



1
2
3
4
5

Document Number: DSP0211

Date: 2013-01-24

Version: 1.0.0

6 **CIM-RS Payload Representation in JSON**

7 **Document Type: Specification**

8 **Document Status: DMTF Standard**

9 **Document Language: en-US**

10

11 Copyright Notice

12 Copyright © 2010-2013 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

33

CONTENTS

34	Foreword	4
35	Introduction.....	5
36	Document conventions.....	5
37	1 Scope	7
38	2 Normative references.....	7
39	3 Terms and definitions	8
40	4 Symbols and abbreviated terms.....	10
41	5 Conformance.....	11
42	6 CIM-RS payload representation in JSON	11
43	6.1 Overview	11
44	6.2 General requirements	11
45	6.2.1 Conformance to the JSON grammar	11
46	6.2.2 Whitespace	11
47	6.2.3 Character repertoire, representation, encoding and escaping	12
48	6.2.4 Version of the payload representation in JSON	12
49	6.2.5 Internet media type	12
50	6.2.6 Representation of CIM-RS abstract datatypes	13
51	6.2.7 Representation of CIM element values.....	13
52	6.2.8 Representation of CIM real32 and real64 datatypes	15
53	6.2.9 Representation of CIM reference datatypes	15
54	6.2.10 Representation of CIM Null values	16
55	6.2.11 Representation of method invocation links	16
56	6.3 Representation of protocol payload elements	16
57	6.3.1 Format of payload element descriptions.....	17
58	6.3.2 ServerEntryPoint payload element	18
59	6.3.3 ListenerEntryPoint payload element	19
60	6.3.4 Instance payload element.....	20
61	6.3.5 InstanceCollection payload element	21
62	6.3.6 MethodInvocationRequest payload element	22
63	6.3.7 MethodInvocationResponse payload element.....	23
64	6.3.8 IndicationDeliveryRequest payload element	24
65	6.3.9 ErrorResponse payload element	24
66	ANNEX A (informative) Change log	26
67	Bibliography	27
68		

69 Tables

70	Table 1 – Representation of CIM-RS abstract datatypes in JSON.....	13
71	Table 2 – Representation of CIM datatypes in JSON	13
72	Table 3 – CIM-RS payload elements	16
73		

74

Foreword

75 The *CIM-RS Payload Representation in JSON* (DSP0211) specification was prepared by the DMTF CIM-
76 RS Working Group, based on work of the DMTF CIM-RS Incubator.

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

79 Acknowledgments

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

- 81 • Cornelia Davis, EMC
- 82 • George Ericson, EMC
- 83 • Johannes Holzer, IBM
- 84 • Robert Kieninger, IBM
- 85 • Wojtek Kozaczynski, Microsoft
- 86 • Larry Lamers, VMware
- 87 • Andreas Maier, IBM (editor)
- 88 • Bob Tillman, EMC
- 89 • Marvin Waschke, CA Technologies

90

Introduction

91 The information in this document should be sufficient to unambiguously identify the representation of the
92 payload elements defined in [DSP0210](#), in JSON (JavaScript Object Notation).

93 The target audience for this specification is typically implementers who are writing WBEM servers, clients,
94 or listeners supporting the CIM-RS protocol with a payload representation in JSON.

95 Document conventions

96 Typographical conventions

97 The following typographical conventions are used in this document:

- 98 • Document titles are marked in *italics*.
- 99 • ABNF rules and JSON text are in `monospaced font`.

100 ABNF usage conventions

101 Format definitions in this document are specified using ABNF (see [RFC5234](#)), with the following
102 deviations:

- 103 • Literal strings are to be interpreted as case-sensitive UCS characters, as opposed to the
104 definition in [RFC5234](#) that interprets literal strings as case-insensitive US-ASCII characters.
105

107

CIM-RS Payload Representation in JSON

1 Scope

109 This specification is a payload representation specification for the CIM-RS protocol defined in [DSP0210](#),
110 describing a representation of CIM-RS payload elements in JSON (JavaScript Object Notation, see
111 [ECMA-262](#)).

112 Specifically, it describes how the abstract payload elements defined in [DSP0210](#) are represented in
113 JSON and how a JSON representation of these payload elements is identified using an Internet media
114 type.

115 Background information for CIM-RS is described in a white paper, [DSP2032](#).

2 Normative references

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

121 ANSI/IEEE 754-1985, *IEEE Standard for Binary Floating-Point Arithmetic*, August 1985,
122 http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=30711

123 DMTF DSP0004, *CIM Infrastructure Specification 2.7*,
124 http://www.dmtf.org/standards/published_documents/DSP0004_2.7.pdf

125 DMTF DSP0210, *CIM-RS Protocol 1.0*,
126 http://www.dmtf.org/standards/published_documents/DSP0210_1.0.pdf

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

129 ECMA-262, *ECMAScript Language Specification, Edition 5.1*, June 2011,
130 <http://www.ecma-international.org/publications/standards/Ecma-262.htm>

131 IETF RFC4627 (Informational), *The application/json Media Type for JavaScript Object Notation (JSON)*,
132 July 2006,
133 <http://tools.ietf.org/html/rfc4627>

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

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

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

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

144 3 Terms and definitions

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

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

153 The terms "clause", "subclause", "paragraph", and "annex" in this document are to be interpreted as
154 described in [ISO/IEC Directives, Part 2](#), Clause 5.

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

158 The terms defined in [DSP0004](#), [DSP0223](#), and [DSP0210](#) apply to this document. Specifically, this
159 document uses the terms "namespace", "qualifier", "qualifier type", "class", "creation class", "ordinary
160 class", "association", "indication", "instance", "property", "ordinary property", "reference", "method",
161 "parameter", and "return value" defined in [DSP0004](#).

162 This document does not define additional terms; some terms defined in these documents are repeated for
163 convenience.

164 3.1

165 CIM-RS payload element

166 a particular type of content of the entity body of the HTTP messages used by the CIM-RS protocol.
167 Payload elements are abstractly defined in [DSP0210](#), and concretely in CIM-RS payload representation
168 specifications, such as this document.

169 3.2

170 CIM-RS payload representation

171 an encoding format that defines how the abstract payload elements defined in [DSP0210](#) are encoded in
172 the entity body of the HTTP messages used by the CIM-RS protocol. This includes resource
173 representations.

174 3.3

175 CIM-RS payload representation specification

176 a specification that defines a CIM-RS payload representation, such as this document.

177 3.4

178 CIM-RS protocol

179 the RESTful protocol defined in [DSP0210](#), for which this document describes a payload representation in
180 JSON.

181 3.5

182 CIM-RS resource

183 an entity in a WBEM server or WBEM listener that can be referenced using a CIM-RS resource identifier
184 and thus can be the target of an HTTP method in the CIM-RS protocol. Also called "resource" in this
185 document.

186 3.6**187 CIM-RS resource identifier**

188 a URI that is a reference to a CIM-RS resource in a WBEM server or WBEM listener, as defined in
189 [DSP0210](#). Also called "resource identifier" in this document.

190 3.7**191 Internet media type**

192 a string identification for representation formats in Internet protocols. Originally defined for email
193 attachments and termed "MIME type". Because the CIM-RS protocol is based on HTTP, it uses the
194 definition of media types from section 3.7 of [RFC2616](#).

195 3.8**196 Normalization Form C**

197 a normalization form for UCS characters that avoids the use of combining marks where possible and that
198 allows comparing UCS character strings on a per-code-point basis. It is defined in [The Unicode Standard,](#)
199 [Annex #15](#).

200 3.9**201 resource representation**

202 a representation of a resource or some aspect thereof, in some format. A particular resource may have
203 any number of representations. The format of a resource representation is identified by a media type. In
204 the CIM-RS protocol, the more general term "payload representation" is used, because not all payload
205 elements are resource representations.

206 3.10**207 UCS character**

208 a character from the Universal Character Set defined in [ISO/IEC 10646:2003](#). See [DSP0004](#) for the
209 usage of UCS characters in CIM strings. An alternative term is "Unicode character".

210 3.11**211 UCS code position**

212 a numeric identification for a UCS character in the range of 0x0 to 0x10FFFF, as defined in [ISO/IEC](#)
213 [10646:2003](#).

214 3.12**215 WBEM client**

216 the client role in the CIM-RS protocol and in other WBEM protocols. For a full definition, see [DSP0210](#).

217 3.13**218 WBEM listener**

219 the event listener role in the CIM-RS protocol and in other WBEM protocols. For a full definition, see
220 [DSP0210](#).

221 3.14**222 WBEM server**

223 the server role in the CIM-RS protocol and in other WBEM protocols. For a full definition, see [DSP0210](#).

224 **4 Symbols and abbreviated terms**

225 The abbreviations defined in [DSP0004](#), [DSP0223](#), and [DSP0210](#) apply to this document. The following
226 additional abbreviations are used in this document.

227 **4.1**

228 **ABNF**

229 Augmented Backus-Naur Form, as defined in [RFC5234](#).

230 **4.2**

231 **CIM**

232 Common Information Model, as defined by DMTF.

233 **4.3**

234 **CIM-RS**

235 **CIM RESTful Services**

236 The RESTful protocol for CIM defined in this document and related documents.

237 **4.4**

238 **ECMAScript**

239 a scripting language that is the standard version of what was called JavaScript. It is defined in [ECMA-
240 262](#).

241 **4.5**

242 **IANA**

243 Internet Assigned Numbers Authority; see <http://www.iana.org>.

244 **4.6**

245 **JSON**

246 JavaScript Object Notation, as defined in [ECMA-262](#).

247 **4.7**

248 **REST**

249 Representational State Transfer, as originally and informally described in Architectural Styles and the
250 Design of Network-based Software Architectures.

251 **4.8**

252 **UCS**

253 Universal Character Set, as defined in [ISO/IEC 10646:2003](#).

254 **4.9**

255 **URI**

256 Uniform Resource Identifier, as defined in [RFC3986](#).

257 **4.10**

258 **UTF-8**

259 UCS Transformation Format 8, as defined in [ISO/IEC 10646:2003](#).

260 **4.11**

261 **WBEM**

262 Web Based Enterprise Management, as defined by DMTF.

263 5 Conformance

264 A representation of CIM-RS payload elements in JSON conforms to this document only if it conforms to
265 all normative rules stated in this document.

266 The term "CIM-RS representation in JSON" shall be used only for representations of CIM-RS payload
267 elements in JSON that conform to this document.

268 6 CIM-RS payload representation in JSON

269 This clause defines the representation of the CIM-RS payload in JSON.

270 The JSON grammar is defined in clause 15.12.1 ("The JSON Grammar") of [ECMA-262](#). Care should be
271 taken to distinguish text in [ECMA-262](#) that applies to the JSON grammar from text that applies to the
272 ECMAScript (formerly: JavaScript) language. However, text in [ECMA-262](#) outside of its clause 15.12.1
273 but referenced from within that clause applies unless otherwise noted in this document.

274 Note that although [RFC4627](#) defines the grammar of the JSON language consistently with clause 15.12.1
275 of [ECMA-262](#), [RFC4627](#) is an informational RFC whose purpose is to describe the Internet media type for
276 JSON but not to be the normative definition of the JSON grammar. For this reason, this document
277 references [ECMA-262](#) as the normative definition of the JSON grammar, but yet references [RFC4627](#)
278 where needed.

279 6.1 Overview

280 This subclause describes informally and at a high level how the CIM-RS payload elements defined in
281 [DSP0210](#) are represented in JSON.

282 CIM-RS payload elements are represented as JSON objects. The attributes of these JSON objects match
283 the properties of the payload elements 1:1, in name, datatype and meaning. Nested elements in these
284 payload elements are represented as nested JSON objects. Arrays in these payload elements are
285 represented as JSON arrays. For details, see 6.3.

286 The Internet media type identifying the JSON representation of CIM-RS is the standard media type
287 registered for JSON at IANA (application/json). For details, see 6.2.5.

288 Defining a new media type specific for the CIM-RS representation of JSON was considered and
289 dismissed, because the value of using a well-known and broadly supported standard media type was
290 deemed higher than the advantage of being able to distinguish JSON for representing CIM-RS from
291 general JSON, or multiple flavors of JSON for representing CIM-RS from each other. Multiple
292 incompatible flavors of JSON representations of CIM-RS can be distinguished with the same media type
293 by using different major version numbers in the version parameter of the media type.

294 6.2 General requirements

295 6.2.1 Conformance to the JSON grammar

296 CIM-RS payload elements represented in JSON shall conform to the grammar defined by the symbol
297 *JSONText* defined in clause 15.12.1 ("The JSON Grammar") of [ECMA-262](#).

298 6.2.2 Whitespace

299 [ECMA-262](#) defines what the set of whitespace characters for JSON is (different from the set of
300 whitespace characters for ECMAScript), but it does not explicitly state whether the whitespace usage
301 rules for ECMAScript also apply to JSON.

302 CIM-RS payload elements represented in JSON shall conform to the rules for whitespace as defined in
303 subclause 7.2 (White Space) of [ECMA-262](#).

304 **6.2.3 Character repertoire, representation, encoding and escaping**

305 The JSON grammar defined in clause 15.12.1 of [ECMA-262](#) references the *SourceCharacter* symbol
306 defined in its clause 6 as the basis for the characters of its grammar be it for identifiers, delimiters or
307 values. The definition of the *SourceCharacter* symbol applies to the ECMAScript use of JSON and uses
308 the character repertoire of Unicode V3, requires a representation of UCS characters in Normalization
309 Form C, and effectively implies a requirement for an encoding in UTF-16 (or one of its little endian and big
310 endian derivatives).

311 The following rules apply to a use of the *SourceCharacter* symbol for the representation of CIM-RS
312 payload elements in JSON:

- 313 1) The character repertoire of *SourceCharacter* shall be that defined for values of the CIM string
314 type (defined in [DSP0004](#)). Note that this character repertoire is larger than the character
315 repertoire defined by [ECMA-262](#).
- 316 2) *SourceCharacter* shall be represented in Normalization Form C.
- 317 3) *SourceCharacter* shall be encoded in UTF-8. As a consequence, the entire payload element will
318 be encoded in UTF-8, and that character encoding is therefore not being indicated in the CIM-
319 RS payload or in any HTTP header fields.

320 [ECMA-262](#) defines backslash-based escaping for the representation of UCS characters, using their UCS
321 code positions. However, in the definition of the *UnicodeEscapeSequence* symbol in its clause 7.8.4
322 ("String Literals"), [ECMA-262](#) limits the representation of UCS code positions to a value range of four hex
323 digits. This is not sufficient for representing the character repertoire defined for values of the CIM string
324 type (it is also not sufficient for representing the character repertoire used by [ECMA-262](#) itself).

325 Therefore, the representation of CIM-RS payload elements in JSON shall support the following extended
326 definition of the *UnicodeEscapeSequence* symbol:

```
327 UnicodeEscapeSequence ::  
328     u HexDigit HexDigit HexDigit HexDigit  
329     u HexDigit HexDigit HexDigit HexDigit HexDigit  
330     u HexDigit HexDigit HexDigit HexDigit HexDigit HexDigit
```

331 NOTE: This extended definition is consistent with the four-to-six-digit form of the short identifier for UCS characters
332 defined in clause 6.5 of [ISO/IEC 10646:2003](#) (for example, U+000A, U+12345, and U+10FFFF).

333 [ECMA-262](#) defines backslash-based escaping for a number of popular characters, e.g. "\n". It states their
334 escape sequences, but it misses to define what they stand for. [RFC4627](#) does define both the escape
335 sequences and what they stand for.

336 **6.2.4 Version of the payload representation in JSON**

337 [DSP0210](#) requires that CIM-RS payload representation specifications define a version for the payload
338 representations they define.

339 The full version (m.n.u) of this document, without any draft levels, shall be used to identify the full version
340 of the JSON payload representation.

341 **6.2.5 Internet media type**

342 [DSP0210](#) requires that CIM-RS payload representation specifications define a unique Internet media type
343 that identifies the representation.

344 Only the standard media type for JSON defined in [RFC4627](#) ("application/json") shall be used to identify
 345 the representation of CIM-RS payload elements in JSON defined in this document. This media type is
 346 registered with IANA (see [IANA MIME Media Types](#)).

347 Note that [DSP0210](#) defines requirements for specifying parameters on media types that identify the
 348 representation of CIM-RS payload elements. One example for such a parameter is "version", specifying
 349 the version of the payload representation.

350 Therefore, the media type identifying version 1.0.0 of the JSON representation would be:

```
351 application/json; version=1.0.0
```

352 **6.2.6 Representation of CIM-RS abstract datatypes**

353 This subclause defines how values of the abstract datatypes used in [DSP0210](#) for the definition of the
 354 attributes of the abstract payload elements are represented in JSON.

355 Table 1 lists the abstract datatypes and their mapping to JSON datatypes.

356 **Table 1 – Representation of CIM-RS abstract datatypes in JSON**

Abstract Datatype	JSON Datatype	Additional Rules
String	string	See 6.2.3 for requirements on escaping and encoding
Integer	number	
ElementValue	object member	Values of CIM elements (that is, properties, parameters, return values) shall be represented in JSON as described in 6.2.7
MethodLink	object member	Method invocation links shall be represented in JSON as described in 6.2.11
URI	string	The string value shall be the CIM-RS resource identifier of the referenced resource in any valid format (see DSP0210).
Instance	object	See 6.3.4

357 **6.2.7 Representation of CIM element values**

358 Values of CIM elements (that is, properties, parameters, return values) shall be represented in JSON as
 359 follows:

360 The element value is represented as a JSON object member, where the name of the object member is
 361 the name of the CIM element; and the value of the JSON object member is a representation of the CIM
 362 element value as defined in Table 2, using the indicated JSON datatype.

363 The CIM datatype of the element (that is, the "type" child attribute of the ElementValue datatype) is
 364 intentionally not represented, for simplicity. It is expected that the JSON representation of CIM-RS is used
 365 by environments with simple and possibly dynamic type systems (such as JavaScript or Python), without
 366 a need to represent the elements using strong typing based on the CIM datatypes.

367 **Table 2 – Representation of CIM datatypes in JSON**

CIM Datatype	JSON Datatype	Additional Rules
boolean	boolean	Note that JSON is case sensitive w.r.t. the literals true and false
string	string	See 6.2.3 for requirements on escaping and encoding

CIM Datatype	JSON Datatype	Additional Rules
char16	string	See 6.2.3 for requirements on escaping and encoding
string, with OctetString qualifier	string	Shall be represented as if it was a normal CIM string-typed value (that is, without the OctetString qualifier)
uint8[], with OctetString qualifier	number array	Shall be represented as if it was a normal uint8-array-typed value (that is, without the OctetString qualifier)
string, with EmbeddedInstance qualifier	object	The embedded instance shall be represented as a JSON object as defined in 6.3.4. Its <i>class</i> attribute shall be the name of the creation class of the embedded instance. Note that the creation class may differ from the class specified in the value of the EmbeddedInstance qualifier.
string, with EmbeddedObject qualifier	object or null value	If the embedded object is an instance, it shall be represented as a JSON object as defined in 6.3.4. Its <i>class</i> attribute shall be the name of the creation class of the embedded instance. If the embedded object is a class, it shall not be represented and instead the JSON null value shall be represented.
datetime	string	The string value shall be the 25-character datetime string defined in DSP0004
uint8,16,32,64	number	
sint8,16,32,64	number	
real32,64	number or string	See 6.2.8
<classname> ref	string	See 6.2.9
array of any CIM type	array of corresponding JSON type	The type string shall reflect the type of the array entries

368 Examples for representing named CIM elements (that is, properties or parameters) of these datatypes:

```

369 . . .
370   "ABoolean": true,
371   "AString": "some text",
372   "AChar16": "Z",
373   "AnOctetstringViaString": "0x00000007616263",
374   "AnOctetstringViaUint8Array": [ 0, 0, 0, 7, 0x61, 0x62, 0x63 ],
375   "AnEmbeddedInstance": {
376     "kind": "instance",
377     "class": "CIM_ComputerSystem",
378     "properties": {
379       "InstanceID": "sys:1",
380       "ElementName": "system #1" }
381   },
382   "AnEmbeddedObjectThatIsAnInstance": {
383     "kind": "instance",
384     "class": "CIM_ComputerSystem",
385     "properties": {
386       "InstanceID": "sys:1",
387       "ElementName": "system #1" }
388   },

```

```

389     "ADatetime": "20120213175830.123456+060",
390     "AUint16": 20000,
391     "ASint16": -16000,
392     "AReal32": 3.1415927,
393     "ARef": "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys:1",
394     "ABooleanArray": [ true, false, true ],
395     "AStringArray": [ "some text", null, "more text\n" ],
396     "AReal64Array": [ 1E-42, NaN, "-Infinity" ],
397     . . .

```

398 6.2.8 Representation of CIM real32 and real64 datatypes

399 The CIM real32 and real64 types are based on the [IEEE 754](#) Single and Double formats (see [DSP0004](#));
400 values of these types shall be represented in JSON as follows, depending on their value:

- 401 • the IEEE special values *positive infinity*, *negative infinity*, and *not-a-number* shall be
402 represented as JSON string-typed values using the following strings:

403 positive infinity: "Infinity"

404 negative infinity: "-Infinity"

405 not-a-number: "NaN"

- 406 • any other values are normal floating point numbers and shall be represented as JSON number-
407 typed values, using a precision for the significand of at least 9 decimal digits for real32 and at
408 least 17 digits for real64.

409 NOTE: The strings used for representing the IEEE special values are consistent with Python's
410 serialization of float-typed values in JSON, and with Java's serialization of float-typed values as strings.
411 These strings are not consistent with the representation of the special values in the XML datatypes
412 xs:float and xs:double.

413 NOTE: JSON numbers only support lexical notations with a basis of 10 (e.g. 4.56E-3). The value space of
414 CIM real32 and real64 typed values is defined by the [IEEE 754](#) Single and Double formats, which have a
415 basis of 2. The definition of a minimum precision for the significand guarantees that the value of CIM real
416 types does not change when converting it back and forth between the (10-based) JSON representation
417 and a (2-based) internal representation (see subclause 5.6 in [IEEE 754](#)).

418 Examples:

```

419     . . .
420     "Throughput": 3.45E3,
421     "ErrorRate": "NaN",
422     . . .

```

423 6.2.9 Representation of CIM reference datatypes

424 Values of CIM reference-typed elements (that is, declared as <classname> ref) shall be represented in
425 JSON such that the JSON value is the CIM-RS resource identifier of the referenced instance in any valid
426 format defined in [DSP0210](#).

427 The class declared in the reference is not represented, again for simplicity of the JSON representation.
428

429 Example for a reference property named System that is declared as type "CIM_ComputerSystem ref":

```
430 . . .
431 "System": "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys:1"
432 . . .
```

433 **6.2.10 Representation of CIM Null values**

434 CIM Null values shall be represented using the JSON literal *null*.

435 Note that the JSON literal null is case sensitive.

436 Example:

```
437 . . .
438 "ElementName": null,
439 "PossibleStates": [1, null, 3],
440 . . .
```

441 **6.2.11 Representation of method invocation links**

442 Method invocation links shall be represented in JSON as a JSON object member, where the name of the
 443 object member is the name of the CIM method (without any parenthesis or parameters); and the value of
 444 the JSON object member is a JSON String typed value that is the resource identifier of the method
 445 invocation resource.

446 Example:

```
447 . . .
448 "RequestStateChange":
449 "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys:1/RequestStateChange"
450 . . .
```

451 **6.3 Representation of protocol payload elements**

452 This subclause defines how the CIM-RS payload elements defined in [DSP0210](#) are represented in JSON.

453 Table 3 lists the payload elements defined in [DSP0210](#).

454 **Table 3 – CIM-RS payload elements**

Payload Element	Meaning	Description
ServerEntryPoint	representation of the server entry point resource of a CIM-RS server, describing protocol-level capabilities of the server, and providing resource identifiers for performing global operations	See 6.3.2
ListenerEntryPoint	representation of the listener entry point resource of a CIM-RS listener, describing protocol-level capabilities of the listener	See 6.3.3
Instance	representation of a CIM instance; that is, a CIM-modeled resource, representing an aspect of a managed object in the managed environment	See 6.3.4
InstanceCollection	representation of a set of CIM instances in an instance collection	See 6.3.5
MethodRequest	the data used to request the invocation of a method	See 6.3.6
MethodResponse	the data used in the response of the invocation of a method	See 6.3.7

Payload Element	Meaning	Description
IndicationDeliveryRequest	the data used to request the delivery of an indication to a listener destination	See 6.3.8
ErrorResponse	the data used in an error response to any request	See 6.3.9

455 6.3.1 Format of payload element descriptions

456 The descriptions in the following subclauses use a lightweight approach for defining the JSON structure
457 for the various payload elements. The following example illustrates this description approach:

```
458 {
459     "kind": "instance",
460     "self": (value),
461     "class": (value),
462     "properties": {
463         (value): (value)#
464     },?
465     "methods": {
466         (value): (value)#
467     }?
468 }
```

469 All text in such a description is to be understood literally as stated, except for whitespace characters used
470 before and after JSON tokens (see 6.2.2), and except for the following special indicators:

471 **#** indicates that the JSON object member or JSON array element to the left of the **#** may be
472 present zero or more times in a comma-separated list.

473 **?** indicates that the JSON object member or JSON array element to the left of the **?** may be
474 present or absent.

475 **(value)** is replaced with a JSON value, according to the description in the respective subclause.
476 The literal inside of the parenthesis is typically more specific than "value".

477 Note that the rules on using **#** and **?** are not precise w.r.t. the presence of commas delimiting JSON
478 object members or JSON array elements. However, the presence of commas results from the general
479 JSON syntax rules; that is, exactly one comma is required between members or elements, and no trailing
480 comma is permitted after the last member or element.

481 An example for a valid payload element conforming to the example description above would be:

```
482 {
483     "kind": "instance",
484     "self": "/cimrs/root%2Fcimv2/CIM_RegisteredProfile/DMTF%3AFan%3A1.1.0",
485     "class": "CIM_RegisteredProfile",
486     "properties": {
487         "InstanceID": "DMTF:Fan:1.1.0",
488         "RegisteredName": "Fan",,
489         "RegisteredOrganization": 2,
490         "RegisteredVersion": "1.1.0"
491     }
492 }
```

493 6.3.2 ServerEntryPoint payload element

494 ServerEntryPoint payload elements (see [DSP0210](#)) shall be represented using the following JSON
495 structure:

```

496 {
497   "kind": "serverentrypoint"
498   "self": (self),
499   "namespaces": [
500     { "name": (name),
501       "enumeration": (enumeration),
502       "creation": (creation),
503       "staticmethods": [
504         (static-method-name): (static-method-uri)#
505       ],?
506       "protocolversions": [
507         (protocolversion)#
508       ],?
509       "contenttypes": [
510         (contenttype)#
511       ]?
512     }#
513   ],?
514   "entitytagging": (entitytagging),
515   "pagedretrieval": (pagedretrieval)
516 }
```

517 Where:

- 518 • (*self*), (*enumeration*), (*creation*), (*entitytagging*), and (*pagedretrieval*) are the values of the
519 like-named attributes of the represented ServerEntryPoint payload element, using the
520 representation defined in 6.2.6.
- 521 • (*static-method-name/uri*), (*name*), (*protocolversion*), and (*contenttype*) are single entries in
522 the respective array attributes, using the representations defined in 6.2.6.
523 If one of these arrays in the JSON representation has no entries, the corresponding JSON
524 object member should not be present (but may be present with a value of an empty JSON
525 array).

526 Example:

```

527 {
528   "kind": "serverentrypoint",
529   "self": "/cimrs",
530   "namespaces": [
531     { "name": "interop",
532       "enumeration": "/cimrs/interop/enum",
533       "creation": "/cimrs/interop/create",
534       "staticmethods": [ MyStatic: "/cimrs/interop/mystatic" ],
535       "protocolversions": [ "1.0.0", "1.0.1" ],
536       "contenttypes": [
537         "application/json;version=1.0.0",
538         "application/json;version=1.0.1" ]
539     }
540   ]
541 }
```

```

539     },
540     { "name": "root/cimv2",
541       "enumeration": "/cimrs/root%2Fcimv2/enum",
542       "creation": "/cimrs/root%2Fcimv2/create",
543       "staticmethods": [ MyStatic": "/cimrs/root%2Fcimv2/mystatic" ],
544       "protocolversions": [ "1.0.0", "1.0.1" ],
545       "contenttypes": [
546         "application/json;version=1.0.0",
547         "application/json;version=1.0.1" ]
548     }
549 ],
550 "entitytagging": true,
551 "pagedretrieval": true
552 }

```

553 6.3.3 ListenerEntryPoint payload element

554 ListenerEntryPoint payload elements (see [DSP0210](#)) shall be represented using the following JSON
555 structure:

```

556 {
557   "kind": "listenerentrypoint"
558   "self": (self),
559   "destinations": [
560     (destination)#
561   ],?
562   "protocolversions": [
563     (protocolversion)#
564   ],?
565   "contenttypes": [
566     (contenttype)#
567   ]?
568 }

```

569 Where:

- 570 • (**self**) is the value of the like-named attribute of the represented ListenerEntryPoint payload
571 element, using the representation defined in 6.2.6.
- 572 • (**destination**), (**protocolversion**), and (**contenttype**) are single entries in the array attributes
573 *destinations*, *protocolversions*, and *contenttypes*, respectively, of the represented
574 ListenerEntryPoint payload element, using the representations defined in 6.2.6.
575 If one of these arrays in the JSON representation has no entries, the corresponding JSON
576 object member should not be present (but may be present with a value of an empty JSON
577 array).

578 Example:

```

579 {
580   "kind": "listenerentrypoint",
581   "self": "/cimrs",
582   "destinations": [ "/cimrs/dest1", "/cimrs/dest2" ],
583   "protocolversions": [ "1.0.0" ],
584   "contenttypes": [

```

```
585     "application/json;version=1.0.0" ]
586 }
```

587 6.3.4 Instance payload element

588 Instance payload elements (see [DSP0210](#)) shall be represented using the following JSON structure:

```
589 {
590     "kind": "instance",
591     "self": (self),
592     "class": (class),
593     "properties": {
594         (property-name): (property-value)#
595     },?
596     "methods": {
597         (method-name): (method-uri)#
598     }?
599 }
```

600 Where:

- 601 • (**self**) and (**class**) are the values of the like-named attributes of the represented Instance
602 payload element, using the representation defined in 6.2.6.
- 603 • Each member of *properties* represents an entry in the *properties* array attribute of the
604 represented Instance payload element; that is, a property of the represented instance.
605 (**property-name**) is the property name; (**property-value**) is the property value represented as
606 defined in 6.2.7.
607 If the *properties* array of the represented Instance payload element has no entries, the
608 *properties* JSON object member should not be present (but may be present with a value of an
609 empty JSON object).
- 610 • Each member of *methods* represents an entry in the *methods* array attribute of the represented
611 Instance payload element; that is, a method invocation link of the represented instance.
612 (**method-name**) is the method name; (**method-uri**) is the resource identifier represented as
613 defined in 6.2.6 for abstract datatype URI.
614 If the *methods* array of the represented Instance payload element has no entries, the *methods*
615 JSON object member should not be present (but may be present with a value of an empty
616 JSON object).
617

618 Example:

```

619 {
620   "kind": "instance",
621   "self": "/cimrs/root%2Fcimv2/CIM_RegisteredProfile/DMTF%3AFan%3A1.1.0",
622   "class": "CIM_RegisteredProfile",
623   "properties": {
624     "InstanceID": "DMTF:Fan:1.1.0",
625     "RegisteredName": "Fan",
626     "RegisteredOrganization": 2,
627     "RegisteredVersion": "1.1.0" }
628   "methods": {
629     "GetCentralInstanceNames": "/cimrs/root%2Fcimv2/CIM_RegisteredProfile/DMTF%3AFa
630     n%3A1.1.0/GetCentralInstanceNames" }
631 }

```

632 6.3.5 InstanceCollection payload element

633 InstanceCollection payload elements (see [DSP0210](#)) shall be represented using the following JSON
634 structure:

```

635 {
636   "kind": "instancecollection",
637   "self": (self),
638   "next": (next), ?
639   "class": (class),
640   "instances": [
641     (instance)#
642   ]?
643 }

```

644 Where:

- 645 • (**self**), (**next**) and (**class**) are the values of the like-named attributes of the represented
646 InstanceCollection payload element, using the representation defined in 6.2.6.
- 647 • Each array entry of *instances* represents an entry in the *instances* array attribute of the
648 represented InstanceCollection payload element; that is, an instance of the represented
649 instance collection. (**instance**) is a representation of the instance as defined in 6.2.6 for abstract
650 datatype Instance.
651 If the array attribute *instances* of the represented InstanceCollection payload element has no
652 entries, the *instances* JSON object member should not be present (but may be present with a
653 value of an empty JSON array).

654 Example for an entire collection (that is, not in paged mode):

```

655 {
656   "kind": "instancecollection",
657   "self": "/cimrs/enum?namespace=root%2Fcimv2&class=CIM_System",
658   "class": "CIM_System",
659   "instances": [
660     {
661       "kind": "instance",
662       "self": "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys:1",

```

```

663     "class": "CIM_ComputerSystem",
664     "properties": {
665         "InstanceID": "sys:1",
666         "ElementName": "System #1" }
667     "methods": {
668         "RequestStateChange": "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys:1/Request
669 StateChange" }
670     },{
671     "kind": "instance",
672     "self": "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys1",
673     "class": "CIM_ComputerSystem",
674     "properties": {
675         "InstanceID": "sys:2",
676         "ElementName": null }
677     "methods": {
678         "RequestStateChange": "/cimrs/root%2Fcimv2/CIM_ComputerSystem/sys:2/Request
679 StateChange" }
680     } ]
681 }

```

682 NOTE: This example assumes that CIM_ComputerSystem is a subclass of CIM_System.

683 6.3.6 MethodInvocationRequest payload element

684 MethodInvocationRequest payload elements (see [DSP0210](#)) shall be represented using the following
685 JSON structure:

```

686 {
687     "kind": "methodrequest",
688     "self": (self),
689     "method": (method),
690     "parameters": {
691         (parameter-name): (parameter-value)#
692     }
693 }

```

694 Where:

- 695 • (***self***) and (***method***) are the values of the like-named attributes of the represented
696 MethodInvocationRequest payload element, using the representation defined in 6.2.6.
- 697 • Each member of *parameters* represents an entry in the *parameters* array attribute of the
698 represented MethodInvocationRequest payload element; that is, a parameter of the request.
699 (***parameter-name***) is the parameter name; (***parameter-value***) is the parameter value
700 represented as defined in 6.2.7.
701 If the *parameters* array of the represented MethodInvocationRequest payload element has no
702 entries, the *parameters* JSON object member should not be present (but may be present with a
703 value of an empty JSON object).

704 Example:

```

705 {
706     "kind": "methodrequest",
707     "self": "/cimrs/root%2Fcimv2/CIM_RegisteredProfile/DMTF%3AFan%3A1.1.0/GetCentralI
708 nstanceNames",

```

```

709     "method": "GetCentralInstanceNames",
710     "parameters": {
711         "MaxNumber": 100 }
712 }

```

713 6.3.7 MethodInvocationResponse payload element

714 MethodInvocationResponse payload elements (see [DSP0210](#)) shall be represented using the following
715 JSON structure:

```

716 {
717     "kind": "methodresponse",
718     "self": (self),
719     "method": (method),
720     "returnvalue": (return-value),
721     "parameters": {
722         (parameter-name): (parameter-value)#
723     }
724 }

```

725 Where:

- 726 • **(self)** and **(method)** are the values of the like-named attributes of the represented
727 MethodInvocationResponse payload element, using the representation defined in 6.2.6.
- 728 • *returnvalue* represents the *returnvalue* attribute of the represented MethodInvocationResponse
729 payload element; that is, the return value of the method. **(return-value)** is the return value
730 represented as defined in 6.2.7.
- 731 • Each member of *parameters* represents an entry in the *parameters* array attribute of the
732 represented MethodInvocationResponse payload element; that is, an output parameter of the
733 method. **(parameter-name)** is the parameter name; **(parameter-value)** is the parameter value
734 represented as defined in 6.2.7.
735 If the *parameters* array of the represented MethodInvocationResponse payload element has no
736 entries, the *parameters* JSON object member should not be present (but may be present with a
737 value of an empty JSON object).

738 Example:

```

739 {
740     "kind": "methodresponse",
741     "self": "/cimrs/root%2Fcimv2/CIM_RegisteredProfile/DMTF%3AFan%3A1.1.0/GetCentralI
742 nstanceNames",
743     "method": "GetCentralInstanceNames",
744     "returnvalue": 0,
745     "parameters": {
746         "CentralInstanceNames": [
747             "/cimrs/root%2Fcimv2/CIM_Fan/fan:1",
748             "/cimrs/root%2Fcimv2/CIM_Fan/fan:2" ]
749     }
750 }

```

751 6.3.8 IndicationDeliveryRequest payload element

752 IndicationDeliveryRequest payload elements (see [DSP0210](#)) shall be represented using the following
753 JSON structure:

```
754 {  
755   "kind": "indicationdeliveryrequest",  
756   "self": (self),  
757   "indication": (indication-instance)  
758 }
```

759 Where:

- 760 • (*self*) and (*indication*) are the values of the like-named attributes of the represented
761 IndicationDeliveryRequest payload element, using the representations defined in 6.2.6.
- 762 • (*indication-instance*) is the value of the attribute *indication* of the represented
763 IndicationDeliveryRequest payload element, using the representation for abstract type Instance
764 defined in 6.2.6.

765 Example:

```
766 {  
767   "kind": "indicationdeliveryrequest",  
768   "self": "/cimrs/dest1",  
769   "indication": {  
770     "kind": "instance",  
771     "class": "CIM_AlertIndication",  
772     "properties": {  
773       "AlertType": 4,  
774       "PerceivedSeverity": 6,  
775       "ProbableCause": 20,  
776       "Message": "ACME0007: Flood detected, height=3m.",  
777       "MessageArguments": [ "3" ],  
778       "MessageID": "ACME0007",  
779       "OwningEntity": "ACME" }  
780     }  
781 }
```

782 6.3.9 ErrorResponse payload element

783 ErrorResponse payload elements (see [DSP0210](#)) shall be represented using the following JSON
784 structure:

```
785 {  
786   "kind": "errorresponse",  
787   "self": (self),  
788   "httpmethod": (httpmethod),  
789   "statuscode": (statuscode),  
790   "statusdescription": (statusdescription),  
791   "errors": [  
792     (error-instance)#  
793   ]?  
794 }
```


795 Where:

- 796 • **(self)**, **(httpmethod)**, **(statuscode)**, and **(statusdescription)** are the values of the like-named
797 attributes of the represented ErrorResponse payload element, using the representation defined
798 in 6.2.6.
- 799 • Each array entry of *errors* represents an entry in the *errors* array attribute of the represented
800 ErrorResponse payload element; that is, an instance of class CIM_Error. **(error-instance)** is a
801 representation of the instance as defined in 6.2.6 for abstract datatype Instance.
802 If the array attribute *errors* of the represented ErrorResponse payload element has no entries,
803 the *errors* JSON object member should not be present (but may be present with a value of an
804 empty JSON array).

805 Example:

```
806 {  
807   "kind": "errorresponse",  
808   "self": "/cimrs/root%2Fcimv2/CIM_RegisteredProfile/DMTF%3AFan%3A1.1.0",  
809   "httpmethod": "GET",  
810   "statuscode": 12,  
811   "statusdescription": "ACME0008: Control program terminated with rc=42.",  
812   "errors": [  
813     {  
814       "kind": "instance",  
815       "class": "CIM_Error",  
816       "properties": {  
817         "ErrorType": 4,  
818         "PerceivedSeverity": 5,  
819         "ProbableCause": 48,  
820         "Message": "ACME0008: Control program terminated with rc=42.",  
821         "MessageArguments": [ "42" ],  
822         "MessageID": "ACME0008",  
823         "OwningEntity": "ACME" }  
824     } ]  
825 }
```

826
827
828
829
830

ANNEX A

(informative)

Change log

Version	Date	Description
1.0.0	2013-01-24	

831

Bibliography

832 This bibliography contains a list of non-normative references for this document.

833 DMTF DSP2032, *CIM-RS White Paper 1.0*,

834 http://www.dmtf.org/standards/published_documents/DSP2032_1.0.pdf

835 IANA MIME Media Types,

836 <http://www.iana.org/assignments/media-types/>

837 J. Holzer, *RESTful Web Services and JSON for WBEM Operations*, Master thesis, University of Applied
838 Sciences, Konstanz, Germany, June 2009,

839 <http://mond.htwg-konstanz.de/Abschlussarbeiten/Details.aspx?id=1120>