST resources, as
1284 listed in Table 1.

1285 **Table 1 – REST resources and HTTP methods**

Target REST Resource	HTTP Methods
WBEM server	OPTIONS, see 8.5.1
Namespace collection resource, see 7.2.1	GET, see 8.6.1
Namespace resource, see 7.2.2	GET, see 8.7.1
Class collection resource, see 7.2.3	GET, see 8.8.1
	POST, see 8.8.2

Target REST Resource	HTTP Methods
Class resource, see 7.2.4	GET, see 8.9.1
	DELETE, see 8.9.2
	PUT, see 8.9.3
Class associator collection resource, see 7.2.5	GET, see 8.10.1
Class reference collection resource, see 7.2.6	GET, see 8.11.1
Class method invocation resource, see 7.2.7	POST, see 8.12.1
Instance collection resource, see 7.2.8	GET, see 8.13.1
	POST, see 8.13.2
Instance resource, see 7.2.9	GET, see 8.14.1
	DELETE, see 8.14.2
	PUT, see 8.14.3
	PATCH, see 8.14.4
Instance associator collection resource, see 7.2.10	GET, see 8.15.1
Instance reference collection resource, see 7.2.11	GET, see 8.16.1
Instance method invocation resource, see 7.2.12	POST, see 8.17.1
Qualifier type collection resource, see 7.2.13	GET, see 8.18.1
	POST, see 8.18.2
Qualifier type resource, see 7.2.14	GET, see 8.19.1
	DELETE, see 8.19.2
	PUT, see 8.19.3
Instance query resource, see 7.2.15	POST, see 8.20.1

8.2 Common behaviors for all operations

8.2.1 Content negotiation

WBEM clients, WBEM servers, and WBEM listeners shall support server-driven content negotiation as defined in RFC2616, based on the Accept request header field (defined in RFC2616 and in 9.4.1), and the Content-Type response header field (defined in RFC2616 and in 9.4.2).

Requirements for the media types used in these header fields are defined in 10.1.

The HTTP OPTIONS method can be used to retrieve the set of CIM-RS payload representations supported by a WBEM server or WBEM listener, as described in 8.5.1.

8.2.2 Links

Some payload elements in an HTTP entity body include WBEM resource identifiers to the resource represented by the payload element, or to related resources or resource collections. Such WBEM resource identifiers in payload elements are called *links*.

Links have a name. The name of a link is case sensitive. Link names, their semantic, and the requirements for including certain named links in the payload element are defined in this document.

Links have a target. The target of a link is a resource or resource collection that can be operated on using CIM-RS operations.

The descriptions of payload elements in 8.4 define requirements for the inclusion of links.

If the paged retrieval feature is implemented, additional links are included in some payload elements, as shown in Table 4. shows an overview of links included in payload elements, and the resources targeted by these links.

If the paged retrieval feature is implemented, additional links are included in some payload elements, as shown in Table 4.

Table 2 – Links included in payload elements

Payload element	Links included	Resource targeted by link
NamespaceCollection, see 8.4.1	"self"	the represented namespace collection itself
Namespace, see 8.4.2	"self"	the represented namespace itself
	"classes"	class collection of all classes in the represented namespace
	"qualifiers"	qualifier type collection of all qualifier types in the represented namespace
	"instancequery"	instance query resource of the represented namespace
ClassCollection, see 8.4.3	"self"	the represented class collection itself
	"namespace"	namespace of the represented class collection
Class, see 8.4.4	"self"	the represented class itself
	"namespace"	namespace of the represented class
	"instances"	instance collection of all instances of the represented class
	"methodinvocation"	class method invocation resource for all methods that can be invoked on the represented class
	"associators"	class collection of all classes associated to the represented class
	"references"	class collection of all classes referencing the represented class
InstanceCollection, see 8.4.5	"self"	the represented instance collection itself
	"class"	common superclass of the creation classes of the instances in the represented instance collection
Instance, see 8.4.6	"self"	the represented instance itself
	"class"	creation class of the represented instance

Payload element	Links included	Resource targeted by link
	"methodinvocation"	instance method invocation resource for all methods that can be invoked on the represented instance
	"associators"	instance collection of all instances associated to the represented instance
	"references"	instance collection of all instances referencing the represented instance
QualifierTypeCollection, see 8.4.7	"self"	the represented qualifier type collection itself
	"namespace"	namespace of the represented qualifier type collection
QualifierType, see 8.4.8	"self"	the represented qualifier type itself
	"namespace"	namespace of the represented qualifier type
MethodInvocationRequest, see 8.4.9	none	N/A
MethodInvocationResponse, see 8.4.10	none	N/A
InstanceModificationRequest, see 8.4.11	none	N/A
InstanceQueryRequest, see 8.4.12	none	N/A

1310

1311 **EXPERIMENTAL**

1312 ### 8.2.3 Verifying the basis of resource modifications

1313 The PUT instance operation (see 8.14.3) supplies a modified instance as input. The CIM-RS protocol
1314 provides for a means to verify for a WBEM server whether the current state of the resource is still the
1315 same as when the WBEM client retrieved the resource as a basis for providing the modified resource.

1316 This is achieved by using the value of the CIM Generation property (defined in CIM_ManagedElement) as
1317 an entity tag with the Etag and If-Match HTTP header fields, as described in 9.4.3 and 9.4.4.

1318 This ability is experimental, because the Generation property is defined as experimental in the current
1319 CIM Schema.

1320 This ability is part of the optional entity tagging feature (see 8.3.1).

1321 **EXPERIMENTAL**

1322 ### 8.2.4 Caching of responses

1323 Caching of responses from WBEM servers and WBEM listeners is described in RFC2616. This document
1324 does not define any additional constraints or restrictions on caching.

1325 Implementing the entity tagging feature (see 8.3.1) improves cache control.

8.2.5 Success and failure

Operations performed within the CIM-RS protocol either succeed or fail. There is no concept of "partial success".

If an operation succeeds, it returns its output data to the operation requester and does not include any error messages.

If it fails, it returns one or more error messages and no output data.

For example, if an instance enumeration operation were able to return some, but not all, instances successfully, then the operation fails without returning any instances.

8.2.6 Error messages

HTTP status codes are used to convey errors. The definition of HTTP status codes defined in RFC2616 is the basis for each operation, and the operation description may specify additional constraints on the use of HTTP status codes.

NOTE: The specification of extended error information (for example, using an ordered list of embedded CIM_Error instances) is not described in this version of the document.

8.2.7 Consistency model

The operations of the CIM-RS protocol shall conform to the consistency model defined in DSP0223.

8.2.8 Common operation parameters for all operations

This clause defines commonly used operation parameters for the operations. The description of the individual operations references these parameters as appropriate. Not every operation uses every one of these common parameters.

8.3 Optional behaviors for CIM-RS protocol

This clause defines optional behaviors for the implementation of the CIM-RS protocol.

8.3.1 Entity tagging feature

Implementation of the entity tagging feature in WBEM servers and WBEM clients provides for verifying the basis of resource modifications and thus for improved consistency control in instance modifications, as described in 8.2.3, and for improved cache control, as described in 8.2.4.

Implementation of the entity tagging feature is optional for WBEM clients and WBEM servers, independently.

Implementation of the entity tagging feature in a WBEM server is indicated through the CIMRS-Entity-Tagging-Feature header field (see 9.4.6).

8.3.2 Paged retrieval feature

Implementation of the paged retrieval feature in WBEM servers and WBEM clients supports *pulled enumerations* as defined in DSP0223, that is, the retrieval of a result set through multiple HTTP GET methods, each of which retrieves a subset of the result set.

The concept for paged retrieval is consistent with the concept for paged feeds defined in RFC5005. The consistency of the result set is defined in DSP0223 (again, consistent with RFC5005).

Implementation of the paged retrieval feature is optional for WBEM clients and WBEM servers, independently.

Implementation of the paged retrieval feature in a WBEM server is indicated through the CIMRS-Paged-Retrieval-Feature header field (see 9.4.7).

CIM-RS supports both modes of maintaining the enumeration state that are defined by DSP0223. The WBEM server decides about the mode that is used, and that decision is transparent to the WBEM client:

- If the WBEM server maintains the enumeration state, it returns to the WBEM client a reference to that state as the value of an enumeration context.
- If the WBEM server does not maintain the enumeration state (that is, it is stateless with respect to the enumeration state), it returns to the WBEM client the entire enumeration state by value, as the value of an enumeration context. The enumeration context value may change as the enumeration proceeds.

WBEM servers implementing paged retrieval should implement the mode where they do not maintain the enumeration state.

In both modes, the enumeration context for paged retrieval of a resource is represented in the WBEM resource identifier of the resource, through the *pr* query parameter (see 7.3.13).

The enumeration context value is passed to the WBEM server by means of the pr query parameter in the WBEM resource identifier referencing the resource to be retrieved. A "next" link in the result set of a paged retrieval is used to pass the new enumeration context value back to the WBEM client. In order to retrieve the next paged retrieval subset, the WBEM client uses the WBEM resource identifier specified in the "next" link as the target of the HTTP GET method.

In both modes, a WBEM server shall choose the value of the enumeration context such that it is different for each distinct paged retrieval subset within an enumeration. This allows a WBEM server to distinguish between a request to retrieve the next subset, and a request to repeat the retrieval of the last subset. A WBEM server shall not reject requests to retrieve a paged retrieval subset that had already been retrieved previously, as identified by the enumeration context value. This rule is motivated by the need to be able to repeat requests and by the need for the HTTP GET method to be idempotent.

This version of this document limits paged retrieval to CIM instances, consistent with the current version of DSP0223. The following CIM-RS operations support paged retrieval of their result:

Table 3 – Operations supporting paged retrieval

Operation	Retrieved resource	Description
GET instance collection	instance collection	see 8.13.1
GET instance associator collection	instance collection	see 8.15.1
GET instance reference collection	instance collection	see 8.16.1

If paged retrieval is implemented, it shall be supported for all operations listed in Table 3.

The actual use of paged retrieval is negotiated between WBEM client and WBEM server. A WBEM client initiates paged retrieval of a resource by specifying the pr query parameter without a value on a HTTP GET method. A WBEM server that does not support paged retrieval shall ignore the presence of the pr query parameter and shall proceed as if no paged retrieval had been requested. A WBEM server that

supports paged retrieval shall return the paged retrieval subset identified by the pr query parameter and shall include the links described in Table 4. The pr query parameter value is the enumeration context value in an encoded form, for details see 7.3.13.

Table 4 – Additional links for paged retrieval

Link name	Absolute / Relative	Requirement	Targeted resource
first	Absolute or relative to self	Conditional	First paged retrieval subset of result set. Condition: The WBEM server supports paged retrieval.
next	Absolute or relative to self	Conditional	Next paged retrieval subset of result set. Condition: The WBEM server supports paged retrieval and the enumeration is not yet exhausted.

NOTE: In a strict sense, the different WBEM resource identifiers used during paged retrieval of the resource each represents a different resource, so that each subset returned by a paged retrieval operation would by a distinct resource. However, that view makes it difficult to distinguish the entire resource from the subset resources retrieved. Therefore, this document does not take that view and instead distinguishes between the subsets retrieved and the (one) resource of which the subsets are retrieved.

NOTE: The use of HTTP range requests as defined in RFC2616 has been considered and dismissed, because the semantics of an ordered sequence of items that can be accessed by item number cannot be provided by implementations that support the opaque server-defined enumeration context values mandated by DSP0223.

8.4 Protocol payload elements

This clause defines the elements that are used in the payload of the messages of the operations defined in clause 8. This clause defines these payload elements in a normative but abstract way. Documents defining payload representation formats are expected to define how each of these payload elements is represented.

8.4.1 NamespaceCollection payload element

A *NamespaceCollection* payload element represents a set of Namespace payload elements (see 8.4.2). The members of the set depend on the context.

A NamespaceCollection payload element includes the following properties:

Table 5 – Properties of NamespaceCollection payload element

Property name	Generic type	Requirement	Description
namespaces	Namespace [ ]	Mandatory	set of Namespace payload elements (see 8.4.2)

A NamespaceCollection payload element includes the following links when used in response messages:

Table 6 – Links of NamespaceCollection payload element

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented namespace collection itself

8.4.2 Namespace payload element

A *Namespace* payload element represents a CIM namespace.

There is no directly matching generic type representing a namespace payload element.

A Namespace payload element includes the following properties:

Table 7 – Properties of Namespace payload element

Property name	Generic type	Requirement	Description
name	string (see DSP0223)	Mandatory	namespace name (for example, "root/cimv2")

A Namespace payload element includes the following links when used in response messages:

Table 8 – Links of Namespace payload element

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented namespace itself
classes	Absolute or relative to self	Mandatory	class collection of all classes in the represented namespace
qualifiers	Absolute or relative to self	Mandatory	qualifier type collection of all qualifier types in the represented namespace
instancequery	Absolute or relative to self	Mandatory	instance query resource of the represented namespace

8.4.3 ClassCollection payload element

A C*lassCollection* payload element represents a set of Class payload elements (see 8.4.4). The members of the set depend on the context.

A ClassCollection payload element includes the following properties:

Table 9 – Properties of ClassCollection payload element

Property name	Generic type	Requirement	Description
classes	Class [ ]	Mandatory	set of Class payload elements (see 8.4.4)

A ClassCollection payload element includes the following links when used in response messages:

Table 10 – Links of ClassCollection payload element

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented class collection itself
namespace	Absolute or relative to self	Mandatory	namespace of the represented class collection

1443

1444 The "namespace" link of the ClassCollection payload element limits the class collection to classes from
1445 the same namespace. In scenarios where association classes cross namespaces, this does not represent
1446 a limitation; because all associated classes need to exist in both namespaces anyway.
1447

1448 8.4.4 Class payload element

1449 A *Class* payload element represents a CIM class specification (that is, the definition of a class and its
1450 elements).

1451 A Class payload element includes the following properties:

1452 **Table 11 – Properties of Class payload payload element**

Property name	Generic type	Requirement	Description
class	Class-Specification (see DSP0223)	Mandatory	class specification

1453

1454 A Class payload element includes the following links when used in response messages:

1455 **Table 12 – Links of Class payload element**

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented class itself
namespace	Absolute or relative to self	Mandatory	namespace of the represented class
instances	Absolute or relative to self	Mandatory	instance collection of all instances of the represented class
method-invocation	Absolute or relative to self	Mandatory	class method invocation resource for all methods that can be invoked on the represented class
associators	Absolute or relative to self	Mandatory	class collection of all classes associated to the represented class
references	Absolute or relative to self	Mandatory	class collection of all classes referencing the represented class

1456
1457

1458 8.4.5 InstanceCollection payload element

1459 An *InstanceCollection* payload element represents a set of Instance payload elements (see 8.4.6). The
1460 members of the set depend on the context.

An InstanceCollection payload element includes the following properties:

Table 13 – Properties of InstanceCollection payload element

Property name	Generic type	Requirement	Description
instances	Instance [ ]	Mandatory	set of Instance payload elements (see 8.4.6)

An InstanceCollection payload element includes the following links when used in response messages:

Table 14 – Links of InstanceCollection payload element

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented instance collection itself
class	Absolute or relative to self	Conditional Exclusive	common superclass of the creation classes of the instances in the represented instance collection

Condition for the "class" link: The instance collection contains instances that have a common superclass. If the condition is met, the "class" link shall be included; otherwise, it shall not be included.

An example for an instance collection containing instances that may not have a common superclass is the result set of traversing a ternary association to both far ends.

8.4.6 Instance payload element

An *Instance* payload element represents a CIM instance specification (that is, an instance including its property values).

An Instance payload element includes the following properties:

Table 15 – Properties of Instance payload element

Property name	Generic type	Requirement	Description
class	creation class of Instance-Specification (see DSP0223)	Mandatory	name of creation class of the represented instance
properties	properties of Instance-Specification (see DSP0223)	Mandatory	properties of the represented instance, including their values

1477 An Instance payload element includes the following links when used in response messages:

1478 **Table 16 – Links of Instance payload element**

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented instance itself
class	Absolute or relative to self	Mandatory	creation class of the represented instance
method-invocation	Absolute or relative to self	Mandatory	instance method invocation resource for all methods that can be invoked on the represented instance
associators	Absolute or relative to self	Mandatory	instance collection of all instances associated to the represented instance
references	Absolute or relative to self	Mandatory	instance collection of all instances referencing the represented instance

1479
1480

1481 8.4.7 QualifierTypeCollection payload element

1482 A *QualifierTypeCollection* payload element represents a set of QualifierType payload elements (see
1483 8.4.8).

1484 A QualifierTypeCollection payload element includes the following properties:

1485 **Table 17 – Properties of QualifierTypeCollection payload element**

Property name	Generic type	Requirement	Description
qualifiertypes	QualifierType [ ]	Mandatory	Set of QualifierType payload elements (see 8.4.8)

1486 A QualifierTypeCollection payload element includes the following links when used in response messages:

1487 **Table 18 – Links of QualifierTypeCollection payload element**

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented qualifier type collection itself
namespace	Absolute or relative to self	Mandatory	namespace of the represented qualifier type collection

1488

1489 8.4.8 QualifierType payload element

1490 A Q*ualifierType* payload element represents a CIM qualifier type specification.

1491 A QualifierType payload element includes the following properties:

1492 **Table 19 – Properties of QualifierType payload element**

Property name	Generic type	Requirement	Description
type	string (see DSP0223)	Mandatory	Name of CIM datatype of the represented qualifier type specification
default	string (see DSP0223)	Mandatory	Default value of the represented qualifier type specification
scope	string [ ] (see DSP0223)	Mandatory	Set of scopes of the represented qualifier type specification
flavor	string [ ] (see DSP0223)	Mandatory	Set of flavors of the represented qualifier type specification

1493 A QualifierType payload element includes the following links when used in response messages:

1494 **Table 20 – Links of QualifierType payload element**

Link name	Absolute / Relative	Requirement	Targeted resource
self	Absolute	Mandatory	the represented qualifier type itself
namespace	Absolute or relative to self	Mandatory	namespace of the represented qualifier type

1495

1496 8.4.9 MethodInvocationRequest payload element

1497 A *MethodInvocationRequest* payload element represents the data used in a POST class method request
1498 (see 8.12.1) or POST instance method request (see 8.17.1).

1499 A MethodInvocationRequest payload element includes the following properties:

1500 **Table 21 – Properties of MethodInvocationRequest payload element**

Property name	Generic type	Requirement	Description
name	MethodName (see DSP0223)	Mandatory	method name
parameters	Parameter-Value [ ] (see DSP0223)	Mandatory	set of input parameters

1501

1502 8.4.10 MethodInvocationResponse payload element

1503 A *MethodInvocationResponse* payload element represents the data used in a POST class method
1504 response (see 8.12.1) or POST instance method response (see 8.17.1).

1505 A MethodInvocationResponse payload element includes the following properties:

1506 **Table 22 – Properties of MethodInvocationResponse payload element**

Property name	Generic type	Requirement	Description
name	MethodName (see DSP0223)	Mandatory	method name
returnvalue	ReturnValue (see DSP0223)	Mandatory	return value
parameters	Parameter-Value [ ] (see DSP0223)	Mandatory	set of output parameters

1507

1508 8.4.11 InstanceModificationRequest payload element

1509 An *InstanceModificationRequest* payload element represents the data used in a PATCH instance request
1510 (see 8.14.4). The InstanceModificationRequest payload element specifies modifications of property
1511 values in a CIM instance.

1512 This version of this document defines property value replacement as the only type of property value
1513 modification.

1514 Documents defining payload representations for the CIM-RS protocol need to ensure future extensibility
1515 to other types of property value modifications, such as replacing, inserting or deleting elements in array
1516 properties, or setting properties to their class-defined default value without specifying that value.

1517 An InstanceModificationRequest payload element includes the following properties:

1518 **Table 23 – Properties of InstanceModificationRequest payload element**

Property name	Generic type	Requirement	Description
replace	PropertyValue [ ] (see DSP0223)	Mandatory	set of properties whose values get replaced with new values

1519 The generic type PropertyReplacement is defined as follows:

1520 **Table 24 – Properties of PropertyValue generic type**

Property name	Generic type	Requirement	Description
name	PropertyName (see DSP0223)	Mandatory	name of the property to be replaced
value	string (see DSP0223)	Mandatory	new value for the property to be replaced, in its string representation as defined in DSP0004

1521

1522 8.4.12 InstanceQueryRequest payload element

1523 An *InstanceQueryRequest* payload element represents the data used in a POST instance query request
1524 (see 8.20.1).

1525 An InstanceQueryRequest payload element includes the following properties:

1526 **Table 25 – Properties of InstanceQueryRequest payload element**

Property name	Generic type	Requirement	Description
querystring	QueryString (see DSP0223)	Mandatory	query string in the query language identified by QueryLanguage
querylanguage	Query-Language (see DSP0223)	Mandatory	query language used in QueryString

1527 ### 8.4.13 ErrorResponse payload element

1528 An *ErrorResponse* payload element represents an error response to any request.

1529 An ErrorResponse payload element includes the following properties:

1530 **Table 26 – Properties of ErrorResponse payload element**

Property name	Generic type	Requirement	Description
statuscode	integer (see DSP0223)	Optional	CIM status code
statusdescription	string (see DSP0223)	Optional	CIM status description
errors	Instance-Specification [ ] (see DSP0223)	Optional	set of embedded instances of class CIM_Error, each specifying an error message

1531

1532 ## 8.5 WBEM server and listener operations

1533 This clause defines operations that target a WBEM server or WBEM listener as a whole (that is, without
1534 targeting resources within a WBEM server).

1535 ### 8.5.1 OPTIONS

1536 **Purpose:** Retrieves WBEM server or WBEM listener options

1537 **HTTP method:** OPTIONS

1538 **Resource URI:** *

1539 **URI query parameters:** None

1540 **Request headers:** Host

1541 **Request payload:** None

1542 **Response headers:** Date, CIM-RS specific header fields as defined in Table 27 and Table 28

1543 **Response payload:** None

1544 **Generic operation:** N/A

Description:

An implementation of the HTTP OPTIONS method for the resource "*" shall return information about the WBEM server that is targeted, using the response header fields described in Table 27:

Table 27 – Response header fields for OPTIONS to WBEM server

Name of response header field	Requirement level	Defined in
CIMRS-Content-Types	Mandatory	9.4.5
CIMRS-Entity-Tagging-Feature **(EXPERIMENTAL)**	Mandatory	9.4.6
CIMRS-Paged-Retrieval-Feature	Mandatory	9.4.7
CIMRS-Filter-Query-Languages	Mandatory	9.4.8
CIMRS-Instance-Query-Languages	Mandatory	9.4.9

An implementation of the HTTP OPTIONS method for the resource "*" shall return information about the WBEM listener that is targeted, using the response header fields described in Table 28:

Table 28 – Response header fields for OPTIONS to WBEM listener

Name of response header field	Requirement level	Defined in
CIMRS-Content-Types	Mandatory	9.4.5

On success, a status code 200 "OK" shall be returned.

Example HTTP conversation with a WBEM server:

```
OPTIONS *
Host: acme.com:5988

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
CIMRS-Content-Types: application/vnd.dmtf.cimrs+json;version=1.0.0
CIMRS-Etag-Feature: true
CIMRS-Paged-Retrieval-Feature: true
```

8.6 Namespace collection operations

This clause defines operations that target a *namespace collection resource* (see 7.2.1).

8.6.1 GET namespace collection

Purpose: Retrieves the set of all namespaces in a WBEM server

HTTP method: GET

Resource URI: Namespace collection resource (defined in 7.2.1)

URI query parameters: See Table 29

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: NamespaceCollection payload element (see 8.4.1) with links

Generic operation: *GetClassInstancesWithPath* on class *CIM_Namespace* in the interop namespace of the WBEM server

Table 29 – Query parameters for GET namespace collection

Name	Defined in
n (namespace)	7.3.13
ico (include class origin)	7.3.6
ip (included properties)	7.3.10
esbp (exclude subclass properties)	7.3.5

Description:

An implementation of the *GET namespace collection* operation shall conform to the behavior of the generic operation *GetClassInstancesWithPath* defined in DSP0223, when invoked with an *EnumClassPath* parameter value that references the CIM_Namespace class in the interop namespace of the WBEM server.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces"}
  },
  "namespaces": {
    "root/cimv2": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2"},
```

```
        "classes": {
          "href": "classes"},
        "qualifiers": {
          "href": "qualifiers"},
        "instancequery": {
          "href": "instancequery"}
      }
    },
    "interop": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/interop"},
        "classes": {
          "href": "classes"},
        "qualifiers": {
          "href": "qualifiers"},
        "instancequery": {
          "href": "instancequery"}
      }
    }
  }
}
```

NOTE: This example has an empty property list specified, so no property values are returned.

8.7 Namespace operations

This clause defines operations that target a *namespace resource* (see 7.2.2).

8.7.1 GET namespace

Purpose:	Retrieves a CIM namespace
HTTP method:	GET
Resource URI:	Namespace resource (defined in 7.2.2)
URI query parameters:	See Table 30
Request headers:	Host, Accept
Request payload:	None
Response headers:	Date, Content-Length, Content-Type, Etag
Response payload:	Namespace payload element (see 8.4.2) with links
Generic operation:	*GetInstance* on *CIM_Namespace* instance for that namespace, in the interop namespace of the WBEM server

Table 30 – Query parameters for GET namespace

Name	Defined in
ico (include class origin)	7.3.6
ip (included properties)	7.3.10

Description:

An implementation of the *GET namespace* operation shall conform to the behavior of the generic operation *GetInstance* defined in DSP0223, when invoked with an *InstancePath* parameter value that references the CIM_Namespace class in the interop namespace of the WBEM server.

On success, a status code 200 "OK" with the representation of the CIM_Namespace instance contained in the entity-body shall be returned, unless the following condition applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"root/cimv2": {
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2"},
    "classes": {
      "href": "classes"},
    "qualifiers": {
      "href": "qualifiers"},
    "instancequery": {
      "href": "instancequery"}
  }
}
```

8.8 Class collection operations

This clause defines operations that target a *class collection resource* (see 7.2.3).

8.8.1 GET class collection

Purpose: Retrieves the set of classes in a class collection

HTTP method: GET

Resource URI: Class collection resource (defined in 7.2.3)

URI query parameters: See Table 31

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: ClassCollection payload element (see 8.4.3) with links

Generic operation: *GetTopClassesWithPath*, *GetSubClassesWithPath*

Table 31 – Query parameters for GET class collection

Name	Defined in
c (class)	7.3.3
spc (superclass)	7.3.17
isbc (include subclasses)	7.3.12
iie (include inherited elements)	7.3.9
iq (include qualifiers)	7.3.11
ico (include class origin)	7.3.6

Description:

Depending on which query parameters are specified, the *GET class collection* operation shall conform to generic operations defined in DSP0223, as follows:

- *GetTopClassesWithPath*, if the spc query parameter is not specified
- *GetSuperClassesWithPath*, if the spc query parameter is specified

If the spc query parameter is not specified, the *GET class collection* operation shall conform to the generic operation *GetTopClassesWithPath* defined in DSP0223, and the returned class collection shall contain all classes in the namespace that have no superclasses.

If the spc query parameter is specified, the *GET class collection* operation shall conform to the generic operation *GetSubClassesWithPath* defined in DSP0223, and the returned class collection shall contain all classes in the namespace that have the specified class as a superclass.

If the ip query parameter is specified with an empty property list, the returned class collection shall consist of *self* URI values only.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

NOTE: This example specifies the spc query parameter.

```
GET /cimrs/namespaces/root%2Fcimv2/classes?spc=CIM_Example
Host: acme.com:5988
```

```
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes"},
    "namespace": {
      "href": ".."}
  },
  "classes": {
    "CIM_SubClass1OfExample": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_S
ubClass1OfExample"},
        "namespace": {
          "href": "../.."},
        "instances": {
          "href": "instances"},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
      },
      "superclass": "CIM_Example",
      "qualifiers": { . . . },
      "properties": { . . . }
    },
    "CIM_SubClass2OfExample": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_S
ubClass2OfExample"},
        "namespace": {
          "href": "../.."},
        "instances": {
          "href": "instances"},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
```

```
      },
      "superclass": "CIM_Example",
      "qualifiers": { . . . ],
      "properties": { . . . ],
      "methods": { . . . ]
    }
  }
}
```

8.8.2 POST class collection

Purpose: Creates a CIM class

HTTP method: POST

Resource URI: Class collection resource (defined in 7.2.3)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type

Request payload: Class payload element (see 8.4.4) without links

Response headers: Date, Location

Response payload: None

Generic operation: *CreateClass*

Description:

The *POST class collection* operation shall conform to the *CreateClass* generic operation defined in DSP0223.

On success, a status code 201 "Created" with the HTTP response header "Location" set to the newly-created instance as indicated by its ClassPath shall be returned.

Example HTTP conversation (using JSON):

```
POST /cimrs/namespaces/root%2Fcimv2/classes
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"CIM_NewClass": {
  "superclass": "CIM_ManagedElement",
  "qualifiers": { . . . },
  "properties": { . . . ],
  "methods": { . . . ]
}

HTTP/1.1 201 Created
Date: Wed, 29 Apr 2009 08:47:22 GMT
Location: http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_NewClass
```

8.9 Class operations

This clause defines operations that target a *class resource* (see 7.2.4).

The *ModifyClass* generic operation defined in DSP0223 does not provide for modifying a class only partially. For this reason, a *PATCH class* operation has not been defined in this document.

8.9.1 GET class

Purpose: Retrieves a CIM class

HTTP method: GET

Resource URI: Class resource (defined in 7.2.4)

URI query parameters: See Table 32

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: Class payload element (see 8.4.4) with links

Generic operation: *GetClass*

Table 32 – Query parameters for GET class

Name	Defined in
iq (include qualifiers)	7.3.11
ico (include class origin)	7.3.6
ip (included properties)	7.3.10

Description:

The *GET class* operation shall conform to the *GetClass* generic operation defined in DSP0223.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless the following condition applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Mon, 27 Apr 2009 17:02:09 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"CIM_RegisteredProfile": {
  "links": {
```

```
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Regis
teredProfile"},
    "namespace": {
      "href": "../.."},
    "instances": {
      "href": "instances"},
    "methodinvocation": {
      "href": "methodinvocation"},
    "associators": {
      "href": "associators"},
    "references": {
      "href": "references"}
  },
  "superclass": "CIM_ManagedElement",
  "qualifiers": { . . . },
  "properties": { . . . ]
}
```

8.9.2 DELETE class

Purpose: Deletes a CIM class

HTTP method: DELETE

Resource URI: Class resource (defined in 7.2.4)

URI query parameters: See Table 33

Request headers: Host, Accept

Request payload: None

Response headers: Date

Response payload: None

Generic operation: *DeleteClass*

Table 33 – Query parameters for DELETE class

Name	Defined in
dd (delete dependents)	7.3.3

Description:

The *DELETE class* operation shall conform to the *DeleteClass* generic operation defined in DSP0223.

On success, a status code 204 "No Content" with an empty entity-body shall be returned.

Example HTTP conversation (using JSON):

```
DELETE /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile
Host: acme.com:5988
```

```
HTTP/1.1 204 No Content
Date: Mon, 27 Apr 2009 17:02:09 GMT
```

8.9.3 PUT class

Purpose: Replaces a CIM class

HTTP method: PUT

Resource URI: Class resource (defined in 7.2.4)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type

Request payload: Class payload element (see 8.4.4) without links

Response headers: Date

Response payload: None

Generic operation: *ModifyClass*

Description:

The *PUT class* operation shall conform to the *ModifyClass* generic operation defined in DSP0223.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
PUT /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"CIM_RegisteredProfile": {
  "superclass": "CIM_ManagedElement",
  "qualifiers": { . . . ],
  "properties": { . . . ]
}

HTTP/1.1 200 OK
Date: Mon, 27 Apr 2009 17:02:09 GMT
```

8.10 Class associator collection operations

This clause defines operations that target a *class associator collection resource* (see 7.2.5).

8.10.1 GET class associator collection

Purpose: Retrieves the list of CIM classes that are associated with a given source class

HTTP method: GET

Resource URI: Class associator collection resource (defined in 7.2.5)

URI query parameters: See Table 34

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: Class collection payload element (see 8.4.3) with links

Generic operation: *GetAssociatedClassesWithPath*

Table 34 – Query parameters for GET class associator collection

Name	Defined in
c (class)	7.3.3
rcn (referencing class name)	7.3.15
acn (associated class name)	7.3.1
srn (source role name)	7.3.18
arn (associated role name)	7.3.2
iq (include qualifiers)	7.3.11
ico (include class origin)	7.3.6
ip (included properties)	7.3.10

Description:

The *GET class associator collection* operation shall conform to the *GetAssociatedClassesWithPath* generic operation defined in DSP0223, with the following modified behavior:

- The result of successive range requests may contain duplicate instances; duplicate instances can be removed by the client by comparing the instances' *self* URI value as defined by *Collection format*.
- An empty *IncludedProperties* value corresponds to the *GetAssociatedClassPaths* generic operation. The collection format returned in this case shall consist of *self* URI values only.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_SourceExample/associators
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
```

```
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Sourc
eExample/associators"},
    "namespace": {
      "href": "../../.."}
  },
  "classes": {
    "CIM_AssociatedExample": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_A
ssociatedExample"},
        "namespace": {
          "href": "../.."},
        "instances": {
          "href": "instances"},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
      },
      "qualifiers": { . . . ],
      "properties": { . . . ]
    },
    . . . # more classes
  }
}
```

8.11 Class reference collection operations

This clause defines operations that target a *class reference collection resource* (see 7.2.6).

8.11.1 GET class reference collection

Purpose:	Retrieves the list of CIM association classes that reference a given source class
HTTP method:	GET
Resource URI:	Class reference collection resource (defined in 7.2.6)
URI query parameters:	See Table 35
Request headers:	Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: Class collection payload element (see 8.4.3) with links

Generic operation: *GetReferencingClassesWithPath*

Table 35 – Query parameters for GET class reference collection

Name	Defined in
c (class)	7.3.3
rcn (referencing class name)	7.3.15
acn (associated class name)	7.3.1
srn (source role name)	7.3.18
arn (associated role name)	7.3.2
iq (include qualifiers)	7.3.11
ico (include class origin)	7.3.6
ip (included properties)	7.3.10

Description:

The *GET class reference collection* operation shall conform to the *GetReferencingClassesWithPath* generic operation defined in DSP0223, with the following modified behavior:

- The result of successive range requests may contain duplicate classes; duplicate classes can be removed by the client by comparing the class *self* URI value as defined by *Collection format*.
- An empty *IncludedProperties* value corresponds to the *GetReferencingClassPaths* generic operation. The Collection format returned in this case shall consist of *self* URI values only.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_SourceExample/references
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
```

```
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Sourc
eExample/references"},
    "namespace": {
      "href": "../../.."}
  },
  "classes": {
    "CIM_AssociationExample": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_A
ssociationExample"},
        "namespace": {
          "href": "../.."},
        "instances": {
          "href": "instances"},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
      },
      "qualifiers": { . . . ],
      "properties": { . . . ]
    },
    . . . # more classes
  }
}
```

8.12 Class method operations

This clause defines operations that target a *class method resource* (see 7.2.7).

8.12.1 POST class method

Purpose:	Invokes a (static) CIM method on a class
HTTP method:	POST
Resource URI:	Class method invocation resource (defined in 7.2.7)
URI query parameters:	None
Request headers:	Host, Accept, Content-Length, Content-Type
Request payload:	MethodInvocationRequest payload element (see 8.4.9)
Response headers:	Date, Content-Length, Content-Type
Response payload:	MethodInvocationResponse payload element (see 8.4.10)

Generic operation: *InvokeStaticMethod*

Description:

The *POSTclass method* operation shall conform to the *InvokeStaticMethod* generic operation defined in DSP0223.

On success, a status code of either 200 "OK" or 202 "Accepted" shall be returned.

Example HTTP conversation (using JSON):

```
POST /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/methodinvocation
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"StaticMethod": {
  "parameters": {
    "Parm1": "Some input value"
  }
}

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"StaticMethod": {
  "returnvalue": 0,
  "parameters": {
    "Parm2": "Some output value"
  }
}
```

8.13 Instance collection operations

This clause defines operations that target an *instance collection resource* (see 7.2.8).

8.13.1 GET instance collection

Purpose: Retrieves the list of CIM instances of a given CIM class

HTTP method: GET

Resource URI: Instance collection resource (defined in 7.2.8)

URI query parameters: See Table 36

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: InstanceCollection payload element (see 8.4.5) with links

Generic operation: *GetClassInstancesWithPath*

Table 36 – Query parameters for GET instance collection

Name	Defined in
fql (filter query language)	7.3.6
fqs (filter query string)	7.3.7
ico (include class origin)	7.3.6
ip (included properties)	7.3.10
esbp (exclude subclass properties)	7.3.5

Description:

The *GET instance collection* operation shall conform to the *GetClassInstancesWithPath* generic operation defined in DSP0223, with the following modified behavior:

- The result of successive range requests may contain duplicate instances; duplicate instances can be removed by the client by comparing the instances' *self* URI value as defined by *Collection format*.
- An empty *IncludedProperties* value corresponds to the *GetClassInstancePaths* generic operation. The Collection format returned in this case shall consist of *self* URI values only.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Sourc
eExample/instances"},
    "class": {
      "href": ".."}
  },
  "instances": [
    {
      "links": {
```

```
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_R
egisteredProfile/instances/DMTF%3AFan%3A1.1.0"},
        "class": {
          "href": "../.."},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
      },
      "class": "CIM_RegisteredProfile",
      "properties": {
        "InstanceID": "DMTF:Fan:1.1.0",
        "RegisteredName": "Fan",
        "RegisteredOrganization": 2,
        "RegisteredVersion": "1.1.0",
        . . . # more properties
      }
    },
    . . . # more instances
  ]
}
```

8.13.2 POST instance collection

Purpose: Creates a CIM instance

HTTP method: POST

Resource URI: Instance collection resource (defined in 7.2.8)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type

Request payload: Instance payload element (see 8.4.6) without links

Response headers: Date, Location

Response payload: None

Generic operation: *CreateInstance*

Description:

The *POST instance collection* operation shall conform to the *CreateInstance* generic operation defined in DSP0223.

On success, a status code 201 "Created" with the HTTP response header "Location" set to the newly created instance as indicated by its InstancePath shall be returned.

Example HTTP conversation (using JSON):

```
POST /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "class": "CIM_RegisteredProfile",
  "properties": {
    "RegisteredName": "Fan",
    "RegisteredOrganization": 2,
    "RegisteredVersion": "1.1.0",
    . . . # more properties
  }
}

HTTP/1.1 201 Created
Date: Wed, 29 Apr 2009 08:47:22 GMT
Location: http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Registered
Profile/instances/DMTF%3AFan%3A1.1.0
```

NOTE: The key property InstanceID is not provided in the request, since key property values are determined by the server. In this example, the InstanceID value has been determined by the server from the provided values of the RegisteredOrganization, RegisteredName, and RegisteredVersion properties.

8.14 Instance operations

This clause defines operations that target an *instance resource* (see 7.2.9).

8.14.1 GET instance

Purpose: Retrieves a CIM instance

HTTP method: GET

Resource URI: Instance resource (defined in 7.2.9)

URI query parameters: See Table 37

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type, Etag

Response payload: Instance payload element (see 8.4.6)

Generic operation: *GetInstance*

Table 37 – Query parameters for GET instance

Name	Defined in
ico (include class origin)	7.3.6
ip (included properties)	7.3.10

Description:

The *GET instance* operation shall conform to the *GetInstance* generic operation defined in DSP0223.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless the following condition applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances/DMTF%3AFan%3A1.1.0
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances/DMTF%3AFan%3A1.1.0"},
    "class": {
      "href": "../.."},
    "methodinvocation": {
      "href": "methodinvocation"},
    "associators": {
      "href": "associators"},
    "references": {
      "href": "references"}
  },
  "class": "CIM_RegisteredProfile",
  "properties": {
    "InstanceID": "DMTF:Fan:1.1.0",
    "RegisteredName": "Fan",
    "RegisteredOrganization": 2,
    "RegisteredVersion": "1.1.0",
    . . . # more properties
  }
}
```

8.14.2 DELETE instance

Purpose: Deletes a CIM instance

HTTP method: DELETE

Resource URI: Instance resource (defined in 7.2.9)

URI query parameters: None

Request headers: Host

Request payload: None

Response headers: Date

Response payload: None

Generic operation: *DeleteInstance*

Description:

The *DELETE instance* operation shall conform to the *DeleteInstance* generic operation defined in DSP0223.

On success, a status code 204 "No Content" with an empty entity-body shall be returned.

Example HTTP conversation (using JSON):

```
DELETE /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances/DMTF%
3AFan%3A1.1.0
Host: acme.com:5988

HTTP/1.1 204 No Content
Date: Wed, 29 Apr 2009 08:47:22 GMT
```

8.14.3 PUT instance

Purpose: Replaces a CIM instance

HTTP method: PUT

Resource URI: InstancePath (defined in 7.2.9)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type, If-Match (EXPERIMENTAL)

Request payload: Instance payload element (see 8.4.6) without links

Response headers: Date

Response payload: None

Generic operation: *ModifyInstance* with *IncludedProperties* being Null

Description:

The *PUT instance* operation shall conform to the *ModifyInstance* generic operation defined in DSP0223, when its *IncludedProperties* parameter is Null. The *ModifyInstance* generic operation will modify all properties of the CIM instance that are modifiable, in this case.

The instance payload element does not need to specify all properties exposed by the creation class of the instance. The values of modifiable properties that are not specified in the instance payload element shall be set to the class-defined default value of the property, or to Null if no such default value is defined.

NOTE: This behavior for properties that are requested to be changed but not specified in the instance payload element is consistent with DSP0200. In contrast, DSP0223 requires that the property is set to Null in this case, even when a non-Null default value for the property is defined in the class.

The values of non-modifiable properties shall not be changed by the WBEM server. *PUT instance* operations that specify non-modifiable properties in the instance payload element with a value that is different from their current value shall be rejected.

EXPERIMENTAL

In addition, the *PUT instance* operation shall reject the modification if an If-Match header field is provided, and the entity tag provided as its value does not match the current entity tag (that is, the value of the Generation property) of the resource.

EXPERIMENTAL

On success, either of the following status codes shall be returned:

- 200 "OK". with an empty entity-body
- 202 "Accepted" with an empty entity-body

Example HTTP conversation (using JSON):

```
PUT /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances/DMTF%3AFan%3A1.1.0
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "class": " CIM_RegisteredProfile",
  "properties": {
    "RegisteredName": "Fan",
    "RegisteredOrganization": 2,
    "RegisteredVersion": "1.1.1",
    "Caption": "A changed caption"
  }
}

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
```

NOTE: In this example, it is assumed that all provided properties are modifiable. The modifiable properties not provided are set to their class-defined default values or to Null. The value of the InstanceID key property remains unchanged, since key properties are never modifiable.

8.14.4 PATCH instance

Purpose: Modifies properties of a CIM instance

HTTP method: PATCH

Resource URI: Instance resource (defined in 7.2.9)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type, If-Match (EXPERIMENTAL)

Request payload: InstanceModificationRequest payload element (see 8.4.11)

Response headers: Date

Response payload: None

Generic operation: *ModifyInstance* with *IncludedProperties* being non-Null

Description:

The *PATCH instance* operation shall conform to the *ModifyInstance* generic operation defined in DSP0223, when its *IncludedProperties* parameter is non-Null. In this case, the *ModifyInstance* generic operation will modify only the properties that are specified in its *IncludedProperties* parameter and that are modifiable.

The InstanceModificationRequest payload element (see 8.4.11) specifies the requested changes for the instance.

The InstanceModificationRequest payload element does not need to specify requested changes for all properties exposed by the creation class of the instance. Properties for which no changes are specified in the InstanceModificationRequest payload element shall be left unchanged.

EXPERIMENTAL

In addition, the *PATCH instance* operation shall reject the modification if an If-Match header field is provided, and the entity tag provided as its value does not match the current entity tag (that is, the value of the Generation property) of the resource.

EXPERIMENTAL

On success, either of the following status codes shall be returned:

- 200 "OK". with an empty entity-body
- 202 "Accepted" with an empty entity-body

Example HTTP conversation (using JSON):

```
PATCH /cimrs/namespaces/root%2Fcimv2/classes/CIM_RegisteredProfile/instances/DMTF%3
AFan%3A1.1.0
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "replace": {
    "Caption": "A changed caption"
  }
}

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
```

NOTE: In this example, only the value of the Caption property is requested to be changed. It is assumed to be modifiable. All other property values remain unchanged, since they are not requested to be changed.

8.15 Instance associator collection operations

This clause defines operations that target an *instance associator collection resource* (see 7.2.10).

8.15.1 GET instance associator collection

Purpose: Retrieves the list of CIM instances that are associated with a given source instance

HTTP method: GET

Resource URI: Instance associator collection resource (defined in 7.2.10)

URI query parameters: See Table 38

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: InstanceCollection payload element (see 8.4.5)

Generic operation: *GetAssociatedInstancesWithPath*

Table 38 – Query parameters for GET instance associator collection

Name	Defined in
fql (filter query language)	7.3.6
fqs (filter query string)	7.3.7
rcn (referencing class name)	7.3.15
acn (associated class name)	7.3.1
srn (source role name)	7.3.18
arn (associated role name)	7.3.2
ico (include class origin)	7.3.6
ip (included properties)	7.3.10
esbp (exclude subclass properties)	7.3.5

Description:

The *GET instance associator collection* operation shall conform to the *GetAssociatedInstancesWithPath* generic operation defined in DSP0223, with the following modified behavior:

- The result of successive range requests may contain duplicate instances; duplicate instances can be removed by the client by comparing the instances' *self* URI value as defined by *Collection format*.
- An empty *IncludedProperties* value corresponds to the *GetAssociatedInstancePaths* generic operation. The Collection format returned in this case shall consist of *self* URI values only.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_SourceExample/instances/acme1/associ
ators
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Sourc
eExample/instances/acme1/associators"},
    "class": {
      "href": "../../.."}
  },
  "instances": [
    {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_A
ssociatedExample/instances/acme2"},
        "class": {
          "href": "../.."},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
      },
      "class": "CIM_AssociatedExample",
      "properties": {
        "InstanceID": "acme2",
        "Caption": "An example associated instance",
        . . . # more properties
      }
    },
    . . . # more instances
  ]
```

```
}
```

8.16 Instance reference collection operations

This clause defines operations that target an *instance reference collection resource* (see 7.2.11).

8.16.1 GET instance reference collection

Purpose: Retrieves the list of CIM association instances that reference a given source instance

HTTP method: GET

Resource URI: Instance reference collection resource (defined in 7.2.11)

URI query parameters: See Table 39

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: InstanceCollection payload element (see 8.4.5)

Generic operation: *GetReferencingInstancesWithPath*

Table 39 – Query parameters for GET instance reference collection

Name	Defined in
fql (filter query language)	7.3.6
fqs (filter query string)	7.3.7
rcn (referencing class name)	7.3.15
can (associated class name)	7.3.1
srn (source role name)	7.3.18
arn (associated role name)	7.3.2
ico (include class origin)	7.3.6
ip (included properties)	7.3.10
esbp (exclude subclass properties)	7.3.5

Description:

The *GET instance reference collection* operation shall conform to the *GetReferencingInstancesWithPath* generic operation defined in DSP0223, with the following modified behavior:

- The result of successive range requests may contain duplicate instances; duplicate instances can be removed by the client by comparing the instances' *self* URI value as defined by *Collection format*.
- An empty *IncludedProperties* value corresponds to the *GetClassInstancePaths* generic operation. The Collection format returned in this case shall consist of *self* URI values only.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/classes/CIM_SourceExample/instances/acme1/refere
nces
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_Sourc
eExample/instances/acme1/references"},
    "class": {
      "href": "../../.."}
  },
  "instances": [
    {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/classes/CIM_A
ssociationExample/instances/http%3A%2F%2Facme.com%3A5988%2Fcimrs%2Fnamespaces%2Froo
t%252Fcimv2%2Fclasses%2FCIM_SourceExample%2Finstances%2Facme1,http%3A%2F%2Facme.com
%3A5988%2Fcimrs%2Fnamespaces%2Froot%252Fcimv2%2Fclasses%2FCIM_AssociatedExample%2Fi
nstances%2Facme2"},
        "class": {
          "href": "../.."},
        "methodinvocation": {
          "href": "methodinvocation"},
        "associators": {
          "href": "associators"},
        "references": {
          "href": "references"}
      },
      "class": "CIM_AssociationExample",
      "properties": {
        "InstanceID": "acme1,acme2",
        "Caption": "An example association instance referencing the given
instance",
        . . . # more properties
      }
```

```
    },
    . . . # more instances
  ]
}
```

NOTE: In this example, the key properties of the CIM_AssociationExample association class are its two reference properties, as usually in the CIM Schema. The key property values are again WBEM resource identifiers, which need to be percent-escaped when used as instance key values in the WBEM resource identifier of the association instance.

8.17 Instance method operations

This clause defines operations that target an *instance method resource* (see 7.2.12).

8.17.1 POST instance method

Purpose: Invokes a (static or non-static) CIM method on a CIM instance

HTTP method: POST

Resource URI: Instance method invocation resource (defined in 7.2.12)

URI query parameters: None

Request headers: Host, Accept, Content-Length, Content-Type

Request payload: MethodInvocationRequest payload element (see 8.4.9)

Response headers: Date, Content-Length, Content-Type

Response payload: MethodInvocationResponse payload element (see 8.4.10)

Generic operation: *InvokeMethod*

Description:

The *POST instance method* operation shall conform to the *InvokeMethod* generic operation defined in DSP0223.

On success, a status code of either 200 "OK" or 202 "Accepted" shall be returned.

Example HTTP conversation (using JSON):

```
POST /cimrs/namespaces/root%2Fcimv2/classes/CIM_Example/instances/acme1/methodinvoc
ation
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"SomeMethod": {
  "parameters": {
    "Parm1": "Some input data"
  }
}

HTTP/1.1 200 OK
```

```
Date: Wed, 29 Apr 2009 08:47:22 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"SomeMethod": {
  "returnvalue": 42,
  "parameters": {
    "Parm2": "Some output data"
  }
}
```

8.18 Qualifier type collection operations

This clause defines operations that target a *qualifier type collection resource* (see 7.2.13).

8.18.1 GET qualifier type collection

Purpose: Retrieves the list of CIM qualifier types in a CIM namespace

HTTP method: GET

Resource URI: Qualifier type collection resource (defined in 7.2.13)

URI query parameters: See Table 40

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: QualifierTypeCollection payload element (see 8.4.7) with links

Generic operation: *EnumerateQualifierTypesWithPath*

Table 40 – Query parameters for GET qualifier type collection

Name	Defined in
q (qualifier)	7.3.15

Description:

The *GET qualifier type collection* operation shall conform to the *EnumerateQualifierTypesWithPath* generic operation defined in DSP0223, with the following modified behavior:

- The result of successive range requests may contain duplicate qualifier types; duplicate qualifier types can be removed by the client by comparing the qualifier types' *self* URI value as defined by *Collection format*.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless one of the following conditions applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.
- For range requests, a status code 206 "Partial Content" with the appropriate subset of entities contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/qualifiers
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Mon, 27 Apr 2009 17:02:09 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/qualifiers"},
    "namespace": {
      "href": ".."}
  },
  "qualifiertypes": {
    "Counter": {
      "links": {
        "self": {
          "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/qualifiers/Co
unter"},
        "namespace": {
          "href": "../.."}
      },
      "type": "boolean",
      "default": false,
      "scope": [ "property", "method", "parameter" ],
      "flavor": [ "ToSubclass" ]
    },
    . . . # more qualifier types
  }
}
```

8.18.2 POST qualifier type collection

Purpose:	Creates a CIM qualifier type
HTTP method:	POST
Resource URI:	Qualifier type collection resource (defined in 7.2.13)
URI query parameters:	None
Request headers:	Host, Content-Length, Content-Type
Request payload:	QualifierType payload element (see 8.4.8) without links
Response headers:	Date, Location
Response payload:	None

Generic operation: *CreateQualifierType*

Description:

The *POST qualifier type collection* operation shall conform to the *CreateQualifierType* generic operation defined in DSP0223.

On success, a status code 201 "Created" with the HTTP response header "Location" set to the newly-created instance as indicated by its QualifierTypePath shall be returned.

Example HTTP conversation (using JSON):

```
POST /cimrs/namespaces/root%2Fcimv2/qualifiers
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json
"NewQualifier": {
  "type": "string",
  "default": null,
  "scope": [ "property" ],
  "flavor": [ "Restricted" ]
}

HTTP/1.1 201 Created
Date: Mon, 27 Apr 2009 17:02:09 GMT
Location: http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/qualifiers/NewQualifie
r
```

8.19 Qualifier type operations

This clause defines operations that target a *qualifier type resource* (see 7.2.14).

The *ModifyQualifierType* generic operation defined in DSP0223 does not provide for modifying a qualifier type only partially. For this reason, a *PATCH qualifier type* operation has not been defined in this document.

8.19.1 GET qualifier type

Purpose: Retrieves a CIM qualifier type

HTTP method: GET

Resource URI: Qualifier type resource (defined in 7.2.14)

URI query parameters: None

Request headers: Host, Accept

Request payload: None

Response headers: Date, Content-Length, Content-Type

Response payload: QualifierType payload element (see 8.4.8) with links

Generic operation: *GetQualifierType*

Description:

The *GET qualifier type* operation shall conform to the *GetQualifierType* generic operation defined in DSP0223.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless the following condition applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.

Example HTTP conversation (using JSON):

```
GET /cimrs/namespaces/root%2Fcimv2/qualifiers/Alias
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json

HTTP/1.1 200 OK
Date: Mon, 27 Apr 2009 17:02:09 GMT
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"Alias": {
  "links": {
    "self": {
      "href": "http://acme.com:5988/cimrs/namespaces/root%2Fcimv2/qualifiers/Alias"
},
    "namespace": {
      "href": "../.."}
  },
  "type": "string",
  "default": null,
  "scope": [ "property" ],
  "flavor": [ "Restricted" ]
}
```

8.19.2 DELETE qualifier type

Purpose:	Deletes a CIM qualifier type
HTTP method:	DELETE
Resource URI:	Qualifier type resource (defined in 7.2.14)
URI query parameters:	None
Request headers:	Host
Request payload:	None
Response headers:	Date
Response payload:	None
Generic operation:	*DeleteQualifierType*

Description:

The *DELETE qualifier type* operation shall conform to the *DeleteQualifierType* generic operation defined in DSP0223.

On success, a status code 204 "No Content" with an empty entity-body shall be returned.

Example HTTP conversation (using JSON):

```
DELETE /cimrs/namespaces/root%2Fcimv2/qualifiers/Alias
Host: acme.com:5988

HTTP/1.1 204 No Content
Date: Mon, 27 Apr 2009 17:02:09 GMT
```

8.19.3 PUT qualifier type

Purpose: Replaces a CIM qualifier type

HTTP method: PUT

Resource URI: Qualifier type resource (defined in 7.2.14)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type

Request payload: QualifierType payload element (see 8.4.8) without links

Response headers: Date

Response payload: None

Generic operation: *ModifyQualifierType*

Description:

The *PUT qualifier type* operation shall conform to the *ModifyQualifierType* generic operation defined in DSP0223.

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned.

Example HTTP conversation (using JSON):

```
PUT /cimrs/namespaces/root%2Fcimv2/qualifiers/Alias
Host: acme.com:5988
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
"Alias": {
  "type": "uint8",
  "default": null,
  "scope": [ "property" ],
  "flavor": [ "Restricted" ]
}

HTTP/1.1 200 OK
```

```
Date: Mon, 27 Apr 2009 17:02:09 GMT
```

8.20 Instance query operations

This clause defines operations that target an *instance query resource* (see 7.2.15).

EXPERIMENTAL

8.20.1 POST instance query

Purpose: Execute a query for instances on a CIM namespace and return the query result

HTTP method: POST

Resource URI: Instance query resource (defined in 7.2.15)

URI query parameters: None

Request headers: Host, Content-Length, Content-Type, Accept

Request payload: InstanceQueryRequest payload element (see 8.4.12)

Response headers: Date, Content-Length, Content-Type

Response payload: InstanceCollection payload element (see 8.4.5) for the instances representing the query result, without links

Generic operation: An assumed future *ExecuteQuery* operation with the abilities of the ExecQuery operation defined in DSP0200

Description:

The *POST instance query* operation shall conform to the (future) *ExecuteQuery* generic operation defined in DSP0223, with the following added behavior:

- The query shall return an instance collection
- The query shall not modify any CIM entities

On success, a status code 200 "OK" with the resource representation contained in the entity-body shall be returned, unless the following condition applies:

- For conditional requests where validators match, a status code 304 "Not Modified" and an empty entity-body should be returned.

Example HTTP conversation (using JSON):

```
POST /cimrs/namespaces/root%2Fcimv2/instancequery
Host: acme.com:5988
Accept: application/vnd.dmtf.cimrs+json
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "querystring": "SELECT OBJECTPATH(CIM_StorageExtent) AS Path, ElementName
                  FROM CIM_StorageExtent",
  "querylanguage": "DMTF:CQL"
```

```
}

HTTP/1.1 200 OK
Content-Length: XXX
Content-Type: application/vnd.dmtf.cimrs+json;version=1.0.0
{
  "instances": [
    {
      "class": "TBD",
      "properties": {
        "Path": "/namespaces/root%2Fcimv2/classes/CIM_StorageExtent/instances/acme50",
        "ElementName": "Example instance"
      }
    },
    . . . # more instances
  ]
}
```

EXPERIMENTAL

8.21 Functionality without specific operations

This clause describes functionality that does not have specific operations defined.

8.21.1 Subscription and other indication related functions

Subscription for indications and other indication-related functions (such as retrieving and managing indication filters, filter collections, listener destinations or indication services) can be performed using other operations defined in this document, if the *Indications Profile* (DSP1054) is implemented.

8.22 Potential future operations

This clause describes potential future operations, in order to get feedback on them.

8.22.1 Indication delivery

Purpose: Delivers an indication to a WBEM listener. Details of this operation are to be defined.

9 Usage of HTTP and HTTPS

9.1 HTTP and HTTPS version requirements

WBEM clients, WBEM servers, and WBEM listeners shall support the use of HTTP for the CIM-RS protocol. The following applies for this case:

- Version 1.1 of HTTP shall be supported as defined in RFC2616.
- Version 1.0 of HTTP shall not be supported.

WBEM clients, WBEM servers, and WBEM listeners should support the use of HTTPS for the CIM-RS protocol. If they do, the following applies:

- HTTPS shall be supported as defined in RFC2818.
- The secure sockets layer used with HTTPS shall be TLS 1.1 as defined in RFC4346; in addition, the secure sockets layer should be TLS 1.2 as defined in RFC5246.
- SSL 2.0, SSL 3.0, or TLS 1.0 (also known as SSL 3.1) shall not be supported as the secure sockets layer used with HTTPS because of security issues in these versions.
- Within HTTPS, version 1.1 of HTTP shall be supported as defined in RFC2616.

NOTE 1 HTTPS should not be confused with *Secure HTTP* defined in RFC2660.

NOTE 2 RFC2818 describes the use of TLS 1.0 and higher but not the use of SSL 2.0 or 3.0.

NOTE 3 RFC2818 describes the use of HTTP/1.0 and HTTP/1.1 within HTTPS.

9.2 Authentication requirements

This clause describes requirements and considerations for authentication between WBEM clients, WBEM servers, and WBEM listeners. Specifically, authentication happens from WBEM clients to WBEM servers for operation messages, and from WBEM servers to WBEM listeners for indication delivery messages.

9.2.1 Operating without authentication

WBEM clients, WBEM servers, and WBEM listeners may support operating without the use of authentication.

This may be acceptable in environments such as physically secured networks or between components on the same operating system.

9.2.2 HTTP basic authentication

HTTP basic authentication provides a rudimentary level of authentication, with the major weakness that the client password is part of the HTTP headers in unencrypted form.

WBEM clients, WBEM servers, and WBEM listeners may support HTTP basic authentication as defined in RFC2617.

HTTP basic authentication may be acceptable in environments such as physically secured networks, between components on the same operating system, or when the messages are encrypted by using HTTPS.

9.2.3 HTTP digest authentication

HTTP digest authentication verifies that both parties share a common secret without having to send that secret in the clear. Thus, it is more secure than HTTP basic authentication.

WBEM clients, WBEM servers, and WBEM listeners should support HTTP digest authentication as defined in RFC2617.

9.2.4 Other authentication mechanisms

WBEM clients, WBEM servers, and WBEM listeners may support authentication mechanisms not covered by RFC2617. One example of such a mechanism is public key certificates as defined in X.509.

9.3 Message encryption

Encryption of HTTP messages can be supported by the use of HTTPS and its secure sockets layer.

Requirements for the use of HTTPS and its underlying secure sockets are defined in 9.1.

It is important to understand that authentication and encryption of messages are separate issues: Encryption of messages requires the use of HTTPS, while the authentication mechanisms defined in 9.2 can be used with both HTTP and HTTPS.

9.4 HTTP header fields

This clause describes the use of HTTP header fields within the CIM-RS protocol, and it defines additional CIM-RS specific header fields.

9.4.1 Accept

The rules for the Accept request header field defined in RFC2616 apply. This clause defines additional constraints on its use.

The Accept request header field may be provided on the request message of any operation that may return a response payload.

If provided, it shall specify all MIME media types identifying valid CIM-RS payload representations that are supported by the WBEM client or WBEM server, using the following format (defined in ABNF):

```
Accept = "Accept" WS ":" 1*( WS media-type [ WS accept-params ] )
```

where `media-type` is a MIME media type identifying a valid CIM-RS payload representation, and `accept-params` is defined in RFC2616.

If an Accept request header field is provided, WBEM servers shall use one of the valid CIM-RS payload representations identified in the Accept request header field for the response payload.

If none of the CIM-RS payload representations identified in the Accept request header field is supported by the WBEM server or WBEM listener, it shall return 406 "not acceptable".

NOTE: RFC2616 recommends returning 406 "not acceptable" in this case, but it does not require it.

WBEM servers and WBEM listeners shall ignore any invalid `media-type` values, as well as the use of media ranges such as "*/*".

Support for `accept-params` is optional.

If no Accept request header field is provided, WBEM servers and WBEM listeners may use any valid CIM-RS payload representation for the response payload.

WBEM clients and WBEM servers are not required to identify the same set of CIM-RS payload representations in an Accept request header field as in previous requests.

When multiple CIM-RS payload representations are indicated in an Accept request header field, WBEM servers and WBEM listeners are not required to use the same CIM-RS payload representation as in previous responses.

9.4.2 Content-Type

The rules for the Content-Type entity header field defined in RFC2616 apply. This clause defines additional constraints on its use.

As defined in RFC2616, the Content-Type entity header field shall be provided on the request message of any operation that passes a request payload and on the response message of any operation that returns a response payload.

The Content-Type entity header field shall specify the MIME media type identifying the CIM-RS payload representation that is used for the payload.

EXPERIMENTAL

9.4.3 Etag

The rules for the Etag response header field defined in RFC2616 apply. This clause defines additional constraints on its use.

The Etag response header field shall be provided in any response by a WBEM server for which all of the following criteria are satisfied:

- The entity tagging feature (see 8.3.1) is implemented by the WBEM server.
- The response is for GET instance (see 8.14.1) or GET namespace (see 8.7.1).
- The CIM instance has a non-Null value of its (string typed) Generation property.

In this case, the Etag response header field shall be specified using the following format (defined in ABNF):

```
Etag = "Etag" WS ":" generation
```

where `generation` is the value of the Generation property of the CIM instance represented in the response.

Otherwise, the Etag response header field shall not be provided by a WBEM server.

The Etag response header field shall not be provided in any responses from a WBEM listener.

EXPERIMENTAL

EXPERIMENTAL

9.4.4 If-Match

The rules for the If-Match request header field defined in RFC2616 apply. This clause defines additional constraints on its use.

The If-Match request header field shall be provided in any request by a WBEM client for which all of the following criteria are satisfied:

- The entity tagging feature (see 8.3.1) is implemented by the WBEM client.
- The request is for a PUT instance (see 8.14.3).
- The new instance supplied in the PUT request is based on an instance that was retrieved earlier and that had the Etag header field (see 9.4.3) set.

In this case, the If-Match request header field shall be specified using the following format (defined in ABNF):

```
If-Match = "If-Match" WS ":" generation
```

where `generation` is the value of the Etag header field of the CIM instance that is the basis for the modification.

Otherwise, the If-Match request header field shall not be provided in any request by a WBEM client.

The If-Match request header field shall not be provided in any request by WBEM server to a WBEM listener.

EXPERIMENTAL

9.4.5 CIMRS-Content-Types

The CIMRS-Content-Types response header field identifies the CIM-RS payload representations supported by the WBEM server and WBEM listener by specifying their MIME media types, using the following format (defined in ABNF):

```
CIMRS-Content-Types = "CIMRS-Content-Types" WS ":" 1*( WS media-type )
```

where `media-type` is a MIME media type identifying a valid CIM-RS payload representation.

See 8.5.1 for its use in the HTTP OPTIONS method.

EXPERIMENTAL

9.4.6 CIMRS-Entity-Tagging-Feature

The CIMRS-Entity-Tagging-Feature response header field indicates with a boolean value whether the entity tagging feature (see 8.3.1) is implemented by the WBEM server, using the following format (defined in ABNF):

```
CIMRS-Etag-Feature = "CIMRS-Entity-Tagging-Feature" WS ":" ( "true" / "false" )
```

See 8.5.1 for its use in the HTTP OPTIONS method targeting a WBEM server.

EXPERIMENTAL

9.4.7 CIMRS-Paged-Retrieval-Feature

The CIMRS-Paged-Retrieval-Feature response header field indicates with a boolean value whether paged retrieval (see 8.3.2) is implemented by the WBEM server, using the following format (defined in ABNF):

```
CIMRS-Paged-Retrieval-Feature = "CIMRS-Paged-Retrieval-Feature" WS ":" ( "true" /
"false" )
```

See 8.5.1 for its use in the HTTP OPTIONS method targeting a WBEM server.

9.4.8 CIMRS-Filter-Query-Languages

The CIMRS-Filter-Query-Languages response header field indicates which query languages are supported for the fql (filter query language, see 7.3.6) query parameter, using the following format (defined in ABNF):

```
CIMRS-Filter-Query-Languages = "CIMRS-Filter-Query-Languages" WS ":" WS [ query-
language *( WS "," WS query-language ) ]
```

Where query-language is a string that uniquely identifies a query language.

See 8.5.1 for its use in the HTTP OPTIONS method targeting a WBEM server.

9.4.9 CIMRS-Instance-Query-Languages

The CIMRS-Instance-Query-Languages response header field indicates which query languages are supported for the query provided with a POST instance query operation (see 8.20.1), using the following format (defined in ABNF):

```
CIMRS-Instance-Query-Languages = "CIMRS-Instance-Query-Languages" WS ":" WS [
query-language *( WS "," WS query-language ) ]
```

Where query-language is a string that uniquely identifies a query language.

See 8.5.1 for its use in the HTTP OPTIONS method targeting a WBEM server.

10 Payload representation

Payload representations for the content of HTTP messages used by the CIM-RS protocol are expected to be described in separate documents.

This clause defines requirements for such CIM-RS payload representation descriptions.

10.1 Media type

A CIM-RS payload representation description shall define a MIME media type by which that payload representation can be uniquely identified within the set of all payload representations defined for the CIM-RS protocol.

This media type shall not use media ranges (that is, the asterisk character "*") for its type or subtype fields.

The combination of type and subtype fields of this media type shall change when the payload representation changes incompatibly.

It is recommended to encode the full version of the payload representation in this media type, preferrably as a media type parameter named "version".

A CIM-RS payload representation description shall define the rules used to determine whether a payload representation identified by a particular media type (including a version) is compatible to a consumer that understands a particular media type (including a version).

10.2 Payload element representations

A CIM-RS payload representation description shall define a representation for each payload element defined in this document (see 8.4).

Generic properties of payload elements may be represented in any way in the payload representation. The generic property name stated in the subclauses of in 8.4 does not need to be retained in the payload representation.

For example, in a JSON representation of a Namespace payload element (see 8.4.2), all of the following options would be valid for representing the "name" generic property for a namespace named "root/cimv2":

- as the JSON object name:

```
"root/cimv2": {
  # remaining generic properties and links
}
```

- as a JSON attribute with the same name as the generic property:

```
{
  "name": "root/cimv2",
  # remaining generic properties and links
}
```

- as a JSON attribute with a different name as the generic property:

```
{
```

```
  "namespace-name": "root/cimv2",
  # remaining generic properties and links
}
```

Links of payload elements may be represented in any way in the payload representation that retains the link name and target.

For example, in a JSON representation of a Namespace payload element (see 8.4.2), all of the following options would be valid for representing the links:

- as a JSON attribute named "links" which is a JSON array of JSON objects named with the link name that has the link target as an attribute:

```
{
  "links": {
    "self": {
      "href": "http://. . ."   # URI of the "self" link
    },
    # remaining links
  },
  # generic properties
}
```

- as a JSON attribute named "links" which is a JSON array of unnamed JSON objects that have both link name and link target as attributes:

```
{
  "links": [
    {
      "rel": "self",
      "href": "http://. . ."   # URI of the "self" link
    },
    # remaining links
  ],
  # generic properties
}
```

ANNEX A
(informative)

Known payload representations

This annex lists the CIM-RS payload representations known at the time of release of this document.

Table 41 – Known CIM-RS payload representations

Name	Underlying format	Defined in
CIM-RS Binding to JSON	Javascript Object Notation (JSON)	DSP-IS0202

It is expected that an XML-based payload representation for CIM-RS will be defined in the future.

ANNEX B
(informative)

Examples for structure of resource URIs

This annex describes examples on how the resource URIs for a CIM-RS implementation could be structured.

B.1 Example using CIM object types as URI segments

In this example, the guiding principle is that the terms for CIM object types (namespace, class, instance, qualifier type) and the terms for certain related resources (associator, reference, etc.) are used as the segments of the resource URIs. The term identifying the type of object is used in its plural form as one URI segment, followed by a segment that identifies the instance of that type of object within the context of the higher segments. Such tupels of segments are repeated along certain hierarchies of these CIM object types. In the last of such tupels in a URI, the second segment may be omitted to address the collection of the respective resources, instead of a single resource.

B.1.1 Namespace collection resource

A WBEM resource identifier addressing a namespace collection resource needs to conform to the structure defined in 7.2.1.

B.1.2 Namespace resource

A WBEM resource identifier addressing a namespace resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = NamespacePath

NamespacePath = NamespaceCollectionPath "/" NamespaceName
```

Where:

- `NamespaceName` is the percent-encoded name of the CIM namespace (see 7.1.2 for percent-encoding rules).

  NOTE Besides other possible transformations, applying these percent-encoding rules causes any slash ("/") characters in the namespace name to be represented as the string "%2F".

- `NamespaceCollectionPath` is defined in 7.2.1.

Examples:

- `http://acme.com:5988/cimrs/namespaces/interop`

  This resource identifier addresses the "interop" namespace of the WBEM server at port 5988 of host "acme.com", using HTTP.

- `/cimrs/namespaces/myns`

  This resource identifier addresses the "myns" namespace of a WBEM server whose host, port, and scheme are known by other means.

- `https://acme.com/cimrs/namespaces/root%2Fcimv2`

  This resource identifier addresses the "root/cimv2" namespace of the WBEM server at the default port 443 of host "acme.com", using HTTPS.

B.1.3 Class collection resource

A WBEM resource identifier addressing a class collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = ClassCollectionPath

ClassCollectionPath = NamespacePath "/classes"
```

Example:

- `/cimrs/namespaces/myns/classes`

  This resource identifier addresses the collection of all classes in the "myns" namespace of a WBEM server whose host, port, and scheme are known by other means.

B.1.4 Class resource

A WBEM resource identifier addressing a class resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = ClassPath

ClassPath = ClassCollectionPath "/" ClassName
```

Where:

- `ClassName` is the percent-encoded name of the CIM class (including its schema prefix). (See 7.1.2 for percent-encoding rules.)

Example:

- `/cimrs/namespaces/myns/classes/CIM_ManagedElement`

  This resource identifier addresses the "CIM_ManagedElement" class in the "myns" namespace of a WBEM server whose host, port, and scheme are known by other means.

B.1.5 Class associator collection resource

A WBEM resource identifier addressing a class associator collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = ClassAssociatorCollectionPath

ClassAssociatorCollectionPath = ClassPath "/associators"
```

Example:

- `/cimrs/namespaces/myns/classes/CIM_ManagedElement/associators`

  This resource identifier addresses the collection of all classes associated with the "CIM_ManagedElement" class in the "myns" namespace of a WBEM server whose host, port, and scheme are known by other means.

B.1.6 Class reference collection resource

A WBEM resource identifier addressing a class reference collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = ClassReferenceCollectionPath

ClassReferenceCollectionPath = ClassPath "/references"
```

Example:

- `/cimrs/namespaces/root%2Fcimv2/classes/CIM_ManagedElement/references`

  This resource identifier addresses the collection of all classes referencing the "CIM_ManagedElement" class in the "root/cimv2" namespace of a WBEM server whose host, port, and scheme are known by other means.

B.1.7 Class method invocation resource

A WBEM resource identifier addressing a class method invocation resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = ClassMethodInvocationPath

ClassMethodInvocationPath = ClassPath "/methodinvocation"
```

Example:

- `/cimrs/namespaces/root%2Fcimv2/classes/CIM_Example/methodinvocation`

  This resource identifier addresses the method invocation point exposed by the "CIM_Example" class in the "root/cimv2" namespace of a WBEM server whose host, port, and scheme are known by other means.

B.1.8 Instance collection resource

A WBEM resource identifier addressing an instance collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = InstanceCollectionPath

InstanceCollectionPath = ClassPath "/instances"
```

Example:

- `/cimrs/namespaces/myns/classes/CIM_ManagedElement/instances`

  This resource identifier addresses the collection of all instances in the "myns" namespace that have a creation class of "CIM_ManagedElement".

B.1.9 Instance resource

A WBEM resource identifier addressing an instance resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = InstancePath

InstancePath = InstanceCollectionPath "/" KeyList
```

```
KeyList             =
    OrdinaryKeyList /              ; for instances of ordinary classes
    AssociationKeyList             ; for instances of association classes

OrdinaryKeyList     =
    OrdinaryKeyPropertyValue *( "," OrdinaryKeyPropertyValue )
                                   ; list is ordered by property names

AssociationKeyList  =
    AssociationKeyValue *( "," AssociationKeyValue )
                                   ; list is ordered by property names

AssociationKeyValue =
    OrdinaryKeyPropertyValue /     ; for ordinary key properties
    "(" ReferenceValue ")"         ; for key references

ReferenceValue      = [ InstanceCollectionPath "/" ] KeyList
                                   ; of the referenced instance
```

Where:

- `KeyList` is `OrdinaryKeyList` for instances of ordinary classes and `AssociationKeyList` for instances of association classes.
- The key value lists in `OrdinaryKeyList` and `AssociationKeyList` are ordered by their property names, using binary ordering based on the data octets of the UTF-8 representation of the normalized Unicode characters of the property name.
- `OrdinaryKeyPropertyValue` is the percent-encoded string representation of the value of an ordinary key property. (See 7.1.2 for percent-encoding rules, and see DSP0004 for the string representations of CIM data types.)
- `ReferenceValue` uses the `KeyList` value of the referenced instance.

  NOTE: This is the same `KeyList` value that would be used in a resource identifier addressing that instance directly.

  NOTE: The use of parentheses around `ReferenceValue` in the `AssociationKeyValue` rule enables references to reference other association instances.

  NOTE: This version of the document does not yet define rules for the use of `InstanceCollectionPath` in `ReferenceValue`.

Examples:

- `/cimrs/namespaces/myns/classes/ACME_System/42,Dent%2CArthur`

  Assuming ACME_System exposes key properties Name (string) and Answer (uint32), this resource identifier addresses the ACME_System instance in the "myns" namespace that has a Name value of `"Dent,Arthur"` and an Answer value of 42.

  This example shows that the order of key values in the resource identifier is in the order of property names.

The comma in the Name value has been percent-encoded, as defined in 7.1.2.

- `/cimrs/namespaces/myns/classes/ACME_LogicalDevice/(babel%2Dfish),(42,Dent%2CArthur)`

  Assuming the ACME_LogicalDevice association class exposes key references System (REF ACME_System) and Device (REF ACME_Device), where ACME_System is defined as in the previous example and ACME_Device exposes a key property Name (string), this resource identifier addresses the ACME_LogicalDevice instance in the "myns" namespace that associates the ACME_System instance with key values 42 and `"Dent,Arthur"` (see previous example) and the ACME_Device instance with a Name value of `"babel-fish"`.

  This example shows that the order of key values in the resource identifier is in the order of property names, at each level of the referencing hierarchy.

  The hash character in the Name value the ACME_Device instance has been percent-encoded, as defined in 7.1.2.

B.1.10 Instance associator collection resource

A WBEM resource identifier addressing an instance associator collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = InstanceAssociatorCollectionPath

InstanceAssociatorCollectionPath = InstancePath "/associators"
```

Example:

- `/cimrs/namespaces/myns/classes/CIM_ManagedElement/instances/acme,1/associators`

  This resource identifier addresses the collection of all instances associated with the "CIM_ManagedElement" instance in the "myns" namespace that is identified by the key list "acme,1"

B.1.11 Instance reference collection resource

A WBEM resource identifier addressing an instance reference collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = InstanceReferenceCollectionPath

InstanceReferenceCollectionPath = InstancePath "/references"
```

Example:

- `/cimrs/namespaces/myns/classes/CIM_ManagedElement/instances/acme,1/references`

  This resource identifier addresses the collection of all instances referencing the "CIM_ManagedElement" instance in the "myns" namespace that is identified by the key list "acme,1".

B.1.12 Instance method invocation resource

A WBEM resource identifier addressing an instance method invocation resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = InstanceMethodInvocationPath

InstanceMethodInvocationPath = InstancePath "/methodinvocation"
```

Example:

- `/cimrs/namespaces/myns/classes/CIM_Example/instances/acme,1/methodinvocation`

  This resource identifier addresses the method invocation point of the "CIM_Example" instance in the "myns" namespace that is identified by the key list "acme,1".

B.1.13 Qualifier type collection resource

A WBEM resource identifier addressing a qualifier type collection resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = QualifierTypeCollectionPath

QualifierTypeCollectionPath = NamespacePath "/qualifiers"
```

Example:

- `/cimrs/namespaces/myns/qualifiers`

  This resource identifier addresses the collection of all CIM qualifier types in the "myns" namespace.

B.1.14 Qualifier type resource

A WBEM resource identifier addressing a qualifier type resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = QualifierTypePath

QualifierTypePath = QualifierTypeCollectionPath "/" QualifierName
```

Where:

- `QualifierName` is the percent-encoded name of the qualifier. (See 7.1.2 for percent-encoding rules.)

Example:

- `/cimrs/namespaces/myns/qualifiers/Description`

  This resource identifier addresses the "Description" qualifier type in the "myns" namespace.

B.1.15 Instance query resource

A WBEM resource identifier addressing an instance query resource could conform to the WBEM-resource-identifier ABNF rule (see 7.1.1) with the following additional rules:

```
path-absolute = InstanceQueryPath

InstanceQueryPath = NamespacePath "/instancequery"
```

Example:

- `/cimrs/namespaces/myns/instancequery`

  This resource identifier addresses the instance query resource of the "myns" namespace.

B.2 Example using WBEM URIs

Because the structure of WBEM resource identifiers for non-top-level resources is not mandated in this document, WBEM URIs (defined in DSP0207) could be used as a basis. However, the addressing abilities of WBEM URIs would need to be extended to cover all resource types defined in this document.

The structure of the top-level resource (that is, the namespace collection resource) still needs to conform to the structure defined in 7.2.1.

ANNEX C
(informative)

Change log

Version	Date	Description
1.0.0a	2010-10-11	Released as Work in Progress (of an Informational Specification)
1.0.0	2011-04-15	Released as DMTF Informational Specification

Bibliography

This annex contains a list of non-normative references for this document.

DMTF DSP0200, *CIM Operations over HTTP 1.3*,
http://www.dmtf.org/standards/published_documents/DSP0200_1.3.pdf

DMTF DSP0202, *CIM Query Language Specification 1.0*,
http://www.dmtf.org/standards/published_documents/DSP0202_1.0.pdf

DMTF DSP0207, *WBEM URI Mapping 1.0*,
http://www.dmtf.org/standards/published_documents/DSP0207_1.0.pdf

DMTF DSP0230, *WS-CIM Mapping Specification 1.0*,
http://www.dmtf.org/standards/published_documents/DSP0230_1.0.pdf

DMTF DSP1054, *Indications Profile 1.1*,
http://www.dmtf.org/sites/default/files/standards/documents/DSP1054_1.1.pdf

DMTF DSP-IS0202, *CIM-RS Binding to JSON 1.0.0a (Work in Progress)*,
http://www.dmtf.org/standards/published_documents/DSP-IS0202_1.0.0a.pdf

ITU-T X.509, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks*,
http://www.itu.int/rec/T-REC-X.509/en

R. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, PhD thesis, University of California, Irvine, 2000,
http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

R. Fielding, *REST APIs must be hypertext driven*, October 2008,
http://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven

J. Holzer, *RESTful Web Services and JSON for WBEM Operations*, Master thesis, University of Applied Sciences, Konstanz, Germany, June 2009,
http://mond.htwg-konstanz.de/Abschlussarbeiten/Details.aspx?id=1120

A. Manes, *Rest principle: Separation of representation and resource*, March 2009,
http://apsblog.burtongroup.com/2009/03/rest-principle-separation-of-representation-and-resource.html

L. Richardson and S. Ruby, *RESTful Web Services*, May 2007, O'Reilly, ISBN 978-0-596-52926-0,
http://www.oreilly.de/catalog/9780596529260/