

OCF Resource to AllJoyn Interface Mapping Specification

VERSION 2.1.1 | February 2020



OPEN CONNECTIVITY
FOUNDATION™

CONTACT admin@openconnectivity.org

Copyright Open Connectivity Foundation, Inc. © 2020.
All Rights Reserved.

Legal Disclaimer

NOTHING CONTAINED IN THIS DOCUMENT SHALL BE DEEMED AS GRANTING YOU ANY KIND OF LICENSE IN ITS CONTENT, EITHER EXPRESSLY OR IMPLIEDLY, OR TO ANY INTELLECTUAL PROPERTY OWNED OR CONTROLLED BY ANY OF THE AUTHORS OR DEVELOPERS OF THIS DOCUMENT. THE INFORMATION CONTAINED HEREIN IS PROVIDED ON AN "AS IS" BASIS, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THE AUTHORS AND DEVELOPERS OF THIS SPECIFICATION HEREBY DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, STATUTORY OR AT COMMON LAW, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OPEN CONNECTIVITY FOUNDATION, INC. FURTHER DISCLAIMS ANY AND ALL WARRANTIES OF NON-INFRINGEMENT, ACCURACY OR LACK OF VIRUSES.

The OCF logo is a trademark of Open Connectivity Foundation, Inc. in the United States or other countries. *Other names and brands may be claimed as the property of others.

Copyright © 2017-2020 Open Connectivity Foundation, Inc. All rights reserved.

Copying or other form of reproduction and/or distribution of these works are strictly prohibited.

CONTENTS

22	1	Scope	1
23	2	Normative references	1
24	3	Terms and definitions	2
25	4	Document conventions and organization.....	2
26	4.1	Conventions.....	2
27	4.2	Notation.....	2
28	5	Theory of operation	3
29	5.1	Interworking approach	3
30	5.2	Mapping syntax.....	3
31	5.2.1	Introduction	3
32	5.2.2	General	3
33	5.2.3	Value assignment	3
34	5.2.4	Property naming	3
35	5.2.5	Arrays.....	4
36	5.2.6	Default mapping	4
37	5.2.7	Conditional mapping.....	4
38	5.2.8	Loops	4
39	5.2.9	Method invocation	4
40	6	AllJoyn translation	5
41	6.1	Operational scenarios	5
42	6.2	Requirements specific to an AllJoyn Bridging Function	5
43	6.2.1	Introduction	5
44	6.2.2	Use of introspection.....	5
45	6.2.3	Stability and loss of data.....	5
46	6.2.4	Exposing AllJoyn producer devices to OCF clients.....	6
47	6.2.5	Exposing OCF resources to AllJoyn consumer applications	14
48	6.2.6	Security	21
49	6.3	On-the-Fly Translation from D-Bus and OCF payloads.....	21
50	6.3.1	Introduction	21
51	6.3.2	Translation without aid of introspection.....	21
52	6.3.3	Translation with aid of introspection.....	27
53	7	Device type mapping	32
54	7.1	AllJoyn device types to OCF device types.....	32
55	7.2	OCF device types with no AllJoyn equivalent	33
56	8	Resource to interface equivalence	34
57	8.1	Introduction.....	34
58	8.2	Environment.CurrentAirQuality mapping	35
59	8.3	Environment.CurrentAirQualityLevel mapping	35
60	8.4	Operation.ClimateControlMode mapping	35
61	8.5	Operation.FanSpeedLevel mapping	35
62	8.6	Operation.HeatingZone mapping.....	36

63	8.7	Operation.OnOffStatus, Operation.OnControl, and Operation.OffControl	
64		mapping.....	36
65	8.8	Operation.OvenCyclePhase	36
66	9	Detailed mapping APIs	36
67	9.1	Introduction.....	36
68	9.2	Current Air Quality	37
69	9.2.1	Derived model	37
70	9.2.2	Property definition	37
71	9.2.3	Derived model definition	37
72	9.3	Current Air Quality Level.....	39
73	9.3.1	Derived model	39
74	9.3.2	Property definition	39
75	9.3.3	Derived model definition	40
76	9.4	Current Humidity	41
77	9.4.1	Derived model	41
78	9.4.2	Property definition	41
79	9.4.3	Derived model definition	41
80	9.5	Current Temperature.....	42
81	9.5.1	Derived model	42
82	9.5.2	Property definition	42
83	9.5.3	Derived model definition	43
84	9.6	Target Humidity	44
85	9.6.1	Derived model	44
86	9.6.2	Property definition	44
87	9.6.3	Derived model definition	45
88	9.7	Target Temperature	46
89	9.7.1	Derived model	46
90	9.7.2	Property definition	46
91	9.7.3	Derived model definition	46
92	9.8	Audio Volume	48
93	9.8.1	Derived model	48
94	9.8.2	Property definition	48
95	9.8.3	Derived model definition	48
96	9.9	Climate Control Mode	49
97	9.9.1	Derived model	49
98	9.9.2	Property definition	49
99	9.9.3	Derived model definition	50
100	9.10	Closed Status	51
101	9.10.1	Derived model	51
102	9.10.2	Property definition	51
103	9.10.3	Derived model definition	52
104	9.11	Cycle Control.....	52
105	9.11.1	Derived model	52
106	9.11.2	Property definition	52

107	9.11.3	Derived model definition	53
108	9.12	Fan Speed Level.....	54
109	9.12.1	Derived model	54
110	9.12.2	Property definition	54
111	9.12.3	Derived model definition	55
112	9.13	Heating Zone	56
113	9.13.1	Derived model	56
114	9.13.2	Property definition	56
115	9.13.3	Derived model definition	56
116	9.14	HVAC Fan Mode	57
117	9.14.1	Derived model	57
118	9.14.2	Property definition	57
119	9.14.3	Derived model definition	58
120	9.15	On/Off Control	59
121	9.15.1	Derived model	59
122	9.15.2	Property definition	59
123	9.15.3	Derived model definition	60
124	9.16	On Off Mapping	60
125	9.16.1	Derived model	60
126	9.16.2	Property definition	60
127	9.16.3	Derived model definition	61
128	9.17	Oven Cycle Phase	61
129	9.17.1	Derived model	61
130	9.17.2	Property definition	61
131	9.17.3	Derived model definition	62
132	10	Resource type definitions	63
133	10.1	List of Resource types	63
134	10.2	AllJoynObject.....	63
135	10.2.1	Introduction	63
136	10.2.2	Example URI	63
137	10.2.3	Resource type	63
138	10.2.4	OpenAPI 2.0 definition.....	63
139	10.2.5	Property definition	67
140	10.2.6	CRUDN behaviour	68
141	10.3	SecureMode	68
142	10.3.1	Introduction	68
143	10.3.2	Example URI	68
144	10.3.3	Resource type	68
145	10.3.4	OpenAPI 2.0 definition.....	68
146	10.3.5	Property definition	70
147	10.3.6	CRUDN behaviour	71
148			

	Figures	
149		
150	Figure 1 – Payload Chain.....	6
151		

Tables

152		
153	Table 1 – AllJoyn Bridging Function Interaction List.....	5
154	Table 2 – AllJoyn to OCF Name Examples.....	7
155	Table 3 – oic.wk.d resource type definition	9
156	Table 4 – oic.wk.con resource type definition.....	11
157	Table 5 – oic.wk.p resource type definition	12
158	Table 6 – oic.wk.con.p resource type definition.....	14
159	Table 7 – Example name mapping	15
160	Table 8 – AllJoyn about data fields	16
161	Table 9 – AllJoyn configuration data fields	20
162	Table 10 – Boolean translation	21
163	Table 11 – Numeric type translation, D-Bus to JSON	22
164	Table 12 – Numeric type translation, JSON to D-Bus	22
165	Table 13 – Text string translation.....	22
166	Table 14 – Byte array translation	23
167	Table 15 – D-Bus variant translation	23
168	Table 16 – D-Bus object path translation	23
169	Table 17 – D-Bus structure translation	23
170	Table 18 – Byte array translation	24
171	Table 19 – Other array translation	24
172	Table 21 – D-Bus dictionary translation.....	25
173	Table 22 – Non-translation types	25
174	Table 23 – D-Bus to JSON translation examples.....	26
175	Table 24 – JSON to D-Bus translation examples.....	27
176	Table 25 – JSON type to D-Bus type translation	29
177	Table 26 – D-Bus type to JSON type translation	29
178	Table 27 – Text string translation.....	30
179	Table 28 – JSON UUID string translation	30
180	Table 29 – D-Bus variant translation	30
181	Table 30 – D-Bus object path translation	30
182	Table 31 – Mapping from AllJoyn using introspection.....	31
183	Table 32 – Mapping from CBOR using introspection	32
184	Table 33 – AllJoyn to OCF device type mapping	32
185	Table 34 – OCF device types with no AllJoyn equivalent.....	33
186	Table 35 – AllJoyn interface to OCF resource type mapping – minimum interface set	34
187	Table 36 – AllJoyn interface to OCF resource type mapping – optional interface set.....	34
188	Table 37 – Interface to resource summary	36
189	Table 38 – The property mapping for "asa.environment.currentairquality".....	37
190	Table 39 – The properties of "asa.environment.currentairquality".....	37

191	Table 40 – The property mapping for "asa.environment.currentairqualitylevel".	39
192	Table 41 – The properties of "asa.environment.currentairqualitylevel".	40
193	Table 42 – The property mapping for "asa.environment.currenthumidity".	41
194	Table 43 – The properties of "asa.environment.currenthumidity".	41
195	Table 44 – The property mapping for "asa.environment.currenttemperature".	42
196	Table 45 – The properties of "asa.environment.currenttemperature".	42
197	Table 46 – The property mapping for "asa.environment.targethumidity".	44
198	Table 47 – The properties of "asa.environment.targethumidity".	44
199	Table 48 – The property mapping for "asa.environment.targettemperature".	46
200	Table 49 – The properties of "asa.environment.targettemperature".	46
201	Table 50 – The property mapping for "asa.operation.audiovolume".	48
202	Table 51 – The properties of "asa.operation.audiovolume".	48
203	Table 52 – The property mapping for "asa.operation.climatecontrolmode".	49
204	Table 53 – The properties of "asa.operation.climatecontrolmode".	50
205	Table 54 – The property mapping for "asa.operation.closedstatus".	51
206	Table 55 – The properties of "asa.operation.closedstatus".	51
207	Table 56 – The property mapping for "asa.operation.cyclecontrol".	52
208	Table 57 – The properties of "asa.operation.cyclecontrol".	53
209	Table 58 – The property mapping for "asa.operation.fanspeedlevel".	54
210	Table 59 – The properties of "asa.operation.fanspeedlevel".	54
211	Table 60 – The property mapping for "asa.operation.heatingzone".	56
212	Table 61 – The properties of "asa.operation.heatingzone".	56
213	Table 62 – The property mapping for "asa.operation.hvacfanmode".	58
214	Table 63 – The properties of "asa.operation.hvacfanmode".	58
215	Table 64 – The property mapping for "asa.operation.offcontrol".	59
216	Table 65 – The properties of "asa.operation.offcontrol".	59
217	Table 66 – The property mapping for "asa.operation.oncontrol".	59
218	Table 67 – The properties of "asa.operation.oncontrol".	59
219	Table 68 – The property mapping for "asa.operation.onoffstatus".	60
220	Table 69 – The properties of "asa.operation.onoffstatus".	61
221	Table 70 – The property mapping for "asa.operation.ovencyclephase".	61
222	Table 71 – The properties of "asa.operation.ovencyclephase".	62
223	Table 72 – Alphabetical list of resource types	63
224	Table 73 – The Property definitions of the Resource with type "rt" = "oic.r.alljoynobject,	67
225	oic.wk.col".	
226	Table 74 – The CRUDN operations of the Resource with type "rt" = "oic.r.alljoynobject,	68
227	oic.wk.col".	
228	Table 75 – The Property definitions of the Resource with type "rt" = "oic.r.securemode".	70
229	Table 76 – The CRUDN operations of the Resource with type "rt" = "oic.r.securemode".	71
230		

231 **1 Scope**

232 This document provides detailed mapping information to provide equivalency between AllJoyn
233 defined Interfaces and OCF defined Resources.

234 This document provides mapping for Device Types (AllJoyn to/from OCF), identifies equivalent
235 OCF Resources for both mandatory and optional AllJoyn interfaces and for each interface defines
236 the detailed Property by Property mapping using OCF defined extensions to JSON schema to
237 programmatically define the mappings.

238 **2 Normative references**

239 The following documents are referred to in the text in such a way that some or all of their content
240 constitutes requirements of this document. For dated references, only the edition cited applies. For
241 undated references, the latest edition of the referenced document (including any amendments)
242 applies.

243 ISO/IEC 30118-1:2018 Information technology -- Open Connectivity Foundation (OCF)
244 Specification -- Part 1: Core specification
245 <https://www.iso.org/standard/53238.html>
246 Latest version available at: https://openconnectivity.org/specs/OCF_Core_Specification.pdf

247 ISO/IEC 30118-2:2018 Information technology -- Open Connectivity Foundation (OCF)
248 Specification -- Part 2: Security specification
249 <https://www.iso.org/standard/74239.html>
250 Latest version available at: https://openconnectivity.org/specs/OCF_Security_Specification.pdf

251 ISO/IEC 30118-4:2018 Information technology -- Open Connectivity Foundation (OCF)
252 Specification -- Part 4: Resource type specification
253 <https://www.iso.org/standard/74241.html>
254 Latest version available at:
255 https://openconnectivity.org/specs/OCF_Resource_Type_Specification.pdf

256 ISO/IEC 30118-5:2019, Information technology – Open Connectivity Foundation (OCF)
257 Specification – Part 5: Smart home device specification
258 <https://www.iso.org/standard/74242.html>
259 Latest version available at: https://openconnectivity.org/specs/OCF_Device_Specification.pdf

260 JSON Hyper-Schema, *JSON Hyper-Schema: A Vocabulary for Hypermedia Annotation of JSON*,
261 October 2016
262 <http://json-schema.org/latest/json-schema-hypermedia.html>

263 Derived Models for Interoperability between IoT Ecosystems, Stevens & Merriam, March 2016
264 [https://www.iab.org/wp-content/IAB-uploads/2016/03/OCF-Derived-Models-for-Interoperability-](https://www.iab.org/wp-content/IAB-uploads/2016/03/OCF-Derived-Models-for-Interoperability-Between-IoT-Ecosystems_v2-examples.pdf)
265 [Between-IoT-Ecosystems_v2-examples.pdf](https://www.iab.org/wp-content/IAB-uploads/2016/03/OCF-Derived-Models-for-Interoperability-Between-IoT-Ecosystems_v2-examples.pdf)

266 IETF RFC 4122, *A Universally Unique Identifier (UUID) URN Namespace*, July 2005
267 <https://www.rfc-editor.org/info/rfc4122>

268 IETF RF 4648, *The Base16, Base32 and Base64 Data Encodings*, October 2006
269 <https://www.rfc-editor.org/info/rfc4648>

270 IETF RFC 6973, *Privacy Considerations for Internet Protocols*, July 2013
271 <https://www.rfc-editor.org/info/rfc6973>

272 IETF RFC 7159, *The JavaScript Object Notation (JSON) Data Interchange Format*, March 2014
273 <https://www.rfc-editor.org/info/rfc7159>

274 AllJoyn Common Data Model Interface Definitions
275 <https://wiki.alljoyn.org/cdm>

276 AllJoyn About Interface Specification, *About Feature Interface Definitions*, Version 14.12
277 <https://allseenalliance.org/framework/documentation/learn/core/about-announcement/interface>

278 AllJoyn Configuration Interface Specification, *Configuration Interface Definition*, Version 14.12
279 <https://allseenalliance.org/framework/documentation/learn/core/configuration/interface>

280 D-Bus Specification, *D-Bus Specification*
281 <https://dbus.freedesktop.org/doc/dbus-specification.html>

282 **3 Terms and definitions**

283 For the purposes of this document, the terms and definitions given in ISO/IEC 30118-1:2018 and
284 the following apply.

285 ISO and IEC maintain terminological databases for use in standardization at the following
286 addresses:

- 287 – ISO Online browsing platform: available at <https://www.iso.org/obp>
- 288 – IEC Electropedia: available at <http://www.electropedia.org/>

289 **4 Document conventions and organization**

290 **4.1 Conventions**

291 In this document a number of terms, conditions, mechanisms, sequences, parameters, events,
292 states, or similar terms are printed with the first letter of each word in uppercase and the rest
293 lowercase (e.g., Network Architecture). Any lowercase uses of these words have the normal
294 technical English meaning.

295 **4.2 Notation**

296 In this document, features are described as required, recommended, allowed or DEPRECATED as
297 follows:

298 Required (or shall or mandatory).

299 These basic features shall be implemented to comply with the Mapping Specification. The
300 phrases "shall not", and "PROHIBITED" indicate behaviour that is prohibited, i.e. that if
301 performed means the implementation is not in compliance.

302 Recommended (or should).

303 These features add functionality supported by the Mapping Specification and should be
304 implemented. Recommended features take advantage of the capabilities the Mapping
305 Specification, usually without imposing major increase of complexity. Notice that for compliance
306 testing, if a recommended feature is implemented, it shall meet the specified requirements to
307 be in compliance with these guidelines. Some recommended features could become
308 requirements in the future. The phrase "should not" indicates behaviour that is permitted but
309 not recommended.

310 Allowed (or allowed).

311 These features are neither required nor recommended by the Mapping Specification, but if the
312 feature is implemented, it shall meet the specified requirements to be in compliance with these
313 guidelines.

314 Conditionally allowed (CA)

315 The definition or behaviour depends on a condition. If the specified condition is met, then the
316 definition or behaviour is allowed, otherwise it is not allowed.

317 Conditionally required (CR)

318 The definition or behaviour depends on a condition. If the specified condition is met, then the
319 definition or behaviour is required. Otherwise the definition or behaviour is allowed as default
320 unless specifically defined as not allowed.

321 DEPRECATED

322 Although these features are still described in this document, they should not be implemented
323 except for backward compatibility. The occurrence of a deprecated feature during operation of
324 an implementation compliant with the current document has no effect on the implementation's
325 operation and does not produce any error conditions. Backward compatibility may require that
326 a feature is implemented and functions as specified but it shall never be used by
327 implementations compliant with this document.

328 Strings that are to be taken literally are enclosed in "double quotes".

329 Words that are emphasized are printed in *italic*.

330 **5 Theory of operation**

331 **5.1 Interworking approach**

332 The interworking between AllJoyn defined interfaces and OCF defined Resource Types is modelled
333 using the derived model syntax described in Derived Models for Interoperability between IoT
334 Ecosystems. Determination of the minimum set of AllJoyn interfaces for which equivalency is
335 required within the OCF data model was done by listing the set of interfaces required for each of
336 the device types defined by the CDM Project inside of AllJoyn. Where the AllJoyn interface supports
337 methods then an actuation design pattern is applied.

338 **5.2 Mapping syntax**

339 **5.2.1 Introduction**

340 Within the defined syntax for derived modelling used by this document there are two blocks that
341 define the actual Property-Property equivalence or mapping. These blocks are identified by the
342 keywords "x-to-ocf" and "x-from-ocf". Derived Models for Interoperability between IoT Ecosystems
343 does not define a rigid syntax for these blocks; they are free form string arrays that contain pseudo-
344 coded mapping logic.

345 Within this document we apply the rules defined in clause 3.2 to these blocks to ensure consistency
346 and re-usability and extensibility of the mapping logic that is defined.

347 **5.2.2 General**

348 All statements are terminated with a carriage return.

349 **5.2.3 Value assignment**

350 The equals sign (=) is used to assign one value to another. The assignee is on the left of the
351 operator; the value being assigned on the right.

352 **5.2.4 Property naming**

353 All Property names are identical to the name used by the original model; for example, from the
354 OCF Temperature Resource the Property name "temperature" is used whereas when referred to
355 the derived ecosystem then the semantically equivalent Property name is used.

356 When the same name is used by both OCF and the derived ecosystem for semantically equivalent
357 values then the name of the OCF defined Property is prepended by the ecosystem designator "ocf"
358 to avoid ambiguity (e.g. "ocf.step").

359 **5.2.5 Arrays**

360 An array element is indicated by the use of square brackets "[]" with the index of the element
361 contained therein, e.g. range[1]. All arrays start at an index of 0. If an entire array is being
362 referenced then no index is included, e.g. selectablehumiditylevels[].

363 **5.2.6 Default mapping**

364 There are cases where the specified mapping is not possible as one or more of the Properties
365 being mapped is optional in the source model. In all such instances a default mapping is provided.
366 The default map is indicated by the prepending of an "otherwise:" modifier to the assignment. (e.g.
367 "otherwise: step = 1").

368 **5.2.7 Conditional mapping**

369 When a mapping is dependent on the meeting of other conditions then the syntax:

370 if "condition", "mapping".

371 Is applied.

372 E.g. if step >0, ocf.step = step.

373 **5.2.8 Loops**

374 When a mapping can be represented by a repeated loop governed by some condition then the
375 syntax:

376 for "initialize", "condition", "increment": "mapping"

377 Where:

378 "initialize" is an initial local loop control variable setting.

379 "condition" is the loop controller, the loop repeats until the condition evaluates to "false".

380 "increment" allows for update of the control variable, if omitted an increment of "1" is assumed.

381 Is applied.

382 E.g. for x=0, x < sizeof(supportedmodes): ocf.supportedmodes[x] = modearray[supportedmodes[x]]

383 **5.2.9 Method invocation**

384 The invocation of a method or remote procedure call (RPC) from the derived ecosystem as part of
385 the mapping from an OCF Resource is indicated by the use of a double colon "::" delimiter between
386 the applicable resource, service, interface or other construct identifier and the method or RPC
387 name. The method name always includes trailing parentheses which would include any parameters
388 should they be passed.

389 For example, when dealing with the switchon() method from AllJoyn this gives a complete method
390 invocation as: operation.oncontrol::switchon().

391 **6 AllJoyn translation**

392 **6.1 Operational scenarios**

393 The overall goals are to:

- 394 1) make Bridged Servers appear to OCF clients as if they were native OCF servers, and
- 395 2) make OCF servers appear to Bridged Clients as if they were native non-OCF servers.

396 **6.2 Requirements specific to an AllJoyn Bridging Function**

397 **6.2.1 Introduction**

398 The Bridge Platform shall be an AllJoyn Router Node. (This is a requirement so that users can
399 expect that a certified Bridge will be able to talk to any AllJoyn device, without the user having to
400 buy some other device.)

401 The requirements in clause 4.2 apply when using algorithmic translation, and by default apply to
402 deep translation unless the relevant clause for such deep translation specifies otherwise.

403 **6.2.2 Use of introspection**

404 Whenever possible, the translation code should make use of metadata available that indicates what
405 the sender and recipient of the message in question are expecting. For example, devices that are
406 AllJoyn Certified are required to carry the introspection data for each object and interface they
407 expose. When the metadata is available, Bridging Functions should convert the incoming payload
408 to exactly the format expected by the recipient and should use information when translating replies
409 to form a more useful message.

410 For example, for an AllJoyn specific Bridging Function, the expected interaction list is presented in
411 Table 1.

412 **Table 1 – AllJoyn Bridging Function Interaction List**

Message Type	Sender	Receiver	Metadata
Request	AllJoyn 16.10	OCF 1.0	Available
Request	OCF 1.0	AllJoyn 16.10	Available
Response	AllJoyn 16.10	OCF 1.0	Available
Response	OCF 1.0	AllJoyn 16.10	Available

413 **6.2.3 Stability and loss of data**

414 Round-tripping through the translation process specified in this document is not expected to
415 reproduce the same original message. The process is, however, designed not to lose data or
416 precision in messages, though it should be noted that both OCF and AllJoyn payload formats allow
417 for future extensions not considered in this document.

418 However, a third round of translation should produce the same identical message as was previously
419 produced, provided the same information is available. That is, in the chain shown in Figure 1,
420 payloads 2 and 4 as well as 3 and 5 should be identical.

421



Figure 1 – Payload Chain.

422

423

424 6.2.4 Exposing AllJoyn producer devices to OCF clients

425 6.2.4.1 Virtual OCF Devices and Resources

426 As specified in ISO/IEC 30118-2:2018 the value of the "di" property of OCF Devices (including
427 VODs) shall be established as part of Onboarding of that VOD.

428 Each AllJoyn object shall be mapped to one or more Virtual OCF Resources. If all AllJoyn interfaces
429 can be translated to resource types on the same resource, there should be a single Virtual OCF
430 Resource, and the path component of the URI of the Virtual OCF Resource shall be the AllJoyn
431 object path, where each "_h" in the AllJoyn object path is transformed to "-" (hyphen), each "_d"
432 in the AllJoyn object path is transformed to "." (dot), each "_t" in the AllJoyn object path is transformed
433 to "~" (tilde), and each "_u" in the AllJoyn object path is transformed to "_" (underscore). Otherwise,
434 a Resource with that path shall exist with a Resource Type of ["oic.wk.col", "oic.r.alljoynobject"]
435 which is a Collection of links, where "oic.r.alljoynobject" is defined in clause 8.2 and the items in
436 the collection are the Resources with the translated Resource Types.

437 The value of the "piid" property of "/oic/d" for each VOD shall be the value of the OCF-defined
438 AllJoyn field "org.openconnectivity.piid" in the AllJoyn About Announce signal, if that field exists,
439 else it shall be calculated by the Bridging Function as follows:

- 440 – If the AllJoyn device supports security, the value of the "piid" property value shall be the peer
441 GUID.
- 442 – If the AllJoyn device does not support security but the device is being bridged anyway (see 8.2),
443 the "piid" property value shall be derived from the DeviceId and AppId properties (in the About
444 data), by concatenating the DeviceId value (not including any null termination) and the AppId
445 bytes and using the result as the "name" to be used in the algorithm specified in IETF RFC 4122
446 clause 4.3, with SHA-1 as the hash algorithm, and 8f0e4e90-79e5-11e6-bdf4-0800200c9a66
447 as the name space ID. (This is to address the problem of being able to de-duplicate AllJoyn
448 devices exposed via separate OCF Bridge Devices.)

449 A Bridging Function implementation is encouraged to listen for AllJoyn About Announce signals
450 matching any AllJoyn interface name. It can maintain a cache of information it received from these
451 signals, and use the cache to quickly handle "/oic/res" queries from OCF Clients (without having to
452 wait for Announce signals while handling the queries).

453 A Bridging Function implementation is encouraged to listen for other signals (including
454 EmitsChangedSignal of properties) only when there is a client subscribed to a corresponding
455 resource on a Virtual AllJoyn Device.

456 There are multiple types of AllJoyn interfaces, which shall be handled as follows.

- 457 1) If the AllJoyn interface is in a well-defined set (defined in 4.2.4.2) of interfaces where standard
458 forms exist on both the AllJoyn and OCF sides, the Bridging Function shall either:
 - 459 a) follow the requirements for translating that interface specially, or

460 b) not translate the AllJoyn interface.
 461 2) If the AllJoyn interface is not in the well-defined set, the Bridging Function shall either:
 462 a) not translate the AllJoyn interface, or
 463 b) algorithmically map the AllJoyn interface as specified in 4.3 to custom/vendor-defined
 464 Resource Types by converting the AllJoyn interface name to OCF resource type name(s).
 465 An AllJoyn interface name shall be converted to a Device Type or a set of one or more OCF
 466 Resource Types as follows:

- 467 3) If the AllJoyn interface has any members, append a suffix ".<seeBelow>" where <seeBelow> is
 468 described in this clause.
- 469 4) For each upper-case letter present in the entire string, replace it with a hyphen followed by the
 470 lower-case version of that letter (e.g., convert "A" to "-a").
- 471 5) If an underscore appears followed by a (lower-case) letter or a hyphen, for each such
 472 occurrence, replace the underscore with two hyphens (e.g., convert "_a" to "--a", "_-a" to "---
 473 a").
- 474 6) For each underscore remaining, replace it with a hyphen (e.g., convert "_1" to "-1").
- 475 7) Prepend the "x." prefix.

476 Some examples are shown in Table 2. The first three are normal AllJoyn names converted to
 477 unusual OCF names. The last three are unusual AllJoyn names converted (perhaps back) to normal
 478 OCF names. ("xn--" is a normal domain name prefix for the Punycode-encoded form of an
 479 Internationalized Domain Name, and hence can appear in a normal vendor-specific OCF name.)

480 **Table 2 – AllJoyn to OCF Name Examples**

From AllJoyn name	To OCF name
example.Widget	x.example.-widget
example.my__widget	x.example.my----widget
example.My_Widget	x.example.-my---widget
xn_p1ai.example	x.xn--p1ai.example
xn__90ae.example	x.xn--90ae.example
example.myName_1	x.example.my-name-1

481 Each AllJoyn interface that has members and is using algorithmic mapping shall be mapped to one
 482 or more Resource Types as follows:

- 483 – AllJoyn Properties with the same EmitsChangedSignal value are mapped to the same Resource
 484 Type where the value of the <seeBelow> label is the value of EmitsChangedSignal. AllJoyn
 485 Properties with EmitsChangedSignal values of "const" or "false", are mapped to Resources that
 486 are not Observable, whereas AllJoyn Properties with EmitsChangedSignal values of "true" or
 487 "invalidates" result in Resources that are Observable. The Version property in an AllJoyn
 488 interface is always considered to have an EmitsChangedSignal value of "const", even if not
 489 specified in introspection XML. The name of each property on the Resource Type shall be
 490 "<ResourceType>.<AllJoynPropertyName>", where each "_d" in the <AllJoynPropertyName> is
 491 transformed to "." (dot), and each "_h" in the <AllJoynPropertyName> is transformed to "-"
 492 (hyphen).
- 493 – Resource Types mapping AllJoyn Properties with access "readwrite" shall support the "oic.if.rw"
 494 OCF Interface. Resource Types mapping AllJoyn Properties with access "read" shall support
 495 the "oic.if.r" OCF Interface. Resource Types supporting both the "oic.if.rw" and "oic.if.r" OCF
 496 Interfaces shall choose "oic.if.r" as the default Interface.

- 497 – Each AllJoyn Method is mapped to a separate Resource Type, where the value of the
498 <seeBelow> label is the AllJoyn Method name. The Resource Type shall support the "oic.if.rw"
499 OCF Interface. Each argument of the AllJoyn Method shall be mapped to a separate Property
500 on the Resource Type, where the name of that Property is prefixed with
501 "<ResourceType>arg<#>", where <#> is the 0-indexed position of the argument in the AllJoyn
502 introspection xml, in order to help get uniqueness across all Resource Types on the same
503 Resource. Therefore, when the AllJoyn argument name is not specified, the name of that
504 property is "<ResourceType>arg<#>", where <#> is the 0-indexed position of the argument in
505 the AllJoyn introspection XML. In addition, that Resource Type has an extra
506 "<ResourceType>validity" property that indicates whether the rest of the properties have valid
507 values. When the values are sent as part of an UPDATE response, the validity property is true,
508 and any other properties have valid values. In a RETRIEVE (GET or equivalent in the relevant
509 transport binding) response, the validity property is false, and any other properties can have
510 meaningless values. If the validity property appears in an UPDATE request, its value shall be
511 true (a value of false shall result in an error response).
- 512 – Each AllJoyn Signal (whether sessionless, sessioncast, or unicast) is mapped to a separate
513 Resource Type on an Observable Resource, where the value of the <seeBelow> label is the
514 AllJoyn Signal name. The Resource Type shall support the "oic.if.r" OCF Interface. Each
515 argument of the AllJoyn Signal is mapped to a separate Property on the Resource Type, where
516 the name of that Property is prefixed with "<ResourceType>arg<#>", where <#> is the 0-indexed
517 position of the argument in the AllJoyn introspection xml, in order to help get uniqueness across
518 all Resource Types on the same Resource. Therefore, when the AllJoyn argument name is not
519 specified, the name of that property is "<ResourceType>arg<#>", where <#> is the 0-indexed
520 position of the argument in the AllJoyn introspection XML. In addition, that Resource Type has
521 an extra "<ResourceType>validity" property that indicates whether the rest of the properties
522 have valid values. When the values are sent as part of a NOTIFY response, the validity property
523 is true, and any other properties have valid values. In a RETRIEVE (GET or equivalent in the
524 relevant transport binding) response, the validity property is false, and any other properties
525 returned can have meaningless values. This is because in AllJoyn, the signals are
526 instantaneous events, and the values are not necessarily meaningful beyond the lifetime of that
527 message. Note that AllJoyn does have a TTL field that allows store-and-forward signals, but
528 such support is not required in OCF 1.0. We expect that in the future, the TTL may be used to
529 allow valid values in response to a RETRIEVE that is within the TTL.

530 When an algorithmic mapping is used, AllJoyn data types shall be mapped to OCF property types
531 according to 4.3.

532 If an AllJoyn operation fails, the Bridging Function shall send an appropriate OCF error response
533 to the OCF client. If an AllJoyn error name is available and does not contain the
534 "org.openconnectivity.Error.Code" prefix, it shall construct an appropriate OCF error message (e.g.,
535 diagnostic payload if using CoAP) from the AllJoyn error name and AllJoyn error message (if any),
536 using the form "<error name>: <error message>", with the <error name> taken from the AllJoyn
537 error name field and the <error message> taken from the AllJoyn error message, and the CoAP
538 error code set to an appropriate value (if CoAP is used). If an AllJoyn error name is available and
539 contains the "org.openconnectivity.Error.Code" prefix, the OCF error message (e.g., diagnostic
540 payload if using CoAP) should be taken from the AllJoyn error message (if any), and the CoAP
541 error code (if CoAP is used) set to a value derived as follows; remove the
542 "org.openconnectivity.Error.Code" prefix, and if the resulting error name is of the form "<#>" where
543 <#> is an error code without a decimal (e.g., "404"), the CoAP error code shall be the error code
544 indicated by the "<#>". Example: "org.openconnectivity.Error.Code404" becomes "404", which shall
545 result in an error 4.04 for a CoAP transport.

546 **6.2.4.2 Exposing an AllJoyn producer application as a Virtual OCF Server**

547 Table 3 shows how OCF Device properties, as specified in Table 27 in ISO/IEC 30118-1:2018 shall
548 be derived, typically from fields specified in the ISO/IEC 30118-1:2018 and AllJoyn Configuration
549 Interface Specification.

550 If the AllJoyn About or Config data field has a mapping rule defined (as in Table 3, Table 4, Table 5,
 551 and Table 6), the field name shall be translated based on that mapping rule; else if the AllJoyn
 552 About or Config data field has a fully qualified name (with a <domain> prefix (such as
 553 "com.example", "org.alljoyn"), the field name shall be translated based on the rules specified in
 554 4.2.4 for mapping AllJoyn fields; else, the field shall not be translated as it may be incorrect (error)
 555 or it has no valid mapping (such as daemonRealm and passCode).

556

Table 3 – oic.wk.d resource type definition

To OCF Property title	OCF Property name	OCF Description	OCF Mandatory	From AJ Field name	AJ Description	AJ Mandatory
(Device) Name	n	Human friendly name For example, "Bob's Thermostat"	Y	AppName (no exact equivalent exists)	Application name assigned by the app manufacturer (developer or the OEM).	Y
Spec Version	icv	Spec version of ISO/IEC 30118-1:2018 this device is implemented to, the syntax is "core.major.minor"]	Y	(none)	Bridge Platform should return its own value	N
Device ID	di	Unique identifier for Device. This value shall be as defined in ISO/IEC 30118-2:2018 for DeviceID.	Y	(none)	Use as defined in ISO/IEC 30118-2:2018	N
Protocol-Independent ID	piid	Unique identifier for OCF Device (UUID)	Y	org.openconnectivity.piid if it exists, else "Peer GUID" (not in About, but exposed by protocol) if authenticated, else Hash(DeviceId,Appld) where the Hash is done by concatenating the Device Id (not including any null terminator) and the Appld and using the algorithm in IETF RFC 4122 clause 4.3, with SHA-1. This means that the value of di may change if the resource is read both before and after authentication, in order to mitigate privacy concerns discussed in RFC 6973.	Peer GUID: The peer GUID is the only persistent identity for a peer. Peer GUIDs are used by the authentication mechanisms to uniquely identify a remote application instance. The peer GUID for a remote peer is only available if the remote peer has been authenticated. DeviceId: Device identifier set by platform-specific means. Appld: A 128-bit globally unique identifier for the application. The Appld shall be a universally unique identifier as specified in IETF RFC 4122 .	Per GUID: conditionally Y DeviceId: Y Appld: Y

Data Model Version	dmv	Spec version(s) of the vertical specifications this device data model is implemented to. The syntax is a comma separated list of "<vertical>.major.minor". <vertical> is the name of the vertical (i.e. sh for Smart Home)	Y	Comma separated list of the Version property values of each interface listed in the objectDescription argument of the Announce signal of About. In addition to the mandatory values specified in ISO/IEC 30118-1:2018, additional values are formatted as "x.<interface name>.<Version property value>".	This document assumes that the value of the Version property is the same as the value of the "org.gtk.GDBus.Since" annotation of the interface in the AllJoyn introspection XML, and therefore the value of the Version property may be determined through introspection alone. Note that AllJoyn specifies that the default value is 1 if the "org.gtk.GDBus.Since" annotation is absent.	N, but required by IRB for all standard interfaces, and absence can be used to imply a constant (e.g., 0)
Localized Descriptions	ld	Detailed description of the Device, in one or more languages. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the device description in the indicated language.	N	Description	Detailed description expressed in language tags as in RFC 5646.	Y
Software Version	sv	Version of the device software.	N	SoftwareVersion	Software version of the app.	Y
Manufacturer Name	dmn	Name of manufacturer of the Device, in one or more languages. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the manufacturer name in the indicated language.	N	Manufacturer	The manufacturer's name of the app.	Y
Model Number	dmno	Model number as designated by manufacturer.	N	ModelNumber	The app model number.	Y

558 In addition, any additional vendor-defined fields in the AllJoyn About data shall be mapped to
 559 vendor-defined properties in the OCF Device resource "/oic/d" (which implements the "oic.wk.d"
 560 resource type), with a property name formed by prepending "x." to the AllJoyn field name.

561 Table 4 shows how OCF Device Configuration properties, as specified in Table 22 in ISO/IEC
 562 30118-1:2018 shall be derived:

563 **Table 4 – oic.wk.con resource type definition**

To OCF Property title	OCF Property name	OCF Description	OCF Mandatory	From AJ Field name	AJ Description	AJ Mandatory
(Device) Name	n	Human friendly name For example, "Bob's Thermostat"	Y	AppName (no exact equivalent exists)	Application name assigned by the app manufacturer (developer or the OEM).	Y
Location	loc	Provides location information where available.	N	org.openconnectivity.loc (if it exists, else property shall be absent)		N
Location Name	locn	Human friendly name for location For example, "Living Room".	N	org.openconnectivity.locn (if it exists, else property shall be absent)		N
Currency	c	Indicates the currency that is used for any monetary transactions	N	org.openconnectivity.c (if it exists, else property shall be absent)		N
Region	r	Free form text Indicating the current region in which the device is located geographically. The free form text shall not start with a quote (").	N	org.openconnectivity.r (if it exists, else property shall be absent)		N
Localized Names	ln	Human-friendly name of the Device, in one or more languages. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the device name in the indicated language. If this	N	AppName	Application name assigned by the app manufacturer (developer or the OEM).	Y

		property and the Device Name (n) property are both supported, the Device Name (n) value shall be included in this array.				
Default Language	dl	The default language supported by the Device, specified as an RFC 5646 language tag. By default, clients can treat any string property as being in this language unless the property specifies otherwise.	N	DefaultLanguage	The default language supported by the device. Specified as an IETF language tag listed in RFC 5646.	Y

564

565 In addition, any additional vendor-defined fields in the AllJoyn Configuration data shall be mapped
566 to vendor-defined properties in the OCF Configuration resource (which implements the "oic.wk.con"
567 resource type and optionally the "oic.wk.con.p" resource type), with a property name formed by
568 prepending "x." to the AllJoyn field name.

569 Table 5 shows how OCF Platform properties, as specified in Table 28 in ISO/IEC 30118-1:2018
570 shall be derived, typically from fields specified in the AllJoyn About Interface Specification and
571 AllJoyn Configuration Interface Specification.

572

Table 5 – oic.wk.p resource type definition

To OCF Property title	OCF Property name	OCF Description	OCF Mandatory	From AJ Field name	AJ Description	AJ Mandatory
Platform ID	pi	Unique identifier for the physical platform (UUID); this shall be a UUID in accordance with IETF RFC 4122. It is recommended that the UUID be created using the random generation scheme (version 4 UUID) specific in the RFC.	Y	DeviceId if it is a UUID, else generate a name-based UUID from the DeviceId using the DeviceId value (not including any null termination) as the "name" to be used in the algorithm specified in IETF RFC 4122 clause 4.3, with SHA-1 as the hash algorithm, and 8f0e4e90-79e5-11e6-bdf4-0800200c9a66 as the name space ID.	Name of the device set by platform-specific means (such as Linux and Android).	Y

Manufacturer Name	mnmn	Name of manufacturer (not to exceed 16 characters)	Y	Manufacturer (in DefaultLanguage, truncated to 16 characters)	The manufacturer's name of the app.	Y
Manufacturer Details Link (URL)	mnml	URL to manufacturer (not to exceed 32 characters)	N	org.openconnectivity.mnml (if it exists, else property shall be absent)		N
Model Number	mnmo	Model number as designated by manufacturer	N	ModelNumber	The app model number.	Y
Date of Manufacture	mndt	Manufacturing date of device	N	DateOfManufacture	Date of manufacture using format YYYY-MM-DD (known as XML DateTime format).	N
Platform Version	mnpv	Version of platform – string (defined by manufacturer)	N	org.openconnectivity.mnpv (if it exists, else property shall be absent)		N
OS Version	mnos	Version of platform resident OS – string (defined by manufacturer)	N	org.openconnectivity.mnos (if it exists, else property shall be absent)		N
Hardware Version	mnhw	Version of platform hardware	N	HardwareVersion	Hardware version of the device on which the app is running.	N
Firmware version	mnfv	Version of device firmware	N	org.openconnectivity.mnfv (if it exists, else property shall be absent)		N
Support URL	mnsi	URL that points to support information from manufacturer	N	SupportUrl	Support URL (populated by the manufacturer)	N
SystemTime	st	Reference time for the device	N	org.openconnectivity.st (if it exists, else property shall be absent)		N
Vendor ID	vid	Vendor defined string for the platform. The string is freeform and up to the vendor on what text to populate it.	N	DeviceId	Name of the device set by platform-specific means (such as Linux and Android).	Y

573 Table 6 shows how OCF Platform Configuration properties, as specified in Table 23 in the ISO/IEC
 574 30118-1:2018 shall be derived:

575 **Table 6 – oic.wk.con.p resource type definition**

To OCF Property title	OCF Property name	OCF Description	OCF Mandatory	From AJ Field name	AJ Description	AJ Mandatory
Platform Names	Mnnp	Platform Identifier	N	DeviceName	Name of the device set by platform-specific means (such as Linux and Android).	Device name assigned by the user. The device name appears on the UI as the friendly name of the device.

576

577 In addition, the "oic.wk.mnt" properties Factory_Reset ("fr") and Reboot ("rb") shall be mapped to
 578 AllJoyn Configuration methods FactoryReset and Restart, respectively.

579 **6.2.5 Exposing OCF resources to AllJoyn consumer applications**

580 **6.2.5.1 Use of AllJoyn Producer Application**

581 Unless specified otherwise, each OCF resource shall be mapped to a separate AllJoyn object.

582 Each OCF Server shall be exposed as a separate AllJoyn producer application, with its own About
 583 data. This allows platform-specific, device-specific, and resource-specific fields to all be preserved
 584 across translation. However, this requires that AllJoyn Claiming of such producer applications be
 585 solved in a way that does not require user interaction, but this is left as an implementation issue.

586 The AllJoyn producer application shall implement the "oic.d.virtual" AllJoyn interface. This allows
 587 Bridge Platforms to determine if a device is already being translated when multiple Bridge Platforms
 588 are present. The "oic.d.virtual" interface is defined as follows:

```
589 <interface name="oic.d.virtual"/>
```

590 The implementation may choose to implement this interface by the AllJoyn object at path "/oic/d".

591 The AllJoyn peer ID shall be the OCF device ID ("di").

592 Unless specified otherwise, the AllJoyn object path shall be the OCF URI path, where each "-"
 593 (hyphen) in the OCF URI path is transformed to "_h", each "." (dot) in the OCF URI path is
 594 transformed to "_d", each "~" (tilde) in the OCF URI path is transformed to "_t", and each "_"
 595 (underscore) in the OCF URI path is transformed to "_u".

596 The AllJoyn About data shall be populated per Table 8.

597 A Bridging Function implementation is encouraged to maintain a cache of OCF resources to handle
 598 the implementation of queries from the AllJoyn side, and emit an Announce Signal for each OCF
 599 Server. Specifically, the implementation could always Observe "/oic/res" changes and only Observe
 600 other resources when there is a client with a session on a Virtual AllJoyn Device.

601 There are multiple types of resources, which shall be handled as follows.

602 1) If the Resource Type is in a well-defined set (defined in 4.2.5.2) of resource types where
 603 standard forms exist on both the AllJoyn and OCF sides, the Bridging Function shall either:

604 a) follow the requirements for translating that resource type specially, or

- 605 b) not translate the Resource Type.
- 606 2) If the Resource Type is not in the well-defined set (but is not a Device Type), the Bridging
607 Function shall either:
- 608 a) not translate the Resource Type, or
- 609 b) algorithmically map the Resource Type as specified in 4.3 to a custom/vendor-defined
610 AllJoyn interface by converting the OCF Resource Type name to an AllJoyn Interface name.
- 611 An OCF Resource Type or Device Type shall be converted to an AllJoyn interface name as follows:

- 612 1) Remove the "x." prefix if present
- 613 2) For each occurrence of a hyphen (in order from left to right in the string):
- 614 a) If the hyphen is followed by a letter, replace both characters with a single upper-case version
615 of that letter (e.g., convert "-a" to "A").
- 616 b) Else, if the hyphen is followed by another hyphen followed by either a letter or a hyphen,
617 replace two hyphens with a single underscore (e.g., convert "--a" to "_a", "---" to "_-").
- 618 c) Else, convert the hyphen to an underscore (i.e., convert "-" to "_").

619 Some examples are shown in the Table 7. The first three are unusual OCF names converted
620 (perhaps back) to normal AllJoyn names. The last three are normal OCF names converted to
621 unusual AllJoyn names. ("xn--" is a normal domain name prefix for the Punycode-encoded form of
622 an Internationalized Domain Name, and hence can appear in a normal vendor-specific OCF name.)

623 **Table 7 – Example name mapping**

From OCF name	To AllJoyn name
x.example.-widget	example.Widget
x.example.my----widget	example.my__widget
x.example.-my---widget	example.My_Widget
x.xn--p1ai.example	xn_p1ai.example
x.xn--90ae.example	xn__90ae.example
x.example.my-name-1	example.myName_1

- 624 An OCF Device Type is mapped to an AllJoyn interface with no members.
- 625 Unless specified otherwise, each OCF Resource Type shall be mapped to an AllJoyn interface as
626 follows:
- 627 – Each OCF property is mapped to an AllJoyn property in that interface, where each "." (dot) in
628 the OCF property is transformed to "_d", and each "-" (hyphen) in the OCF property is
629 transformed to "_h".
 - 630 – The EmitsChangedSignal value for each AllJoyn property shall be set to "true" if the resource
631 supports NOTIFY, or "false" if it does not. (The value is never set to "const" or "invalidates"
632 since those concepts cannot currently be expressed in OCF.)
 - 633 – The "access" attribute for each AllJoyn property shall be "read" if the OCF property is read-only,
634 or "readwrite" if the OCF property is read-write.
 - 635 – If the resource supports DELETE, a Delete() method shall appear in the interface.
 - 636 – If the resource supports CREATE, a Create() method shall appear in the interface, with input
637 arguments of each property of the resource to create. (Such information is not available
638 algorithmically can be determined via introspection.) If such information is not available, a
639 CreateWithDefaultValues() method shall appear which takes no input arguments. In either case,
640 the output argument shall be an OBJECT_PATH containing the path of the created resource.

- 641 – If the resource supports UPDATE (i.e., the "oic.if.rw" or "oic.if.a" OCF Interface) then an AllJoyn
642 property set operation (i.e., an org.freedesktop.DBus.Properties.Set() method call) shall be
643 mapped to a Partial UPDATE (e.g., POST in CoAP) with the corresponding OCF property.
- 644 – If a Resource has a Resource Type "oic.r.alljoynobject", then instead of separately translating
645 each of the Resources in the collection to its own AllJoyn object, all Resources in the collection
646 shall instead be translated to a single AllJoyn object whose object path is the OCF URI path of
647 the collection.

648 OCF property types shall be mapped to AllJoyn data types according to 4.3.

649 If an OCF operation fails, the Bridging Function shall send an appropriate AllJoyn error response
650 to the AllJoyn consumer. If an error message is present in the OCF response, and the error
651 message (e.g., diagnostic payload if using CoAP) fits the pattern "<error name>: <error message>"
652 where <error name> conforms to the AllJoyn error name syntax requirements, the AllJoyn error
653 name and AllJoyn error message shall be extracted from the error message in the OCF response.
654 Otherwise, the AllJoyn error name shall be "org.openconnectivity.Error.Code<#>" where <#> is the
655 error code (e.g., CoAP error code) in the OCF response without a decimal (e.g., "404") and the
656 AllJoyn error message is the error message in the OCF response.

657 6.2.5.2 Exposing an OCF server as a Virtual AllJoyn Producer

658 The object description returned in the About interface shall be formed as specified in the AllJoyn
659 About Interface Specification, and Table 8 shows how AllJoyn About Interface fields shall be
660 derived, based on properties in "oic.wk.d", "oic.wk.con", "oic.wk.p", and "oic.wk.con.p".

661

Table 8 – AllJoyn about data fields

To AJ Field name	AJ Description	AJ Mandatory	From OCF Property title	OCF Property name	OCF Description	OCF Mandatory
Appld	A 128-bit globally unique identifier for the application. The Appld shall be a universally unique identifier as specified in RFC 4122.	Y	Device ID (no exact equivalent exists)	di	Unique identifier for OCF Device (UUID)	Y
DefaultLanguage	The default language supported by the device. Specified as an IETF language tag listed in RFC 5646.	Y	Default Language	dl	The default language supported by the Device, specified as an RFC 5646 language tag. By default, clients can treat any string property as being in this language unless the property specifies otherwise. If absent, the Bridge Platform shall return a	N

					constant, e.g., empty string	
DeviceName (per supported language)	Name of the device set by platform-specific means (such as Linux and Android).	N	Platform Names	mnpn	Friendly name of the Platform. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the platform friendly name in the indicated language. For example, [{"language": "en", "value": "Dave's Laptop"}]	N
DeviceId	Device identifier set by platform-specific means.	Y	Platform ID	pi	Platform Identifier	Y
AppName (per supported language)	Application name assigned by the app manufacturer (developer or the OEM).	Y	Localized Names, if it exists, else (Device) Name	ln or n	Human-friendly name of the Device, in one or more languages. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the device name in the indicated language. If this property and the Device Name (n) property are both supported, the Device Name (n) value shall be included in this array.	N (ln), Y (n)
Manufacturer (per supported language)	The manufacturer's name of the app.	Y	Manufacturer Name	dmn	Name of manufacturer of the Device, in one or more languages. This property is an array of objects where each object has a "language" field (containing an RFC 5646	N

					language tag) and a "value" field containing the manufacturer name in the indicated language.	
ModelNumber	The app model number.	Y	Model Number	dmno	Model number as designated by manufacturer	N
SupportedLanguages	List of supported languages.	Y	language fields of Localized Names	ln	If ln is supported, return the list of values of the language field of each array element, else return empty array	N
Description (per supported language)	Detailed description expressed in language tags as in RFC 5646.	Y	Localized Descriptions	ld	Detailed description of the Device, in one or more languages. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the device description in the indicated language.	N
DateOfManufacture	Date of manufacture using format YYYY-MM-DD (known as XML DateTime format).	N	Date of Manufacture	mndt	Manufacturing date of device	N
SoftwareVersion	Software version of the app.	Y	Software Version	sv	Software version of the device.	N
AJSoftwareVersion	Current version of the AllJoyn SDK used by the application.	Y	(none)		Bridge Platform should return its own value	
HardwareVersion	Hardware version of the device on which the app is running.	N	Hardware Version	mnhw	Version of platform hardware	N
SupportUrl	Support URL (populated by the manufacturer)	N	Support URL	mnsI	URL that points to support information from manufacturer	N

org.openconnectivity.mnml		N	Manufacturer Details Link (URL)	mnml (if it exists, else field shall be absent)	URL to manufacturer (not to exceed 32 characters)	N
org.openconnectivity.mnpv		N	Platform Version	mnpv (if it exists, else field shall be absent)	Version of platform – string (defined by manufacturer)	N
org.openconnectivity.mnos		N	OS Version	mnos (if it exists, else field shall be absent)	Version of platform resident OS – string (defined by manufacturer)	N
org.openconnectivity.mnfv		N	Firmware version	mnfv (if it exists, else field shall be absent)	Version of device firmware	N
org.openconnectivity.st		N	SystemTime	st (if it exists, else field shall be absent)	Reference time for the device	N
org.openconnectivity.piid		N	Protocol-Independent ID	piid	A unique and immutable Device identifier. A Client can detect that a single Device supports multiple communication protocols if it discovers that the Device uses a single Protocol Independent ID value for all the protocols it supports.	Y

662

663 The AllJoyn field "org.openconnectivity.piid" shall be announced but shall not be localized and its
664 D-Bus type signature shall be "s". All other AllJoyn field names listed in Table 5 which have the
665 prefix "org.openconnectivity." shall be neither announced nor localized and their D-Bus type
666 signature shall be "s".

667 In addition, any additional vendor-defined properties in the OCF Device resource "/oic/d" (which
668 implements the "oic.wk.d" resource type) and the OCF Platform resource "/oic/p" (which
669 implements the "oic.wk.p" resource type) shall be mapped to vendor-defined fields in the AllJoyn
670 About data, with a field name formed by removing the leading "x." from the property name.

671 Table 9 shows how AllJoyn Configuration Interface fields shall be derived, based on properties in
672 "oic.wk.con" and "oic.wk.con.p".

Table 9 – AllJoyn configuration data fields

To AJ Field name	AJ Description	AJ Mandatory	From OCF Property title	OCF Property name	OCF Description	OCF Mandatory
DefaultLanguage	Default language supported by the device.	N	Default Language	dl	The default language supported by the Device, specified as an RFC 5646 language tag. By default, clients can treat any string property as being in this language unless the property specifies otherwise.	N
DeviceName	Device name assigned by the user. The device name appears on the UI as the friendly name of the device.	N	PlatformNames	mnpn	Friendly name of the Platform. This property is an array of objects where each object has a "language" field (containing an RFC 5646 language tag) and a "value" field containing the platform friendly name in the indicated language. For example, [{"language": "en", "value": "Dave's Laptop"}]	N
org.openconnectivity.loc		N	Location	loc (if it exists, else field shall be absent)	Provides location information where available.	N
org.openconnectivity.locn		N	Location Name	locn (if it exists, else field shall be absent)	Human friendly name for location For example, "Living Room".	N
org.openconnectivity.c		N	Currency	c (if it exists, else field shall be absent)	Indicates the currency that is used for any monetary transactions	N
org.openconnectivity.r		N	Region	r (if it exists, else field)	Free form text Indicating the current region in which the device is located	N

				shall be absent)	geographically. The free form text shall not start with a quote (").	
--	--	--	--	------------------	--	--

674
675 The AllJoyn field "org.openconnectivity.loc" shall be neither announced nor localized and its D-Bus
676 type signature shall be "ad". All other AllJoyn field names listed in Table 5 which have the prefix
677 "org.openconnectivity." shall be neither announced nor localized and their D-Bus type signature
678 shall be "s".

679 In addition, the Configuration methods FactoryReset and Restart shall be mapped to "oic.wk.mnt"
680 properties Factory_Reset ("fr") and Reboot ("rb"), respectively, and any additional vendor-defined
681 properties in the OCF Configuration resource (which implements the "oic.wk.con" resource type
682 and optionally the "oic.wk.con.p" resource type) shall be mapped to vendor-defined fields the
683 AllJoyn Configuration data, with a field name formed by removing the leading "x." from the property
684 name.

685 **6.2.6 Security**

686 For AllJoyn bridging, an OCF Onboarding Tool shall be able to block the communication of all OCF
687 Devices with all Bridged Devices that don't communicate securely with the Bridge, by using the
688 Bridge Device's "oic.r.securemode" Resource.

689 **6.3 On-the-Fly Translation from D-Bus and OCF payloads**

690 **6.3.1 Introduction**

691 The "dbus1" payload format is specified in the D-Bus Specification and AllJoyn adopted the D-Bus
692 protocol and made it distributed over the network. The modifications done by AllJoyn to the format
693 are all in the header part of the packet, not in the data payload itself, which remains compatible
694 with "dbus1". Other variants of the protocol that have been proposed by the Linux community
695 ("GVariant" and "kdbus" payloads) contain slight incompatibilities and are not relevant for this
696 discussion.

697 **6.3.2 Translation without aid of introspection**

698 **6.3.2.1 Introduction**

699 Clause 4.3.2 describes how Bridging Functions shall translate messages between the two payload
700 formats in the absence of introspection metadata from the actual device. This situation arises in
701 the when there is content not described by introspection, such as the inner payload of AllJoyn
702 properties of type "D-Bus VARIANT".

703 Since introspection is not available, the Bridging Function cannot know the rich JSON sub-type,
704 only the underlying CBOR type and from that it can infer the JSON generic type, and hence
705 translation is specified in terms of those generic types.

706 **6.3.2.2 Booleans**

707 Boolean conversion is trivial since both sides support this type.

708 **Table 10 – Boolean translation**

D-Bus type	JSON type
"b" – BOOLEAN	boolean (true or false)

709

710 **6.3.2.3 Numeric types**

711 The translation of numeric types is lossy and that is unavoidable due to the limited expressiveness
712 of the JSON generic types. This can only be solved with introspection.

713 The translation of numeric types is direction-specific.

714 **Table 11 – Numeric type translation, D-Bus to JSON**

From D-Bus type	To JSON type
"y" - BYTE (unsigned 8-bit)	Number
"n" - UINT16 (unsigned 16-bit)	
"u" - UINT32 (unsigned 32-bit)	
"t" - UINT64 (unsigned 64-bit) ^a	
"q" - INT16 (signed 16-bit)	
"" - INT32 (signed 32-bit)	
"x" - INT64 (signed 64-bit) ^a	
"d" - DOUBLE (IEEE 754 double precision)	
^a D-Bus payloads of types "t" (UINT64) and "x" (INT64) can contain values that cannot be perfectly represented in IEEE 754 double-precision floating point. The RFCs governing JSON do not forbid such numbers but caution that many implementations may not be able to deal with them. Currently, OCF transports its payload using CBOR instead of JSON, which can represent those numbers with fidelity. However, it should be noted that ISO/IEC 30118-1:2018 does not allow for integral numbers outside the range $-2^{53} \leq x \leq 2^{53}$.	

715

716 **Table 12 – Numeric type translation, JSON to D-Bus**

From JSON type	To D-Bus type
number	"d" - DOUBLE ^a
^a To provide the most predictable result, all translations from OCF to AllJoyn produce values of type "d" DOUBLE (IEEE 754 double precision).	

717

718

719 **6.3.2.4 Text strings**

720 **Table 13 – Text string translation**

D-Bus type	JSON type
"s" – STRING	string

721 Conversion between D-Bus and JSON strings is simple, as both require their content to be valid
722 Unicode. For example, an implementation can typically do a direct byte copy, as both protocols
723 specify UTF-8 as the encoding of the data, neither constrains the data to a given normalisation
724 format nor specify whether private-use characters or non-characters should be disallowed.

725 Since the length of D-Bus strings is always known, it is recommended Bridging Functions not use
726 CBOR indeterminate text strings (first byte 0x7f).

727 **6.3.2.5 Byte arrays**

728 The translation of a byte array is direction-specific.

729 **Table 14 – Byte array translation**

From D-Bus type	To JSON type
"ay" - ARRAY of BYTE	(base64-encoded) string

730 The base64url encoding is specified in IETF RF 4648 clause 5.

731 **6.3.2.6 D-Bus variants**

732 **Table 15 – D-Bus variant translation**

D-Bus type	JSON type
"v" – VARIANT	see clause 4.3.2.6

733

734 D-Bus has a type called VARIANT ("v") that is a wrapper around any other D-Bus type. It's a way
 735 for the type system to perform type-erasure. JSON, on the other hand, is not type-safe, which
 736 means that all JSON values are, technically, variants. The conversion for a D-Bus variant to JSON
 737 is performed by entering that variant and encoding the type carried inside as per the rules in this
 738 document.

739 The algorithm must be recursive, as D-Bus variants are allowed to contain variants themselves.

740 **6.3.2.7 D-Bus object paths and signatures**

741 The translation of D-Bus object paths and signatures is unidirectional (there is no mapping *to* them,
 742 only *from* them). This is shown in Table 16. In the reverse direction, clause 4.3.2.4 always converts
 743 to D-Bus STRING rather than OBJECT_PATH or SIGNATURE since it is assumed that "s" is the
 744 most common string type in use.

745 **Table 16 – D-Bus object path translation**

From D-Bus type	To JSON type
"o" - OBJECT_PATH	string
"g" – SIGNATURE	

746

747 Both D-Bus object paths and D-Bus type signatures are US-ASCII strings with specific formation
 748 rules, found in the D-Bus Specification. They are very seldom used and are not expected to be
 749 found in resources subject to translation without the aid of introspection.

750 **6.3.2.8 D-Bus structures**

751 The translation of the types in Table 17 is direction-specific:

752 **Table 17 – D-Bus structure translation**

From D-Bus type	To JSON type
"r" – STRUCT	array, length > 0

753

754 D-Bus structures can be interpreted as a fixed-length array containing a pre-determined list of types
 755 for each member. This is how such a structure is mapped to JSON: as an array of heterogeneous
 756 content, which are the exact members of the D-Bus structure, in the order in which they appear in
 757 the structure.

758 **6.3.2.9 Arrays**

759 The translation of the types in Table 18 is bidirectional:

760 **Table 18 – Byte array translation**

D-Bus type	JSON type
"ay" - ARRAY of BYTE	(base64-encoded) string – see 4.3.2.5
"ae" - ARRAY of DICT_ENTRY	object – see 4.3.2.10

761

762

763 The translation of the types in Table 19 is direction-specific:

764 **Table 19 – Other array translation**

From D-Bus type	To JSON type
"a" – ARRAY of anything else not specified	array

765

766 Aside from arrays of bytes and arrays of dictionary entries, which are mapped to JSON strings and
 767 objects respectively, arrays in JSON cannot be constrained to a single type (i.e., heterogeneous
 768 arrays). For that reason, strictly speaking all D-Bus arrays excepting arrays of bytes and arrays of
 769 dictionary entries must first be converted to arrays of variant "av" and then that array can be
 770 converted to JSON. See Table 20.

771 **Table 20 – JSON array translation**

From JSON type	Condition	To D-Bus type
array	length=0	"av" – ARRAY of VARIANT
array	length>0, all elements of same type	"a" – ARRAY
array	length>0, elements of different types	"r" – STRUCT

772 Conversion of D-Bus arrays of variants uses the conversion of variants as specified, which simply
 773 eliminates the distinction between a variant containing a given value and that value outside a
 774 variant. In other words, the elements of a D-Bus array are extracted and sent as elements of the
 775 JSON array, as per the other rules of this document.

776 **6.3.2.10 Dictionaries / Objects**

777 The choice of "dictionary of STRING to VARIANT" is made because that is the most common type
 778 of dictionary found in payloads and is an almost perfect superset of all possible dictionaries in D-
 779 Bus anyway. Moreover, it can represent JSON Objects with fidelity, which is the representation that
 780 OCF uses in its data models, which in turn means those D-Bus dictionaries will be able to carry
 781 with fidelity any OCF JSON Object in current use. See Table 21

782

Table 21 – D-Bus dictionary translation

D-Bus type	JSON type
"a{sv}" - dictionary of STRING to VARIANT	object

783 D-Bus dictionaries that are not mapping string to variant are first converted to those constraints
784 and then encoded in CBOR.

785 **6.3.2.11 Non-translatable types**

786 The types in Table 22 are not translatable, and the Bridging Function should drop the incoming
787 message. None of the types in Table 22 are in current use by either AllJoyn or OCF 1.0 devices,
788 so the inability to translate them should not be a problem.

789 **Table 22 – Non-translation types**

Type Scope	Type Name	Description
D-Bus	"h"	UNIX_FD (Unix File Descriptor)
JSON	Null	
JSON	undefined	Not officially valid JSON, but some implementations permit it

790

791 **6.3.2.12 Examples**

792 Table 23 and Table 24 provide some translation examples.

Table 23 – D-Bus to JSON translation examples

Source D-Bus	JSON Result
BOOLEAN(FALSE)	false
BOOLEAN(TRUE)	true
VARIANT(BOOLEAN(FALSE))	false
VARIANT(BOOLEAN(TRUE))	true
BYTE(0)	0.0
BYTE(255)	255.0
INT16(0)	0.0
INT16(-1)	-1.0
INT16(-32768)	-32768.0
UINT16(0)	0.0
UINT16(65535)	65535.0
INT32(0)	0.0
INT32(-2147483648)	-2147483648.0
INT32(2147483647)	2147483647.0
UINT32(0)	0.0
UINT32(4294967295)	4294967295.0
INT64(0)	0.0
INT64(-1)	-1.0
UINT64(18446744073709551615)	18446744073709551615.0 ⁽¹⁾
DOUBLE(0.0)	0.0
DOUBLE(0.5)	0.5
STRING("")	""
STRING("Hello")	"Hello"
ARRAY<BYTE>()	""
ARRAY<BYTE>(0x48, 0x65, 0x6c, 0x6c, 0x6f)	"SGVsbG8"
OBJECT_PATH("/")	"/"
SIGNATURE()	""
SIGNATURE("s")	"s"
VARIANT(INT32(0))	0
VARIANT(VARIANT(INT32(0)))	0
VARIANT(STRING("Hello"))	"Hello"

Table 24 – JSON to D-Bus translation examples

Source JSON	D-Bus Result
false	BOOLEAN(false)
true	BOOLEAN(true)
0	DOUBLE(0.0)
-1	DOUBLE(-1.0)
-2147483648	DOUBLE(-2147483648.0)
2147483647	DOUBLE(2147483647.0)
2147483648	DOUBLE(2147483648.0)
-2147483649	DOUBLE(-2147483649.0)
9223372036854775808 ⁽¹⁾	DOUBLE(9223372036854775808.0)
0.0	DOUBLE(0.0)
0.5	DOUBLE(0.5)
0.0f	DOUBLE(0.0)
0.5f	DOUBLE(0.5)
""	STRING("")
"Hello"	STRING("Hello")
[]	ARRAY<VARIANT>()
[1]	ARRAY<DOUBLE>(DOUBLE(1.0))
[1, 2147483648, false, "Hello"]	STRUCT<DOUBLE, DOUBLE, BOOLEAN, STRING>(DOUBLE(1.0), DOUBLE(2147483648.0), BOOLEAN(false), STRING("Hello"))
{}	map<STRING, VARIANT>()
{1: 1}	map<STRING, VARIANT>("1" → VARIANT(DOUBLE(1.0)))
{"1": 1}	map<STRING, VARIANT>("1" → VARIANT(DOUBLE(1.0)))
{ "rep": { "state": false, "power": 1.0, "name": "My Light" } }	map<STRING, VARIANT>({STRING("rep"), VARIANT(map<STRING, VARIANT>({STRING("state") → VARIANT(BOOLEAN(FALSE))}, {STRING("power") → VARIANT(DOUBLE(1.0))}, {STRING("name") → VARIANT(STRING("My Light"))}))))

796 NOTE This value cannot be represented with IEEE754 double-precision floating point without loss of information. It is
797 also outside the currently-allowed range of integrals in OCF.

798 6.3.3 Translation with aid of introspection

799 6.3.3.1 Introduction to Introspection Metadata

800 When introspection is available, the Bridging Function can use the extra metadata provided by the
801 side offering the service to expose a higher-quality reply to the other side. This chapter details
802 modifications to the translation described in the previous chapter when the metadata is found.

803 Introspection metadata can be used for both translating requests to services and replies from those
804 services. When used to translate requests, the introspection is "constraining", since the Bridging
805 Function must conform exactly to what that service expects. When used to translate replies, the
806 introspection is "relaxing", but may be used to inform the receiver what other possible values may
807 be encountered in the future.

808 Note that OCF introspection uses JSON types, media attributes, and format attributes, not CBOR
809 encoding. The actual encoding of each JSON type is discussed in clause 12.4 of ISO/IEC 30118-
810 1:2018 , JSON format attribute values are as defined in ISO/IEC 30118-1:2018, and JSON media
811 attribute values are as defined in JSON Hyper-Schema.

812 **6.3.3.2 Translation of the introspection itself**

813 Note that both OCF 1.0 and AllJoyn require all services exposed to include introspection metadata,
814 which means the Bridging Function will need to translate the introspection information on-the-fly
815 for each OCF resource or AllJoyn producer it finds. The Bridging Function shall preserve as much
816 of the original information as can be represented in the translated format. This includes both the
817 information used in machine interactions and the information used in user interactions, such as
818 description and documentation text.

819 **6.3.3.3 Variability of introspection data**

820 Introspection data is not a constant and the Bridging Function may find, upon discovering further
821 services, that the D-Bus interface or OCF Resource Type it had previously encountered is different
822 than previously seen. The Bridging Function needs to take care about how the destination side will
823 react to a change in introspection.

824 D-Bus interfaces used by AllJoyn services may be updated to newer versions, which means a given
825 type of service may be offered by two distinct versions of the same interface. Updates to
826 standardised interfaces must follow strict guidelines established by the AllSeen Interface Review
827 Board, mapping each version to a different OCF Resource Type should be possible without much
828 difficulty. However, there's no guarantee that vendor-specific extensions follow those requirements.
829 Indeed, there's nothing preventing two revisions of a product to contain completely incompatible
830 interfaces that have the same name and version number.

831 On the opposite direction, the rules are much laxer. Since OCF specifies optional properties to its
832 Resource Types, a simple monotonically-increasing version number like AllJoyn consumer
833 applications expect is not possible.

834 However, it should be noted that services created by the Bridging Function by "on-the-fly"
835 translation will only be accessed by generic client applications. Dedicated applications will only use
836 "deep binding" translation.

837 **6.3.3.4 Numeric types**

838 For numeric values, all D-Bus and JSON numeric types are treated equally as source and may all
839 be translated into any of the other side's types. When translating a request to a service, the Bridging
840 Function need only verify whether there would be loss of information when translating from source
841 to destination. For example, when translating the number 1.5 to either a JSON integer or to one of
842 the D-Bus integral types, there would be loss of information, in which case the Bridging Function
843 should refuse the incoming message. Similarly, the value 1,234,567 does not fit the range of a D-
844 Bus byte, 16-bit signed or unsigned integer.

845 When translating the reply from the service, the Bridging Function shall use the following rules.

846 Table 25 indicates how to translate from a JSON type to the corresponding D-Bus type, where the
847 first matching row shall be used. If the JSON schema does not indicate the minimum value of a
848 JSON integer, 0 is the default. If the JSON schema does not indicate the maximum value of a JSON
849 integer, $2^{32} - 1$ is the default. The resulting AllJoyn introspection XML shall contain

850 "org.alljoyn.Bus.Type.Min" and "org.alljoyn.Bus.Type.Max" annotations whenever the minimum or
 851 maximum, respectively, of the JSON value is different from the natural minimum or maximum of
 852 the D-Bus type.

853 **Table 25 – JSON type to D-Bus type translation**

From JSON type	Condition	To D-Bus Type
integer	minimum ≥ 0 AND maximum $< 2^8$	"y" (BYTE)
	minimum ≥ 0 AND maximum $< 2^{16}$	"q" (UINT16)
	minimum $\geq -2^{15}$ AND maximum $< 2^{15}$	"n" (INT16)
	minimum ≥ 0 AND maximum $< 2^{32}$	"u" (UINT32)
	minimum $\geq -2^{31}$ AND maximum $< 2^{31}$	"i" (INT32)
	minimum ≥ 0	"t" (UINT64)
		"x" (INT64)
Number		"d" (DOUBLE)
String	pattern = <code>"^0 ([1-9][0-9]{0,19})\$"</code>	"t" (UINT64)
	pattern = <code>"^0 (-?[1-9][0-9]{0,18})\$"</code>	"x" (INT64)

854 Table 26 indicates how to translate from a D-Bus type to the corresponding JSON type.

855 **Table 26 – D-Bus type to JSON type translation**

From D-Bus type	To JSON type	Note
"y" (BYTE)	integer	"minimum" and "maximum" in the JSON schema shall be set to the value of the "org.alljoyn.Bus.Type.Min" and "org.alljoyn.Bus.Type.Max" (respectively) annotations if present, or to the min and max values of the D-Bus type's range if such annotations are absent.
"n" (UINT16)		
"q" (INT16)		
"u" (UINT32)		
"i" (INT32)		
"t" (UINT64)	integer if org.alljoyn.Bus.Type.Max $\leq 2^{53}$, else string with JSON pattern attribute <code>"^0 ([1-9][0-9]{0,19})\$"</code> .	IETF RFC 7159 clause 6 explains that higher JSON integers are not interoperable.
"x" (INT64)	integer (if org.alljoyn.Bus.Type.Min $\geq -2^{53}$ AND org.alljoyn.Bus.Type.Max $\leq 2^{53}$), else string with JSON pattern attribute <code>"^0 (-?[1-9][0-9]{0,18})\$"</code> .	IETF RFC 7159 clause 6 explains that other JSON integers are not interoperable.
"d" (double)	number	

856

857

858 **6.3.3.5 Text string and byte arrays**

859 There's no difference in the translation of text strings and byte arrays compared to clause 4.3.2.
 860 Clause 4.3.3 simply lists the JSON equivalent types for the generated OCF introspection. See
 861 Table 27.

862

Table 27 – Text string translation

D-Bus Type	JSON type	JSON media attribute, binaryEncoding property
"s" – STRING	string	(none)
"ay" - ARRAY of BYTE	string	base64

863 In addition, the mapping of the JSON Types in Table 28 is direction-specific:

864

Table 28 – JSON UUID string translation

From JSON type	Condition	To D-Bus Type
string	pattern = "[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}\$"	"ay" – ARRAY of BYTE

865 JSON strings with any other format value (e.g., date-time, uri, etc.) or pattern value not shown in
 866 Table 28 shall be treated the same as if the format and pattern attributes were absent, by simply
 867 mapping the value to a D-Bus string.

868 **6.3.3.6 D-Bus Variants**

869 If the introspection of an AllJoyn producer indicates a value in a request should be a D-Bus
 870 VARIANT, the Bridging Function should create such a variant and encode the incoming value as
 871 the variant’s payload as per the rules in the rest of this document. See Table 29.

872

Table 29 – D-Bus variant translation

D-Bus Type	JSON Type
"v" – VARIANT	see clause 4.3.3.6

873 **6.3.3.7 D-Bus Object Paths and Signatures**

874 If the introspection of an AllJoyn producer indicates a value in a request should be a D-Bus Object
 875 Path or D-Bus Signature, the Bridging Function should perform a validity check in the incoming
 876 CBOR Text String. If the incoming data fails to pass this check, the message should be rejected.
 877 See Table 30.

878

Table 30 – D-Bus object path translation

From D-Bus Type	To JSON Type
"o" – OBJECT_PATH	string
"g" – SIGNATURE	

879 **6.3.3.8 D-Bus structures**

880 D-Bus structure members are described in the introspection XML with the
 881 "org.alljoyn.Bus.Struct.*StructureName*.Field.*fieldName*.Type" annotation. The Bridging Function
 882 shall use the AJSoftwareVersion field of the About data obtained from a bridged AllJoyn producer
 883 as follows. When the version of AllJoyn implemented on the Bridged Device is v16.10.00 or greater
 884 and the member annotations are present, the Bridging Function shall use a JSON object to
 885 represent a structure, mapping each member to the entry with that name. The Bridging Function
 886 needs to be aware that the incoming CBOR payload may have changed the order of the fields,
 887 when compared to the D-Bus structure. When the version of AllJoyn implemented on the Bridged
 888 Device is less than v16.10.00, the Bridging Function shall follow the rule for translating D-Bus
 889 structures without the aid of introspection data.

890 **6.3.3.9 Arrays and dictionaries**

891 If the introspection of the AllJoyn interface indicates that the array is neither an ARRAY of BYTE
 892 ("ay") nor an ARRAY of VARIANT ("av") or that the dictionary is not mapping STRING to VARIANT
 893 ("a{sv}"), the Bridging Function shall apply the constraining or relaxing rules specified in other
 894 clauses.

895 Similarly, if the OCF introspection indicates a homogeneous array type, the information about the
 896 array's element type should be used as the D-Bus array type instead of VARIANT ("v").

897 **6.3.3.10 Other JSON format attribute values**

898 The JSON format attribute may include other custom attribute types. They are not known at this
 899 time, but it is expected that those types be handled by their type and representation alone.

900 **6.3.3.11 Examples**

901 Table 31 and Table 32 provide examples using introspection.

902 **Table 31 – Mapping from AllJoyn using introspection**

AllJoyn Source	AllJoyn Introspection Notes	Translated JSON Payload	OCF Introspection Notes
UINT32 (0)		0	JSON schema should indicate: "type": "integer", "minimum": 0, "maximum": 4294967295
INT64 (0)		0	Since no Min/Max annotations exist in AllJoyn, JSON schema should indicate: "type": "string", "pattern": "^0 (-?[1-9][0-9]{0,18})\$"
UINT64 (0)		"0"	Since no Max annotation exists in AllJoyn, JSON schema should indicate: "type": "string", "pattern": "^0 ([1-9][0-9]{0,19})\$"
STRING("Hello")		"Hello"	JSON schema should indicate: "type": "string"
OBJECT_PATH("/")		"/"	JSON schema should indicate: "type": "string"
SIGNATURE("g")		"g"	JSON schema should indicate: "type": "string"
ARRAY<BYTE>(0x48, 0x65, 0x6c, 0x6c, 0x6f)		"SGVsbG8"	JSON schema should indicate: "type": "string", "media binaryEncoding": "base64"
VARIANT(<i>anything</i>)		?	JSON schema should indicate: "type": ["boolean", "object", "array", "number", "string", "integer"]
ARRAY<INT32>()		[]	JSON schema should indicate: "type": "array", "items": { "type": "integer" }

ARRAY<INT64>()		[]	JSON schema should indicate: "type": "array", "items": { "type": "string", "pattern": "^0 ([1-9][0-9]{0,18})\$" }
STRUCT< INT32, INT32>(0, 1)	AllJoyn introspection specifies the argument with the annotation: <struct name="Point"> <field name="x" type="i" /> <field name="y" type="i" /> </struct>	{"x": 0, "y": 1}	JSON schema should indicate: "type": "object", "properties": { "x": { "type": "integer" }, "y": { "type": "integer" } }

903

904

Table 32 – Mapping from CBOR using introspection

CBOR Payload	OCF Introspection Notes	Translated AllJoyn	AllJoyn Introspection Notes
0	"type": "integer"	INT32(0)	
0	"type": "integer", "minimum": -240, "maximum": 240	INT64(0)	org.alljoyn.Bus.Type.Min = -240 org.alljoyn.Bus.Type.Max = 240
0	"type": "integer", "minimum": 0, "maximum": 248	UINT64(0)	org.alljoyn.Bus.Type.Max = 248
0.0	"type": "number"	DOUBLE(0.0)	
[1]	JSON schema indicates: "type": "array", "items": { "type": "integer", "minimum": 0, "maximum": 246 }	ARRAY<UINT64>(1)	org.alljoyn.Bus.Type.Max = 246

905

7 Device type mapping

7.1 AllJoyn device types to OCF device types

Table 33 captures the equivalency mapping between AllJoyn defined Device Types (see AllJoyn Common Data Model Interface Definitions) and OCF defined Device Types (see Table 10-1 in ISO/IEC 30118-5:2019). The minimum interface set for the AllJoyn definitions is provided in the HAE Theory of Operation; the minimum Resource sets for each OCF Device is provided in ISO/IEC 30118-5:2019.

912

Table 33 – AllJoyn to OCF device type mapping

Classification	AllJoyn Device Type	AllJoyn ID	OCF Device Type
Air Care	Air Conditioner	5	oic.d.airconditioner

	Air Purifier	9	oic.d.airpurifier
	Air Quality Monitor	11	oic.d.airqualitymonitor
	Dehumidifier	8	oic.d.dehumidifier
	Humidifier	7	oic.d.humidifier
	Electric Fan	10	oic.d.fan
	Thermostat	6	oic.d.thermostat
Fabric Care	Clothes Washer	12	oic.d.washer
	Clothes Dryer	13	oic.d.dryer
	Clothes Washer-Dryer	14	oic.d.washerdryer
Food Preservation	Refrigerator	2	oic.d.refrigerator
	Ice-Maker	4	oic.r.icemaker (maps to Resource)
	Freezer	3	oic.d.freezer
Food Preparation	Oven	17	oic.d.oven
	Cooktop	18	oic.d.cooktop
	Cookerhood	19	oic.d.cookerhood
	Food probe	20	oic.d.foodprobe
Dish Care	Dishwasher	15	oic.d.dishwasher
Floor Care	Robot Cleaner	16	oic.d.robotcleaner
Entertainment	Television	21	oic.d.tv
	Set Top Box (STB)	22	oic.d.stb

914 **7.2 OCF device types with no AllJoyn equivalent**

915 Table 34 captures the Device Types defined by OCF have no direct equivalent in AllJoyn, they
916 shall all be mapped to an AllJoyn Device Type of "Other" (Id of "1").

917 **Table 34 – OCF device types with no AllJoyn equivalent**

OCF Device Name	OCF Device Type
Receiver	oic.d.receiver
Blind	oic.d.blind
Door	oic.d.door
Garage Door	oic.d.garagedoor
Generic Sensor	oic.d.sensor
Light	oic.d.light
Smart Plug	oic.d.smartplug
Switch	oic.d.switch
Water Valve	oic.d.watervalve
Printer	oic.d.printer
Multi-Function Printer	oic.d.multifunctionprinter
Scanner	oic.r.scanner
Camera	oic.d.camera
Security Panel	oic.d.securitypanel

Smart Lock	oic.d.smartlock
------------	-----------------

918 **8 Resource to interface equivalence**

919 **8.1 Introduction**

920 Clause 6 captures the equivalency mapping between AllJoyn defined Interfaces (see AllJoyn
921 Common Data Model Interface Definitions) and OCF defined Resource Types (see ISO/IEC 30118-
922 4:2018). Detailed Property by Property mappings are provided in clause 7.

923 Table 35 captures the mappings for Interfaces that are part of the minimum set for an AllJoyn
924 Device.

925 Table 36 captures the mappings for Interfaces that are optional for an AllJoyn Device; deep
926 translation for these interfaces via derived modelling is not within the scope of this release of the
927 document.

928 **Table 35 – AllJoyn interface to OCF resource type mapping – minimum interface set**

AllJoyn Interface	OCF Resource Type Name	OCF Resource Type ID	OCF Interface(s)
Environment.CurrentAirQuality	Air Quality Collection	oic.r.airqualitycollection	oic.if.s
Environment.CurrentAirQualityLevel	Air Quality Collection	oic.r.airqualitycollection	oic.if.s
Environment.CurrentHumidity	Humidity	oic.r.humidity	oic.if.s
Environment.CurrentTemperature	Temperature	oic.r.temperature	oic.if.s
Environment.TargetHumidity	Humidity	oic.r.humidity, oic.r.selectablelevels	oic.if.a
Environment.TargetTemperature	Temperature	oic.r.temperature	oic.if.a
Operation.AudioVolume	Audio Controls	oic.r.audio	oic.if.a
Operation.Channel	Not mapped		
Operation.ClimateControlMode	Mode	oic.r.mode	oic.if.a
	Operational State	oic.r.operational.state	oic.if.s
Operation.ClosedStatus	Door	oic.r.door	oic.if.s
Operation.CycleControl	Operational State	oic.r.operational.state	oic.if.s
Operation.FanSpeedLevel	Air Flow	oic.r.airflow	oic.if.a
Operation.HeatingZone	Heating Zone Collection	oic.r.heatingzonecollection	oic.if.s
Operation.HvacFanMode	Mode	oic.r.mode	oic.if.a
Operation.OnOffStatus	Binary Switch	oic.r.switch.binary	oic.if.s
Operation.OvenCyclePhase	Operational State	oic.r.operationalstate	oic.if.s

929

930 **Table 36 – AllJoyn interface to OCF resource type mapping – optional interface set**

AllJoyn Interface	OCF Resource Type Name	OCF Resource Type ID	OCF Interface(s)
Environment.TargetTemperatureLevel	Mode	oic.r.mode	oic.if.a
Environment.WaterLevel	TBD	TBD	oic.if.s

Environment.WindDirection	Air Flow	oic.r.airflow	oic.if.a
Operation.AirRecirculationMode	Mode	oic.r.mode	oic.if.a
Operation.Alerts	TBD	TBD	TBD
Operation.AudioVideoInput	Media Source List	oic.r.media.input	oic.if.a
Operation.BatteryStatus	Battery	oic.r.energy.battery	oic.if.s
Operation.CurrentPower	Energy Usage	oic.r.energy.usage	oic.if.s
Operation.DishWashingCyclePhase	Operational State	oic.r.operationalstate	oic.if.s
Operation.EnergyUsage	Energy Usage	oic.r.energy.usage	oic.if.s
Operation.FilterStatus	TBD	TBD	TBD
Operation.LaundryCyclePhase	Mode	oic.r.mode	oic.if.s
Operation.MoistureOutputLevel	Mode	oic.r.mode	oic.if.a
Operation.PlugInUnits	TBD	TBD	TBD
Operation.RapidMode	Refrigeration	oic.r.refrigeration	oic.if.a
Operation.RemoteControllability	TBD	TBD	TBD
Operation.RepeatMode	Ecomode	oic.r.ecomode	oic.if.a
Operation.ResourceSaving	TBD	TBD	TBD
Operation.RobotCleaningCyclePhase	Mode	oic.r.mode	oic.if.s
Operation.SoilLevel	Mode	oic.r.mode	oic.if.a
Operation.SpinSpeedLevel	Mode	oic.r.mode	oic.if.a
Operation.Timer	Time Period	oic.r.time.period	oic.if.s

931 **8.2 Environment.CurrentAirQuality mapping**

932 If more than one instance of the AirQuality interface is exposed, then each instance maps to an
933 instance of the OCF AirQuality Resource. The mapping defined in clause 7.2 describes the
934 population of the OCF AirQuality Resource. Even if there is only a single instance of an OCF
935 AirQuality Resource this shall be included in an instance of an OCF AirQualityCollection. The
936 number of links in the collection equates to the number of instances of the AllJoyn CurrentAirQuality
937 interface that are exposed. When mapping from OCF the valueType of the Resource shall be
938 introspected, this API is invoked only if this is set to "Measured".

939 **8.3 Environment.CurrentAirQualityLevel mapping**

940 If more than one instance of the AirQualityLevel interface is exposed, then each instance maps to
941 an instance of the OCF AirQuality Resource. The mapping defined in clause 7.2 describes the
942 population of the OCF AirQuality Resource. Even if there is only a single instance of an OCF
943 AirQuality Resource then this shall be included in an instance of an OCF AirQualityCollection. The
944 number of links in the collection equates to the number of instances of the AllJoyn CurrentAirQuality
945 interface that are exposed. When mapping from OCF the valueType of the Resource shall be
946 introspected, this API is invoked only if this is set to "Qualitative".

947 **8.4 Operation.ClimateControlMode mapping**

948 ClimateControlMode has three Properties; these map as follows: mode and supportedmodes maps
949 to the Mode Resource, operationalstate maps to the OperationalState Resource This can be
950 represented in OCF either as two distinct Resource instances or a single instance with two
951 Resource Types (oic.r.mode, oic.r.operationalstate).

952 **8.5 Operation.FanSpeedLevel mapping**

953 The setting of the FanSpeedLevel to "0x00" (off) is handled via the "OffControl" interface rather
954 than writing directly to this interface. In such a case an instance of Binary Switch shall be exposed

955 on the OCF side; this can be modelled as AirFlowControl which is then a collection of Binary Switch
 956 and AirFlow.

957 **8.6 Operation.HeatingZone mapping**

958 Each element in the array of heating zones within the AllJoyn HeatingZone interface maps to an
 959 instance of OCF HeatingZone, itself a link in an instance of an OCF HeatingZoneCollection. The
 960 mapping defined clause 7.13 describes the population of the OCF HeatingZone Resource that
 961 constitutes the Resources that are contained in the collection.

962 **8.7 Operation.OnOffStatus, Operation.OnControl, and Operation.OffControl mapping**

963 A discovered instance of a Binary Switch is always mapped to an Operation.OnOffStatus interface.
 964 A RETRIEVE on a Binary Switch maps to an action on an instance of an Operation.OnOffStatus
 965 Interface. An UPDATE on a Binary Switch maps to a method invocation on either
 966 Operation.OnControl or OffControl. value = true maps to Operation.OnControl value = false maps
 967 to Operation.OffControl.

968 **8.8 Operation.OvenCyclePhase**

969 OvenCyclePhase cyclephase Property pre-defines values 0x00-0x7F, 0x80-0xFF is for vendor
 970 specific values. The mapping defined in clause 7 covers only specification defined values. Any
 971 vendor defined value shall be represented in OCF using the x.<organization> syntax for a vendor
 972 defined Property.

973 **9 Detailed mapping APIs**

974 **9.1 Introduction**

975 This clause provides a mapping description (using JSON that aligns with the Derived Modelling
 976 syntax described in Derived Models for Interoperability between IoT Ecosystems for all Interfaces
 977 and Resources that are within scope.

978 The derived model definitions presented in clause 7 are formatted for readability, and so may
 979 appear to have extra line breaks.

980 Table 37 provides a reference and link to the per Interface clauses.

981 **Table 37 – Interface to resource summary**

AllJoyn Interface Name	Equivalent Resource(s)	Clause
Environment.CurrentAirQuality	oic.r.airqualitycollection	8.2
Environment.CurrentAirQualityLevel	oic.r.airqualitycollection	8.3
Environment.CurrentHumidity	oic.r.humidity	8.4
Environment.CurrentTemperature	oic.r.temperature	8.5
Environment.TargetHumidity	oic.r.humidity, oic.r.selectablelevels	8.6
Environment.TargetTemperature	oic.r.temperature	8.7
Operation.AudioVolume	oic.r.audio	8.8
Operation.ClimateControlMode	oic.r.mode, oic.r.operationalstate	8.9
Operation.ClosedStatus	oic.r.door	8.10
Operation.CycleControl	oic.r.operational.state	8.11
Operation.FanSpeedLevel	oic.r.airflow	8.12
Operation.HeatingZone	oic.r.heatingzonecollection	8.13

Operation.HvacFanMode	oic.r.mode	8.14
Operation.OnControl, Operation.OffControl	oic.r.switch.binary	8.15
Operation.OnOffStatus,	oic.r.switch.binary	8.16
Operation.OvenCyclePhase	oic.r.operationalstate	8.17

982 **9.2 Current Air Quality**

983 **9.2.1 Derived model**

984 The derived model: "asa.environment.currentairquality".

985 **9.2.2 Property definition**

986 Table 38 provides the detailed per Property mapping for "asa.environment.currentairquality".

987 **Table 38 – The property mapping for "asa.environment.currentairquality".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
minvalue	oic.r.airquality	range[0] = minvalue	minvalue = range[0]
maxvalue	oic.r.airquality	range[1] = maxvalue	maxvalue = range[1]
contaminanttype	oic.r.airquality	valuetype = Measuredcontaminanttypearray [CH2O,CO2,CO,PM2_5,PM10,VOC] ocf.contaminanttype = contaminanttypearray[contaminanttype]	contaminanttype = indexof contaminanttypearray[ocf.contaminanttype]
currentvalue	oic.r.airquality	contaminantvalue = currentvalue	currentvalue = contaminantvalue
updatemintime	oic.r.value.conditional	ocf.minnotifyperiod = updatemintime	updatemintime = ocf.minnotifyperiod
precision	oic.r.airquality	ocf.precision = precision	precision = ocf.precision

988 Table 39 provides the details of the Properties that are part of "asa.environment.currentairquality".

989 **Table 39 – The properties of "asa.environment.currentairquality".**

AllJoyn Property name	Property	Type	Required	Description
minvalue		number	yes	
maxvalue		number	yes	
contaminanttype		integer	yes	The contaminant type
currentvalue		number	yes	
updatemintime		integer	yes	
precision		number	yes	

990 **9.2.3 Derived model definition**

```

991 {
992   "id": "http://openinterconnect.org/asamapping/schemas/asa.environment.currentairquality.json#",
993   "$schema": "http://json-schema.org/draft-04/schema#",
994   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
995   "title": "Current Air Quality",
996   "definitions": {
997     "asa.environment.currentairquality": {
998       "type": "object",

```

```

999     "properties": {
1000       "contaminanttype": {
1001         "type": "integer",
1002         "description": "The contaminant type",
1003         "x-ocf-conversion": {
1004           "x-ocf-alias": "oic.r.airquality",
1005           "x-to-ocf": [
1006             "valuetype = Measured",
1007             "contaminanttypearray = [CH2O,CO2,CO,PM2_5,PM10,VOC]",
1008             "ocf.contaminanttype = contaminanttypearray[contaminanttype]"
1009           ],
1010           "x-from-ocf": [
1011             "contaminanttype = indexof contaminanttypearray[ocf.contaminanttype]"
1012           ]
1013         }
1014       },
1015       "currentvalue": {
1016         "type": "number",
1017         "x-ocf-conversion": {
1018           "x-ocf-alias": "oic.r.airquality",
1019           "x-to-ocf": [
1020             "contaminantvalue = currentvalue"
1021           ],
1022           "x-from-ocf": [
1023             "currentvalue = contaminantvalue"
1024           ]
1025         }
1026       },
1027       "minvalue": {
1028         "type": "number",
1029         "x-ocf-conversion": {
1030           "x-ocf-alias": "oic.r.airquality",
1031           "x-to-ocf": [
1032             "range[0] = minvalue"
1033           ],
1034           "x-from-ocf": [
1035             "minvalue = range[0]"
1036           ]
1037         }
1038       },
1039       "maxvalue": {
1040         "type": "number",
1041         "x-ocf-conversion": {
1042           "x-ocf-alias": "oic.r.airquality",
1043           "x-to-ocf": [
1044             "range[1] = maxvalue"
1045           ],
1046           "x-from-ocf": [
1047             "maxvalue = range[1]"
1048           ]
1049         }
1050       },
1051       "precision": {
1052         "type": "number",
1053         "x-ocf-conversion": {
1054           "x-ocf-alias": "oic.r.airquality",
1055           "x-to-ocf": [
1056             "ocf.precision = precision"
1057           ],
1058           "x-from-ocf": [
1059             "precision = ocf.precision"
1060           ]
1061         }
1062       },
1063       "updatemintime": {
1064         "type": "integer",
1065         "x-ocf-conversion": {
1066           "x-ocf-alias": "oic.r.value.conditional",
1067           "x-to-ocf": [
1068             "ocf.minnotifyperiod = updatemintime"
1069           ],

```

```

1070         "x-from-ocf": [
1071             "updatemintime = ocf.minnotifyperiod"
1072         ]
1073     }
1074 }
1075 }
1076 }
1077 },
1078 "type": "object",
1079 "allOf": [
1080     {"$ref": "#/definitions/asa.environment.currentairquality"}
1081 ],
1082 "required": ["contaminanttype", "currentvalue", "minvalue", "maxvalue", "precision", "updatemintime"]
1083 }
1084

```

1085 **9.3 Current Air Quality Level**

1086 **9.3.1 Derived model**

1087 The derived model: "asa.environment.currentairqualitylevel".

1088 **9.3.2 Property definition**

1089 Table 40 provides the detailed per Property mapping for "asa.environment.currentairqualitylevel".

1090 **Table 40 – The property mapping for "asa.environment.currentairqualitylevel".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
contaminanttype		oic.r.airquality	valuetype = Qualitativeif contaminanttype = 0, ocf.contaminanttype = CH2Oif contaminanttype = 1, ocf.contaminanttype = CO2if contaminanttype = 2, ocf.contaminanttype = COif contaminanttype = 3, ocf.contaminanttype = PM2_5if contaminanttype = 4, ocf.contaminanttype = PM10if contaminanttype = 5, ocf.contaminanttype = VOCif contaminanttype = 253, ocf.contaminanttype = Smokeif contaminanttype = 254, ocf.contaminanttype = Odorif contaminanttype = 255, ocf.contaminanttype = AirPollution	if ocf.contaminanttype = CH2O, contaminanttype = 0if ocf.contaminanttype = CO2, contaminanttype = 1if ocf.contaminanttype = CO, contaminanttype = 2if ocf.contaminanttype = PM2_5, contaminanttype = 3if ocf.contaminanttype = PM10, contaminanttype = 4if ocf.contaminanttype = VOC, contaminanttype = 5if ocf.contaminanttype = Smoke, contaminanttype = 253if ocf.contaminanttype = Odor, contaminanttype = 254if ocf.contaminanttype = AirPollution, contaminanttype = 255

maxlevel	oic.r.airquality	range[0] = Orange[1] = maxvalue	maxvalue = range[1]
currentlevel	oic.r.airquality	contaminantvalue = currentlevel	currentlevel = contaminantvalue

1091 Table 41 provides the details of the Properties that are part of
1092 "asa.environment.currentairqualitylevel".

1093 **Table 41 – The properties of "asa.environment.currentairqualitylevel".**

AllJoyn name	Property	Type	Required	Description
contaminanttype		integer	yes	The contaminant type
maxlevel		integer	yes	
currentlevel		integer	yes	

1094 9.3.3 Derived model definition

```

1095 {
1096   "id":
1097   "http://openinterconnect.org/asamapping/schemas/asa.environment.currentairqualitylevel.json#",
1098   "$schema": "http://json-schema.org/draft-04/schema#",
1099   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1100   "title": "Current Air Quality Level",
1101   "definitions": {
1102     "asa.environment.currentairqualitylevel": {
1103       "type": "object",
1104       "properties": {
1105         "contaminanttype": {
1106           "type": "integer",
1107           "description": "The contaminant type",
1108           "x-ocf-conversion": {
1109             "x-ocf-alias": "oic.r.airquality",
1110             "x-to-ocf": [
1111               "valuetype = Qualitative",
1112               "if contaminanttype = 0, ocf.contaminanttype = CH20",
1113               "if contaminanttype = 1, ocf.contaminanttype = CO2",
1114               "if contaminanttype = 2, ocf.contaminanttype = CO",
1115               "if contaminanttype = 3, ocf.contaminanttype = PM2_5",
1116               "if contaminanttype = 4, ocf.contaminanttype = PM10",
1117               "if contaminanttype = 5, ocf.contaminanttype = VOC",
1118               "if contaminanttype = 253, ocf.contaminanttype = Smoke",
1119               "if contaminanttype = 254, ocf.contaminanttype = Odor",
1120               "if contaminanttype = 255, ocf.contaminanttype = AirPollution"
1121             ],
1122             "x-from-ocf": [
1123               "if ocf.contaminanttype = CH20, contaminanttype = 0",
1124               "if ocf.contaminanttype = CO2, contaminanttype = 1",
1125               "if ocf.contaminanttype = CO, contaminanttype = 2",
1126               "if ocf.contaminanttype = PM2_5, contaminanttype = 3",
1127               "if ocf.contaminanttype = PM10, contaminanttype = 4",
1128               "if ocf.contaminanttype = VOC, contaminanttype = 5",
1129               "if ocf.contaminanttype = Smoke, contaminanttype = 253",
1130               "if ocf.contaminanttype = Odor, contaminanttype = 254",
1131               "if ocf.contaminanttype = AirPollution, contaminanttype = 255"
1132             ]
1133           }
1134         },
1135         "currentlevel": {
1136           "type": "integer",
1137           "x-ocf-conversion": {
1138             "x-ocf-alias": "oic.r.airquality",
1139             "x-to-ocf": [
1140               "contaminantvalue = currentlevel"
1141             ],
1142             "x-from-ocf": [
1143               "currentlevel = contaminantvalue"
1144             ]
1145           }
1146         }
1147       }
1148     }
1149   }

```



```

1145     }
1146   },
1147   "maxlevel": {
1148     "type": "integer",
1149     "x-ocf-conversion": {
1150       "x-ocf-alias": "oic.r.airquality",
1151       "x-to-ocf": [
1152         "range[0] = 0",
1153         "range[1] = maxvalue"
1154       ],
1155       "x-from-ocf": [
1156         "maxvalue = range[1]"
1157       ]
1158     }
1159   }
1160 }
1161 }
1162 },
1163 "type": "object",
1164 "allOf": [
1165   {"$ref": "#/definitions/asa.environment.currentairqualitylevel"}
1166 ],
1167 "required": ["contaminanttype", "currentlevel", "maxlevel"]
1168 }
1169

```

1170 9.4 Current Humidity

1171 9.4.1 Derived model

1172 The derived model: "asa.environment.currenthumidity".

1173 9.4.2 Property definition

1174 Table 42 provides the detailed per Property mapping for "asa.environment.currenthumidity".

1175 **Table 42 – The property mapping for "asa.environment.currenthumidity".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
maxvalue		oic.r.humidity	range[0] = 0 range[1] = maxvalue	maxvalue = range[1]
currentvalue		oic.r.humidity	humidity currentValue	= currentvalue = humidity

1176 Table 43 provides the details of the Properties that are part of "asa.environment.currenthumidity".

1177 **Table 43 – The properties of "asa.environment.currenthumidity".**

AllJoyn name	Property	Type	Required	Description
maxvalue		number	yes	Max measured value for humidity
currentvalue		number	yes	Measured value

1178 9.4.3 Derived model definition

```

1179 {
1180   "id": "http://openinterconnect.org/asamapping/schemas/asa.environment.currenthumidity.json#",
1181   "$schema": "http://json-schema.org/draft-04/schema#",
1182   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1183   "title": "Current Humidity",
1184   "definitions": {
1185     "asa.environment.currenthumidity": {
1186       "type": "object",
1187       "properties": {
1188         "currentvalue": {
1189           "type": "number",

```

```

1190     "description": "Measured value",
1191     "x-ocf-conversion": {
1192       "x-ocf-alias": "oic.r.humidity",
1193       "x-to-ocf": [
1194         "humidity = currentValue"
1195       ],
1196       "x-from-ocf": [
1197         "currentvalue = humidity"
1198       ]
1199     },
1200   },
1201   "maxvalue": {
1202     "type": "number",
1203     "description": "Max measured value for humidty",
1204     "x-ocf-conversion": {
1205       "x-ocf-alias": "oic.r.humidity",
1206       "x-to-ocf": [
1207         "range[0] = 0",
1208         "range[1] = maxvalue"
1209       ],
1210       "x-from-ocf": [
1211         "maxvalue = range[1]"
1212       ]
1213     },
1214   },
1215 },
1216 },
1217 },
1218 "type": "object",
1219 "allOf": [
1220   {"$ref": "#/definitions/asa.environment.currenthumidity"}
1221 ],
1222 "required": [ "currentvalue", "maxvalue" ]
1223 }
1224

```

1225 **9.5 Current Temperature**

1226 **9.5.1 Derived model**

1227 The derived model: "asa.environment.currenttemperature".

1228 **9.5.2 Property definition**

1229 Table 44 provides the detailed per Property mapping for "asa.environment.currenttemperature".

1230 **Table 44 – The property mapping for "asa.environment.currenttemperature".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
precision		oic.r.temperature	ocf.precision = precision	precision = ocf.precision
currentvalue		oic.r.temperature	temperature = currentValueunits = C	oneOf
updatemintime		oic.r.value.conditional	ocf.minnotifyperiod = updatemintime	updatemintime = ocf.minnotifyperiod

1231 Table 45 provides the details of the Properties that are part of
1232 "asa.environment.currenttemperature".

1233 **Table 45 – The properties of "asa.environment.currenttemperature".**

AllJoyn name	Property	Type	Required	Description
precision		number	yes	
currentvalue		number	yes	Measured value
updatemintime		integer	yes	

1234 9.5.3 Derived model definition

```
1235 {
1236   "id": "http://openinterconnect.org/asamapping/schemas/asa.environment.currenttemperature.json#",
1237   "$schema": "http://json-schema.org/draft-04/schema#",
1238   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1239   "title": "Current Temperature",
1240   "definitions": {
1241     "asa.environment.currenttemperature": {
1242       "type": "object",
1243       "properties": {
1244         "currentvalue": {
1245           "type": "number",
1246           "description": "Measured value",
1247           "x-ocf-conversion": {
1248             "x-ocf-alias": "oic.r.temperature",
1249             "x-to-ocf": [
1250               "temperature = currentValue",
1251               "units = C"
1252             ],
1253             "x-from-ocf": {
1254               "oneOf": [
1255                 {
1256                   "properties": {
1257                     "units": "string",
1258                     "enum": ["C"]
1259                   },
1260                   "x-from-ocf": [
1261                     "currentvalue = temperature"
1262                   ]
1263                 },
1264                 {
1265                   "properties": {
1266                     "units": "string",
1267                     "enum": ["F"]
1268                   },
1269                   "x-from-ocf": [
1270                     "currentvalue = (temperature-32)*5/9"
1271                   ]
1272                 },
1273                 {
1274                   "properties": {
1275                     "units": "string",
1276                     "enum": ["K"]
1277                   },
1278                   "x-from-ocf": [
1279                     "currentvalue = temperature-273.15"
1280                   ]
1281                 }
1282               ]
1283             }
1284           },
1285         },
1286         "precision": {
1287           "type": "number",
1288           "x-ocf-conversion": {
1289             "x-ocf-alias": "oic.r.temperature",
1290             "x-to-ocf": [
1291               "ocf.precision = precision"
1292             ],
1293             "x-from-ocf": [
1294               "precision = ocf.precision"
1295             ]
1296           },
1297         },
1298         "updatemintime": {
1299           "type": "integer",
1300           "x-ocf-conversion": {
1301             "x-ocf-alias": "oic.r.value.conditional",
1302             "x-to-ocf": [
1303               "ocf.minnotifyperiod = updatemintime"

```

```

1304     ],
1305     "x-from-ocf": [
1306       "updatemintime = ocf.minnotifyperiod"
1307     ]
1308   }
1309 }
1310 }
1311 }
1312 },
1313 "type": "object",
1314 "allOf": [
1315   {"$ref": "#/definitions/asa.environment.currenttemperature"}
1316 ],
1317 "required": [ "currentvalue", "precision", "updatemintime" ]
1318 }
1319 }

```

1320 **9.6 Target Humidity**

1321 **9.6.1 Derived model**

1322 The derived model: "asa.environment.targethumidity".

1323 **9.6.2 Property definition**

1324 Table 46 provides the detailed per Property mapping for "asa.environment.targethumidity".

1325 **Table 46 – The property mapping for "asa.environment.targethumidity".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
minvalue	oic.r.humidity	range[0] = minvalue	minvalue = range[0] otherwise: minvalue = 0
targetvalue	oic.r.humidity,oic.r.selectablelevels	if minvalue != maxvalue, ocf.desiredhumidity = targetvalue;ocf.targetlevel = selectablehumiditylevels[0].if minvalue == maxvalue, ocf.targetlevel = targetvalue.	if x-ocf-alias == oic.r.humidity, targetvalue = desiredhumidity.if x-ocf-alias == oic.r.selectablelevels, targetvalue = targetlevel.
maxvalue	oic.r.humidity	range[1] = maxvalue	maxvalue = range[1] otherwise: maxvalue = 100
stepvalue	oic.r.humidity	step = stepvalue	stepvalue = step otherwise: stepvalue = 1
selectablehumidity levels	oic.r.selectablelevels	availablelevels[] = selectablehumiditylevels[]	selectablehumiditylevels[] = availablelevels[]

1326 Table 47 provides the details of the Properties that are part of "asa.environment.targethumidity".

1327 **Table 47 – The properties of "asa.environment.targethumidity".**

AllJoyn Property name	Type	Required	Description
minvalue	number	yes	
targetvalue	number	yes	Measured value
maxvalue	number	yes	
stepvalue	number	yes	

selectablehumiditylevels	array	yes	
--------------------------	-------	-----	--

1328 **9.6.3 Derived model definition**

```

1329 {
1330   "id": "http://openinterconnect.org/asamapping/schemas/asa.environment.targethumidity.json#",
1331   "$schema": "http://json-schema.org/draft-04/schema#",
1332   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1333   "title": "Target Humidity",
1334   "definitions": {
1335     "asa.environment.targethumidity": {
1336       "type": "object",
1337       "properties": {
1338         "targetvalue": {
1339           "type": "number",
1340           "description": "Measured value",
1341           "x-ocf-conversion": {
1342             "x-ocf-alias": "oic.r.humidity,oic.r.selectablelevels",
1343             "x-to-ocf": [
1344               "if minvalue != maxvalue, ocf.desiredhumidity = targetvalue;ocf.targetlevel =
1345 selectablehumiditylevels[0].",
1346               "if minvalue == maxvalue, ocf.targetlevel = targetvalue."
1347             ],
1348             "x-from-ocf": [
1349               "if x-ocf-alias == oic.r.humidity, targetvalue = desiredhumidity.",
1350               "if x-ocf-alias == oic.r.selectablelevels, targetvalue = targetlevel."
1351             ]
1352           }
1353         },
1354         "minvalue": {
1355           "type": "number",
1356           "x-ocf-conversion": {
1357             "x-ocf-alias": "oic.r.humidity",
1358             "x-to-ocf": [
1359               "range[0] = minvalue"
1360             ],
1361             "x-from-ocf": [
1362               "minvalue = range[0]",
1363               "otherwise: minvalue = 0"
1364             ]
1365           }
1366         },
1367         "maxvalue": {
1368           "type": "number",
1369           "x-ocf-conversion": {
1370             "x-ocf-alias": "oic.r.humidity",
1371             "x-to-ocf": [
1372               "range[1] = maxvalue"
1373             ],
1374             "x-from-ocf": [
1375               "maxvalue = range[1]",
1376               "otherwise: maxvalue = 100"
1377             ]
1378           }
1379         },
1380         "stepvalue": {
1381           "type": "number",
1382           "x-ocf-conversion": {
1383             "x-ocf-alias": "oic.r.humidity",
1384             "x-to-ocf": [
1385               "step = stepvalue"
1386             ],
1387             "x-from-ocf": [
1388               "stepvalue = step",
1389               "otherwise: stepvalue = 1"
1390             ]
1391           }
1392         },
1393         "selectablehumiditylevels": {
1394           "type": "array",
1395           "items": {

```

```

1396         "type": "number"
1397     },
1398     "x-ocf-conversion": {
1399         "x-ocf-alias": "oic.r.selectablelevels",
1400         "x-to-ocf": [
1401             "availablelevels[] = selectablehumiditylevels[]"
1402         ],
1403         "x-from-ocf": [
1404             "selectablehumiditylevels[] = availablelevels[]"
1405         ]
1406     }
1407 }
1408 }
1409 }
1410 },
1411 "type": "object",
1412 "allOf": [
1413     { "$ref": "#/definitions/asa.environment.targethumidity" }
1414 ],
1415 "required": [ "targetvalue", "minvalue", "maxvalue", "stepvalue", "selectablehumiditylevels" ]
1416 }
1417

```

1418 **9.7 Target Temperature**

1419 **9.7.1 Derived model**

1420 The derived model: "asa.environment.targettemperature".

1421 **9.7.2 Property definition**

1422 Table 48 provides the detailed per Property mapping for "asa.environment.targettemperature".

1423 **Table 48 – The property mapping for "asa.environment.targettemperature".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
minvalue		oic.r.temperature	range[0] = minvalue	minvalue = range[0] otherwise: minvalue = -MAXINT
targetvalue		oic.r.temperature	temperature = targetvalueunits = C	oneOf
maxvalue		oic.r.temperature	range[1] = maxvalue	maxvalue = range[1] otherwise: maxvalue = MAXINT
step		oic.r.temperature	ocf.step = step	step = ocf.step otherwise: step = undefined (0x00)

1424 Table 49 provides the details of the Properties that are part of "asa.environment.targettemperature".

1425 **Table 49 – The properties of "asa.environment.targettemperature".**

AllJoyn name	Property	Type	Required	Description
minvalue		number	yes	
targetvalue		number	yes	Measured value
maxvalue		number	yes	
step		number	yes	

1426 **9.7.3 Derived model definition**

```

1427 {
1428     "id": "http://openinterconnect.org/asamapping/schemas/asa.environment.targettemperature.json#",
1429     "$schema": "http://json-schema.org/draft-04/schema#",

```

```

1430 "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1431 "title": "Target Temperature",
1432 "definitions": {
1433   "asa.environment.targettemperature": {
1434     "type": "object",
1435     "properties": {
1436       "targetvalue": {
1437         "type": "number",
1438         "description": "Measured value",
1439         "x-ocf-conversion": {
1440           "x-ocf-alias": "oic.r.temperature",
1441           "x-to-ocf": [
1442             "temperature = targetvalue",
1443             "units = C"
1444           ],
1445           "x-from-ocf": {
1446             "oneOf": [
1447               {
1448                 "properties": {
1449                   "units": "string",
1450                   "enum": ["C"]
1451                 },
1452                 "x-from-ocf": [
1453                   "targetvalue = temperature"
1454                 ]
1455               },
1456               {
1457                 "properties": {
1458                   "units": "string",
1459                   "enum": ["F"]
1460                 },
1461                 "x-from-ocf": [
1462                   "targetvalue = (temperature-32)*5/9"
1463                 ]
1464               },
1465               {
1466                 "properties": {
1467                   "units": "string",
1468                   "enum": ["K"]
1469                 },
1470                 "x-from-ocf": [
1471                   "targetvalue = temperature-273.15"
1472                 ]
1473               }
1474             ]
1475           }
1476         },
1477         "minvalue": {
1478           "type": "number",
1479           "x-ocf-conversion": {
1480             "x-ocf-alias": "oic.r.temperature",
1481             "x-to-ocf": [
1482               "range[0] = minvalue"
1483             ],
1484             "x-from-ocf": [
1485               "minvalue = range[0]",
1486               "otherwise: minvalue = -MAXINT"
1487             ]
1488           }
1489         },
1490         "maxvalue": {
1491           "type": "number",
1492           "x-ocf-conversion": {
1493             "x-ocf-alias": "oic.r.temperature",
1494             "x-to-ocf": [
1495               "range[1] = maxvalue"
1496             ],
1497             "x-from-ocf": [
1498               "maxvalue = range[1]",
1499               "otherwise: maxvalue = MAXINT"
1500             ]

```

```

1501     ]
1502   }
1503 },
1504   "step": {
1505     "type": "number",
1506     "x-ocf-conversion": {
1507       "x-ocf-alias": "oic.r.temperature",
1508       "x-to-ocf": [
1509         "ocf.step = step"
1510       ],
1511       "x-from-ocf": [
1512         "step = ocf.step",
1513         "otherwise: step = undefined (0x00)"
1514       ]
1515     }
1516   }
1517 }
1518 }
1519 },
1520 "type": "object",
1521 "allOf": [
1522   {"$ref": "#/definitions/asa.environment.targettemperature"}
1523 ],
1524 "required": [ "targetvalue", "minvalue", "maxvalue", "step" ]
1525 }
1526

```

1527 **9.8 Audio Volume**

1528 **9.8.1 Derived model**

1529 The derived model: "asa.operation.audiovolume".

1530 **9.8.2 Property definition**

1531 Table 50 provides the detailed per Property mapping for "asa.operation.audiovolume".

1532 **Table 50 – The property mapping for "asa.operation.audiovolume".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
mute		oic.r.audio	ocf.mute = mute	mute = ocf.mute
maxvolume		oic.r.audio	range[0] = 0range[1] = maxvolume	maxvolume = range[1]otherwise: maxvalue = 100
volume		oic.r.audio	ocf.volume = volume	volume = ocf.volume

1533 Table 51 provides the details of the Properties that are part of "asa.operation.audiovolume".

1534 **Table 51 – The properties of "asa.operation.audiovolume".**

AllJoyn name	Property	Type	Required	Description
mute		boolean	yes	
maxvolume		integer	yes	
volume		integer	yes	Speaker volume index

1535 **9.8.3 Derived model definition**

```

1536 {
1537   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.audiovolume.json#",
1538   "$schema": "http://json-schema.org/draft-04/schema#",
1539   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1540   "title": "Audio Volume",
1541   "definitions": {
1542     "asa.operation.audiovolume": {
1543       "type": "object",
1544       "properties": {

```



```

1545     "volume": {
1546         "type": "integer",
1547         "description": "Speaker volume index",
1548         "x-ocf-conversion": {
1549             "x-ocf-alias": "oic.r.audio",
1550             "x-to-ocf": [
1551                 "ocf.volume = volume"
1552             ],
1553             "x-from-ocf": [
1554                 "volume = ocf.volume"
1555             ]
1556         }
1557     },
1558     "maxvolume": {
1559         "type": "integer",
1560         "x-ocf-conversion": {
1561             "x-ocf-alias": "oic.r.audio",
1562             "x-to-ocf": [
1563                 "range[0] = 0",
1564                 "range[1] = maxvolume"
1565             ],
1566             "x-from-ocf": [
1567                 "maxvolume = range[1]",
1568                 "otherwise: maxvalue = 100"
1569             ]
1570         }
1571     },
1572     "mute": {
1573         "type": "boolean",
1574         "x-ocf-conversion": {
1575             "x-ocf-alias": "oic.r.audio",
1576             "x-to-ocf": [
1577                 "ocf.mute = mute"
1578             ],
1579             "x-from-ocf": [
1580                 "mute = ocf.mute"
1581             ]
1582         }
1583     }
1584 }
1585 }
1586 },
1587 "type": "object",
1588 "allOf": [
1589     {"$ref": "#/definitions/asa.operation.audiovolume"}
1590 ],
1591 "required": [ "volume", "maxvolume", "mute" ]
1592 }
1593

```

1594 **9.9 Climate Control Mode**

1595 **9.9.1 Derived model**

1596 The derived model: "asa.operation.climatecontrolmode".

1597 **9.9.2 Property definition**

1598 Table 52 provides the detailed per Property mapping for "asa.operation.climatecontrolmode".

1599 **Table 52 – The property mapping for "asa.operation.climatecontrolmode".**

AllJoya n Proper ty name	OCF Resourc e	To OCF	From OCF

operationalstate	oic.r.operationalstate	machinestates = [Idle,Heating,Cooling,PendingHeat,PendingCool,AuxilliaryHeat]currentmachinestate = machinestates[operationalstate]	statearray = [Idle,Heating,Cooling,PendingHeat,PendingCool,AuxilliaryHeat]operationalstate = indexof statearray[currentmachinestate[0]]
supportedmodes	oic.r.mode	modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]for x=0, x < sizeof(supportedmodes): ocf.supportedmodes[x] = modearray[supportedmodes[x]]	modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]for x=0, x < sizeof(supportedmodes): supportedmodes[x] = indexof modearray[ocf.supportedmodes[x]]
mode	oic.r.mode	modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]ocf.mode[0] = modearray[mode]	modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]mode = indexof modeArray[ocf.mode[0]]

1600 Table 53 provides the details of the Properties that are part of "asa.operation.climatecontrolmode".

1601 **Table 53 – The properties of "asa.operation.climatecontrolmode".**

AllJoyn name	Property	Type	Required	Description
operationalstate		integer	yes	Current status of device
supportedmodes		array	yes	Array of supported modes
mode		integer	yes	Current mode of device.

1602 **9.9.3 Derived model definition**

```

1603 {
1604   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.climatecontrolmode.json#",
1605   "$schema": "http://json-schema.org/draft-04/schema#",
1606   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1607   "title": "Climate Control Mode",
1608   "definitions": {
1609     "asa.operation.climatecontrolmode": {
1610       "type": "object",
1611       "properties": {
1612         "mode": {
1613           "type": "integer",
1614           "description": "Current mode of device.",
1615           "x-ocf-conversion": {
1616             "x-ocf-alias": "oic.r.mode",
1617             "x-to-ocf": [
1618               "modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]",
1619               "ocf.mode[0] = modearray[mode]"
1620             ],
1621             "x-from-ocf": [
1622               "modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]",
1623               "mode = indexof modeArray[ocf.mode[0]]"
1624             ]
1625           }
1626         },
1627         "supportedmodes": {
1628           "type": "array",
1629           "items": {
1630             "type": "integer"
1631           },
1632           "description": "Array of supported modes",
1633           "x-ocf-conversion": {
1634             "x-ocf-alias": "oic.r.mode",
1635             "x-to-ocf": [
1636               "modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]",

```

```

1637         "for x=0, x < sizeof(supportedmodes): ocf.supportedmodes[x] =
1638 modearray[supportedmodes[x]]"
1639     ],
1640     "x-from-ocf": [
1641         "modearray = [Off,Heat,Cool,Auto,AuxilliaryHeat,Dry,ContinuousDry]",
1642         "for x=0, x < sizeof(supportedmodes): supportedmodes[x] = indexof
1643 modearray[ocf.supportedmodes[x]]"
1644     ]
1645 }
1646 },
1647 "operationalstate": {
1648     "type": "integer",
1649     "description": "Current status of device",
1650     "x-ocf-conversion": {
1651         "x-ocf-alias": "oic.r.operationalstate",
1652         "x-to-ocf": [
1653             "machinestates = [Idle,Heating,Cooling,PendingHeat,PendingCool,AuxilliaryHeat]",
1654             "currentmachinestate = machinestates[operationalstate]"
1655         ],
1656         "x-from-ocf": [
1657             "statearray = [Idle,Heating,Cooling,PendingHeat,PendingCool,AuxilliaryHeat]",
1658             "operationalstate = indexof statearray[currentmachinestate[0]]"
1659         ]
1660     }
1661 }
1662 }
1663 }
1664 },
1665 "type": "object",
1666 "allOf": [
1667     {"$ref": "#/definitions/asa.operation.climatecontrolmode"}
1668 ],
1669 "required": [ "mode","supportedmodes","operationalstate" ]
1670 }
1671

```

1672 **9.10 Closed Status**

1673 **9.10.1 Derived model**

1674 The derived model: "asa.operation.closedstatus".

1675 **9.10.2 Property definition**

1676 Table 54 provides the detailed per Property mapping for "asa.operation.closedstatus".

1677 **Table 54 – The property mapping for "asa.operation.closedstatus".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
isclosed		oic.r.door	if isClosed ocf.openState = Closed.if !isClosed ocf.openState = Open.	isClosed = (openState == Closed)

1678 Table 55 provides the details of the Properties that are part of "asa.operation.closedstatus".

1679 **Table 55 – The properties of "asa.operation.closedstatus".**

AllJoyn name	Property	Type	Required	Description
isclosed		boolean	yes	Open/Closed status Indicator

1680 **9.10.3 Derived model definition**

```

1681 {
1682   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.closedstatus.json#",
1683   "$schema": "http://json-schema.org/draft-04/schema#",
1684   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1685   "title": "Closed Status",
1686   "definitions": {
1687     "asa.operation.closedstatus": {
1688       "type": "object",
1689       "properties": {
1690         "isclosed": {
1691           "type": "boolean",
1692           "description": "Open/Closed status Indicator",
1693           "x-ocf-conversion": {
1694             "x-ocf-alias": "oic.r.door",
1695             "x-to-ocf": [
1696               "if isClosed ocf.openState = Closed.",
1697               "if !isClosed ocf.openState = Open."
1698             ],
1699             "x-from-ocf": [
1700               "isClosed = (openState == Closed)"
1701             ]
1702           }
1703         }
1704       }
1705     }
1706   },
1707   "type": "object",
1708   "allOf": [
1709     {"$ref": "#/definitions/asa.operation.closedstatus"}
1710   ],
1711   "required": [ "isclosed" ]
1712 }
1713

```

1714 **9.11 Cycle Control**

1715 **9.11.1 Derived model**

1716 The derived model: "asa.operation.cyclecontrol".

1717 **9.11.2 Property definition**

1718 Table 56 provides the detailed per Property mapping for "asa.operation.cyclecontrol".

1719 **Table 56 – The property mapping for "asa.operation.cyclecontrol".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
operationalstate	oic.r.operationalstate	statearray [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]currentmachinestate = statearray[operationalstate]	statearray [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]operationalstate = indexof statearray[currentmachinestate[0]]
executeoperationalcommand	oic.r.action		
SupportedOperationalcommands	oic.r.action		
supportedoperationalstates	oic.r.operationalstate	statearray [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]for x=0, x <	statearray [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]for x=0, x < sizeof(machinestates):

		sizeof(supportedoperationalstates): machinestates[x] = statearray[supportedoperationalstates[x]]	supportedoperationalstates[x] = indexof statearray[machinestates[x]]
--	--	--	--

1720 Table 57 provides the details of the Properties that are part of "asa.operation.cyclecontrol".

1721 **Table 57 – The properties of "asa.operation.cyclecontrol".**

AllJoyn Property name	Type	Required	Description
operationalstate	integer	yes	Current operational state of the appliance
executeoperationalcomand		no	Execute an operational command
SupportedOperationalcommands	array	no	Array of operatinal commands supported by the appliance
supportedoperationalstates	array	yes	Array of operational states supported by the Appliance.

1722 **9.11.3 Derived model definition**

```

1723 {
1724   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.cyclecontrol.json#",
1725   "$schema": "http://json-schema.org/draft-04/schema#",
1726   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1727   "title": "Cycle Control",
1728   "definitions": {
1729     "asa.operation.cyclecontrol": {
1730       "type": "object",
1731       "properties": {
1732         "operationalstate": {
1733           "type": "integer",
1734           "description": "Current operational state of the appliance",
1735           "x-ocf-conversion": {
1736             "x-ocf-alias": "oic.r.operationalstate",
1737             "x-to-ocf": [
1738               "statearray = [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]",
1739               "currentmachinestate = statearray[operationalstate]"
1740             ],
1741             "x-from-ocf": [
1742               "statearray = [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]",
1743               "operationalstate = indexof statearray[currentmachinestate[0]]"
1744             ]
1745           }
1746         },
1747         "supportedoperationalstates": {
1748           "type": "array",
1749           "items": {
1750             "type": "integer"
1751           },
1752           "description": "Array of operational states supported by the Appliance.",
1753           "x-ocf-conversion": {
1754             "x-ocf-alias": "oic.r.operationalstate",
1755             "x-to-ocf": [
1756               "statearray = [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]",
1757               "for x=0, x < sizeof(supportedoperationalstates): machinestates[x] =
1758 statearray[supportedoperationalstates[x]]"
1759             ],
1760             "x-from-ocf": [
1761               "statearray = [Idle,Working,ReadyToStart,DelayedStart,Pause,EndOfCycle]",

```

```

1762         "for x=0, x < sizeof(machinestates): supportedoperationalstates[x] = indexof
1763 statearray[machinestates[x]]"
1764     ]
1765 }
1766 },
1767 "SupportedOperationalcommands": {
1768     "type": "array",
1769     "items": {
1770         "type": "integer"
1771     },
1772     "description": "Array of operatinal commands supported by the appliance",
1773     "x-ocf-conversion": {
1774         "x-ocf-alias": "oic.r.action"
1775     }
1776 },
1777 "executeoperationalcomand": {
1778     "x-ocf-type": "method",
1779     "description": "Execute an operational command",
1780     "x-ocf-conversion": {
1781         "x-ocf-alias": "oic.r.action"
1782     }
1783 }
1784 }
1785 }
1786 },
1787 "type": "object",
1788 "allOf": [
1789     {"$ref": "#/definitions/asa.operation.cyclecontrol"}
1790 ],
1791 "required": [ "operationalstate", "supportedoperationalstates" ]
1792 }
1793

```

1794 **9.12 Fan Speed Level**

1795 **9.12.1 Derived model**

1796 The derived model: "asa.operation.fanspeedlevel".

1797 **9.12.2 Property definition**

1798 Table 58 provides the detailed per Property mapping for "asa.operation.fanspeedlevel".

1799 **Table 58 – The property mapping for "asa.operation.fanspeedlevel".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
fanspeedlevel		oic.r.airflow	speed = fanspeedlevel	fanspeedlevel = speed
maxfanspeedlevel		oic.r.airflow	range[0] = 0 range[1] = maxfanspeedlevel	maxfanspeedlevel = range[1] otherwise: maxfanspeedlevel = 100
automode		oic.r.airflow	if automode != NotSupported(0xFF) ocf.automode = automodeelse no mapping	automode = ocf.automodeotherwise: automode = NotSupported(0xFF)

1800 Table 59 provides the details of the Properties that are part of "asa.operation.fanspeedlevel".

1801 **Table 59 – The properties of "asa.operation.fanspeedlevel".**

AllJoyn name	Property	Type	Required	Description
--------------	----------	------	----------	-------------

fanspeedlevel	integer	yes	Fan speed level. 0 = off.
maxfanspeedlevel	integer	yes	Max level allowed for fan speed
automode	integer	yes	Auto mode status.

1802 **9.12.3 Derived model definition**

```

1803 {
1804   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.fanspeedlevel.json#",
1805   "$schema": "http://json-schema.org/draft-04/schema#",
1806   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1807   "title": "Fan Speed Level",
1808   "definitions": {
1809     "asa.operation.fanspeedlevel": {
1810       "type": "object",
1811       "properties": {
1812         "fanspeedlevel": {
1813           "type": "integer",
1814           "description": "Fan speed level. 0 = off.",
1815           "x-ocf-conversion": {
1816             "x-ocf-alias": "oic.r.airflow",
1817             "x-to-ocf": [
1818               "speed = fanspeedlevel"
1819             ],
1820             "x-from-ocf": [
1821               "fanspeedlevel = speed"
1822             ]
1823           }
1824         },
1825         "maxfanspeedlevel": {
1826           "type": "integer",
1827           "description": "Max level allowed for fan speed",
1828           "x-ocf-conversion": {
1829             "x-ocf-alias": "oic.r.airflow",
1830             "x-to-ocf": [
1831               "range[0] = 0",
1832               "range[1] = maxfanspeedlevel"
1833             ],
1834             "x-from-ocf": [
1835               "maxfanspeedlevel = range[1]",
1836               "otherwise: maxfanspeedlevel = 100"
1837             ]
1838           }
1839         },
1840         "automode": {
1841           "type": "integer",
1842           "description": "Auto mode status.",
1843           "x-ocf-conversion": {
1844             "x-ocf-alias": "oic.r.airflow",
1845             "x-to-ocf": [
1846               "if automode != NotSupported(0xFF)",
1847               " ocf.automode = automode",
1848               "else no mapping"
1849             ],
1850             "x-from-ocf": [
1851               "automode = ocf.automode",
1852               "otherwise: automode = NotSupported(0xFF)"
1853             ]
1854           }
1855         }
1856       }
1857     }
1858   },
1859   "type": "object",
1860   "allOf": [
1861     {"$ref": "#/definitions/asa.operation.fanspeedlevel"}
1862   ],
1863   "required": [ "fanspeedlevel", "maxfanspeedlevel", "automode" ]

```

1864 }
 1865

1866 **9.13 Heating Zone**

1867 **9.13.1 Derived model**

1868 The derived model: "asa.operation.heatingzone".

1869 **9.13.2 Property definition**

1870 Table 60 provides the detailed per Property mapping for "asa.operation.heatingzone".

1871 **Table 60 – The property mapping for "asa.operation.heatingzone".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
numberofheatingzones	oic.r.heatingzonecollection	number of links in the collection = numberofheatingzones	numberofheatingzones = number of links in the collection
heatinglevels	oic.r.heatingzone	Instance of oic.r.heatingzone per array item for x=0, x<sizeof(heatinglevels): ocf.heatinglevel = maxheatinglevels[x]	for x=0;x<numlinks(oic.r.heatingzonecollection): heatinglevels[x] = ocf.heatinglevel
maxheatinglevels	oic.r.heatingzone	Instance of oic.r.heatingzone per array item for x=0, x<sizeof(maxheatinglevels): ocf.maxheatinglevel = maxheatinglevels[x]	for x=0;x<numlinks(oic.r.heatingzonecollection): maxheatinglevels[x] = ocf.maxheatinglevel

1872 Table 61 provides the details of the Properties that are part of "asa.operation.heatingzone".

1873 **Table 61 – The properties of "asa.operation.heatingzone".**

AllJoyn Property name	Type	Required	Description
numberofheatingzones	integer	yes	Number of heating zones.
heatinglevels	array	yes	Current heating levels for each zone.
maxheatinglevels	array	yes	Max heating levels for each zone

1874 **9.13.3 Derived model definition**

```

1875 {
1876   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.heatingzone.json#",
1877   "$schema": "http://json-schema.org/draft-04/schema#",
1878   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1879   "title": "Heating Zone",
1880   "definitions": {
1881     "asa.operation.heatingzone": {
1882       "type": "object",
1883       "properties": {

```



```

1884     "numberofheatingzones": {
1885         "type": "integer",
1886         "description": "Number of heating zones.",
1887         "x-ocf-conversion": {
1888             "x-ocf-alias": "oic.r.heatingzonecollection",
1889             "x-to-ocf": [
1890                 "number of links in the collection = numberofheatingzones"
1891             ],
1892             "x-from-ocf": [
1893                 "numberofheatingzones = number of links in the collection"
1894             ]
1895         },
1896     },
1897     "maxheatinglevels": {
1898         "type": "array",
1899         "items": {
1900             "type": "integer"
1901         },
1902         "description": "Max heating levels for each zone",
1903         "x-ocf-conversion": {
1904             "x-ocf-alias": "oic.r.heatingzone",
1905             "x-to-ocf": [
1906                 "Instance of oic.r.heatingzone per array item ",
1907                 "for x=0, x<sizeof(maxheatinglevels): ocf.maxheatinglevel = maxheatinglevels[x]"
1908             ],
1909             "x-from-ocf": [
1910                 "for x=0;x<numlinks(oic.r.heatingzonecollection): maxheatinglevels[x] =
1911 ocf.maxheatinglevel"
1912             ]
1913         },
1914     },
1915     "heatinglevels": {
1916         "type": "array",
1917         "items": {
1918             "type": "integer"
1919         },
1920         "description": "Current heating levels for each zone.",
1921         "x-ocf-conversion": {
1922             "x-ocf-alias": "oic.r.heatingzone",
1923             "x-to-ocf": [
1924                 "Instance of oic.r.heatingzone per array item ",
1925                 "for x=0, x<sizeof(heatinglevels): ocf.heatinglevel = maxheatinglevels[x]"
1926             ],
1927             "x-from-ocf": [
1928                 "for x=0;x<numlinks(oic.r.heatingzonecollection): heatinglevels[x] = ocf.heatinglevel"
1929             ]
1930         },
1931     }
1932 }
1933 }
1934 },
1935 "type": "object",
1936 "allOf": [
1937     {"$ref": "#/definitions/asa.operation.heatingzone"}
1938 ],
1939 "required": [ "numberofheatingzones", "maxheatinglevels", "heatinglevels" ]
1940 }
1941

```

1942 9.14 HVAC Fan Mode

1943 9.14.1 Derived model

1944 The derived model: "asa.operation.hvacfanmode".

1945 9.14.2 Property definition

1946 Table 62 provides the detailed per Property mapping for "asa.operation.hvacfanmode".

1947

Table 62 – The property mapping for "asa.operation.hvacfanmode".

AllJoyn Property name	OCF Resource	To OCF	From OCF
mode	oic.r.mode	modearray = [Auto,Circulation,Continuous]ocf.mode[0] = modearray[mode]	modearray = [Auto,Circulation,Continuous]mode = indexof modeArray[ocf.mode[0]]
supportedmodes	oic.r.mode	modearray = [Auto,Circulation,Continuous]for x=0, x < sizeof(supportedmodes): ocf.supportedmodes[x] = modearray[supportedmodes[x]]	modearray = [Auto,Circulation,Continuous]for x=0, x < sizeof(supportedmodes): supportedmodes[x] = indexof modearray[ocf.supportedmodes[x]]

1948

Table 63 provides the details of the Properties that are part of "asa.operation.hvacfanmode".

1949

Table 63 – The properties of "asa.operation.hvacfanmode".

AllJoyn Property name	Type	Required	Description
mode	integer	yes	Current mode of device.
supportedmodes	array	yes	Array of supported modes

1950

9.14.3 Derived model definition

1951

```
{
1952   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.hvacfanmode.json#",
1953   "$schema": "http://json-schema.org/draft-04/schema#",
1954   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
1955   "title": "HVAC Fan Mode",
1956   "definitions": {
1957     "asa.operation.hvacfanmode": {
1958       "type": "object",
1959       "properties": {
1960         "mode": {
1961           "type": "integer",
1962           "description": "Current mode of device.",
1963           "x-ocf-conversion": {
1964             "x-ocf-alias": "oic.r.mode",
1965             "x-to-ocf": [
1966               "modearray = [Auto,Circulation,Continuous]",
1967               "ocf.mode[0] = modearray[mode]"
1968             ],
1969             "x-from-ocf": [
1970               "modearray = [Auto,Circulation,Continuous]",
1971               "mode = indexof modeArray[ocf.mode[0]]"
1972             ]
1973           }
1974         },
1975         "supportedmodes": {
1976           "type": "array",
1977           "items": {
1978             "type": "integer"
1979           },
1980           "description": "Array of supported modes",
1981           "x-ocf-conversion": {
1982             "x-ocf-alias": "oic.r.mode",
1983             "x-to-ocf": [
1984               "modearray = [Auto,Circulation,Continuous]",
1985               "for x=0, x < sizeof(supportedmodes): ocf.supportedmodes[x] ="
```

```

1986 modearray[supportedmodes[x]]"
1987     ],
1988     "x-from-ocf": [
1989         "modearray = [Auto,Circulation,Continuous]",
1990         "for x=0, x < sizeof(supportedmodes): supportedmodes[x] = indexof
1991 modearray[ocf.supportedmodes[x]]"
1992     ]
1993     }
1994 }
1995 }
1996 }
1997 },
1998 "type": "object",
1999 "allOf": [
2000     {"$ref": "#/definitions/asa.operation.hvacfanmode"}
2001 ],
2002 "required": [ "mode","supportedmodes" ]
2003 }
2004

```

2005 **9.15 On/Off Control**

2006 **9.15.1 Derived model**

2007 The derived model: "asa.operation.offcontrol".

2008 The derived model: "asa.operation.oncontrol".

2009 **9.15.2 Property definition**

2010 Table 64 provides the detailed per Property mapping for "asa.operation.offcontrol".

2011 **Table 64 – The property mapping for "asa.operation.offcontrol".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
switchon	oic.r.switch.binary	value = false	if ocf.value = false, asa.operation.offcontrol::switchoff().

2012 Table 65 provides the details of the Properties that are part of "asa.operation.offcontrol".

2013 **Table 65 – The properties of "asa.operation.offcontrol".**

AllJoyn name	Property	Type	Required	Description
switchon		string	no	Turn off the device

2014 Table 66 provides the detailed per Property mapping for "asa.operation.oncontrol".

2015 **Table 66 – The property mapping for "asa.operation.oncontrol".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
switchon	oic.r.switch.binary	value = true	if ocf.value = true, asa.operation.oncontrol::switchon().

2016 Table 67 provides the details of the Properties that are part of "asa.operation.oncontrol".

2017 **Table 67 – The properties of "asa.operation.oncontrol".**

AllJoyn name	Property	Type	Required	Description
switchon		string	no	Turn on the device

2018 **9.15.3 Derived model definition**

```

2019 {
2020   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.oncontrol.json#",
2021   "$schema": "http://json-schema.org/draft-04/schema#",
2022   "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
2023   "title": "On/Off Control",
2024   "definitions": {
2025     "asa.operation.oncontrol": {
2026       "type": "object",
2027       "properties": {
2028         "switchon": {
2029           "type": "string",
2030           "format": "method",
2031           "description": "Turn on the device",
2032           "x-ocf-conversion": {
2033             "x-ocf-alias": "oic.r.switch.binary",
2034             "x-to-ocf": [
2035               "value = true"
2036             ],
2037             "x-from-ocf": [
2038               "if ocf.value = true, asa.operation.oncontrol::switchon()."
2039             ]
2040           }
2041         }
2042       },
2043     },
2044     "asa.operation.offcontrol": {
2045       "type": "object",
2046       "properties": {
2047         "switchon": {
2048           "type": "string",
2049           "format": "method",
2050           "description": "Turn off the device",
2051           "x-ocf-conversion": {
2052             "x-ocf-alias": "oic.r.switch.binary",
2053             "x-to-ocf": [
2054               "value = false"
2055             ],
2056             "x-from-ocf": [
2057               "if ocf.value = false, asa.operation.offcontrol::switchoff()."
2058             ]
2059           }
2060         }
2061       },
2062     },
2063   },
2064   "type": "object",
2065   "oneOf": [
2066     {"$ref": "#/definitions/asa.operation.oncontrol"},
2067     {"$ref": "#/definitions/asa.operation.offcontrol"}
2068   ]
2069 }
2070

```

2071 **9.16 On Off Mapping**

2072 **9.16.1 Derived model**

2073 The derived model: "asa.operation.onoffstatus".

2074 **9.16.2 Property definition**

2075 Table 68 provides the detailed per Property mapping for "asa.operation.onoffstatus".

2076 **Table 68 – The property mapping for "asa.operation.onoffstatus".**

AllJoyn name	Property	OCF Resource	To OCF	From OCF
onoff		oic.r.switch.binary	value = onoff	onoff = value

2077 Table 69 provides the details of the Properties that are part of "asa.operation.onoffstatus".

2078 **Table 69 – The properties of "asa.operation.onoffstatus".**

AllJoyn name	Property	Type	Required	Description
onoff		boolean	yes	On/Off status of the device

2079 **9.16.3 Derived model definition**

```

2080 {
2081   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.onoffstatus.json#",
2082   "$schema": "http://json-schema.org/draft-04/schema#",
2083   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
2084   "title": "On Off Mapping",
2085   "definitions": {
2086     "asa.operation.onoffstatus": {
2087       "type": "object",
2088       "properties": {
2089         "onoff": {
2090           "type": "boolean",
2091           "description": "On/Off status of the device",
2092           "x-ocf-conversion": {
2093             "x-ocf-alias": "oic.r.switch.binary",
2094             "x-to-ocf": [
2095               "value = onoff"
2096             ],
2097             "x-from-ocf": [
2098               "onoff = value"
2099             ]
2100           }
2101         }
2102       }
2103     },
2104   },
2105   "type": "object",
2106   "allOf": [
2107     {"$ref": "#/definitions/asa.operation.onoffstatus"}
2108   ],
2109   "required": [ "onoff" ]
2110 }
2111

```

2112 **9.17 Oven Cycle Phase**

2113 **9.17.1 Derived model**

2114 The derived model: "asa.operation.ovencyclephase".

2115 **9.17.2 Property definition**

2116 Table 70 provides the detailed per Property mapping for "asa.operation.ovencyclephase".

2117 **Table 70 – The property mapping for "asa.operation.ovencyclephase".**

AllJoyn Property name	OCF Resource	To OCF	From OCF
getvendorphasesdescription	oic.r.action		
supportedcyclephases	oic.r.operationalstate	phasearray = [Unavailable,Preheating,Cooking,Cleaning]for x=0, x < sizeof(supportedcyclephases): machinestates[x] =	phasearray = [Unavailable,Preheating,Cooking,Cleaning]for x=0, x < sizeof(machinestates): supportedcyclephases[x] = indexof

		phasearray[supportedcyclephases[x]]	phasearray[machinestates[x]]
cyclephase	oic.r.operationalstate	phasearray [Unavailable,Preheating,Cooking,Cleaning]currentmachinestate = phasearray[cyclephase]	phasearray [Unavailable,Preheating,Cooking,Cleaning]cyclephase = indexof statearray[currentmachinestate[0]]

2118 Table 71 provides the details of the Properties that are part of "asa.operation.ovencyclephase".

2119 **Table 71 – The properties of "asa.operation.ovencyclephase".**

AllJoyn Property name	Type	Required	Description
getvendorphasesdescription		no	Get cycle phases description
supportedcyclephases	array	yes	Array of cycle phases supported by the Appliance.
cyclephase	integer	yes	Current phase of the operational cycle

2120 **9.17.3 Derived model definition**

```

2121 {
2122   "id": "http://openinterconnect.org/asamapping/schemas/asa.operation.ovencyclephase.json#",
2123   "$schema": "http://json-schema.org/draft-04/schema#",
2124   "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights reserved.",
2125   "title": "Oven Cycle Phase",
2126   "definitions": {
2127     "asa.operation.ovencyclephase": {
2128       "type": "object",
2129       "properties": {
2130         "cyclephase": {
2131           "type": "integer",
2132           "description": "Current phase of the operational cycle",
2133           "x-ocf-conversion": {
2134             "x-ocf-alias": "oic.r.operationalstate",
2135             "x-to-ocf": [
2136               "phasearray = [Unavailable,Preheating,Cooking,Cleaning]",
2137               "currentmachinestate = phasearray[cyclephase]"
2138             ],
2139             "x-from-ocf": [
2140               "phasearray = [Unavailable,Preheating,Cooking,Cleaning]",
2141               "cyclephase = indexof statearray[currentmachinestate[0]]"
2142             ]
2143           }
2144         },
2145         "supportedcyclephases": {
2146           "type": "array",
2147           "items": {
2148             "type": "integer"
2149           },
2150           "description": "Array of cycle phases supported by the Appliance.",
2151           "x-ocf-conversion": {
2152             "x-ocf-alias": "oic.r.operationalstate",
2153             "x-to-ocf": [
2154               "phasearray = [Unavailable,Preheating,Cooking,Cleaning]",
2155               "for x=0, x < sizeof(supportedcyclephases): machinestates[x] =
2156 phasearray[supportedcyclephases[x]]"
2157             ],
2158             "x-from-ocf": [
2159               "phasearray = [Unavailable,Preheating,Cooking,Cleaning]",
2160               "for x=0, x < sizeof(machinestates): supportedcyclephases[x] = indexof
2161 phasearray[machinestates[x]]"
2162             ]
2163           }
2164         }
2165       }
2166     }
2167   }

```

```

2164     },
2165     "getvendorphasesdescription": {
2166       "x-ocf-type": "method",
2167       "description": "Get cycle phases description",
2168       "x-ocf-conversion": {
2169         "x-ocf-alias": "oic.r.action"
2170       }
2171     }
2172   }
2173 },
2174 },
2175 "type": "object",
2176 "allOf": [
2177   { "$ref": "#/definitions/asa.operation.ovencyclephase" }
2178 ],
2179 "required": [ "cyclephase", "supportedcyclephases" ]
2180 }
2181

```

2182 10 Resource type definitions

2183 10.1 List of Resource types

2184 Table 72 lists the Resource Types defined in this document.

2185 **Table 72 – Alphabetical list of resource types**

Friendly Name (informative)	Resource Type (rt)	Clause
AllJoyn Object	oic.r.alljoynobject	10.2
Secure Mode	oic.r.securemode	10.3

2186 10.2 AllJoynObject

2187 10.2.1 Introduction

2188 This Resource is a Collection of Resources that were all derived from the same AllJoyn object.

2189 10.2.2 Example URI

2190 /example/AllJoynObject

2191 10.2.3 Resource type

2192 The Resource Type is defined as: "oic.r.alljoynobject, oic.wk.col".

2193 10.2.4 OpenAPI 2.0 definition

```

2194 {
2195   "swagger": "2.0",
2196   "info": {
2197     "title": "AllJoynObject",
2198     "version": "2019-03-19",
2199     "license": {
2200       "name": "OCF Data Model License",
2201       "url":
2202         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
2203         CENSE.md",
2204       "x-copyright": "Copyright 2016-2019 Open Connectivity Foundation, Inc. All rights reserved."
2205     },
2206     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
2207   },
2208   "schemes": ["http"],
2209   "consumes": ["application/json"],
2210   "produces": ["application/json"],
2211   "paths": {
2212     "/example/AllJoynObject?if=oic.if.ll": {
2213       "get": {
2214         "description": "This Resource is a Collection of Resources that were all derived from the
2215         same AllJoyn object.\n",

```

```

2216     "parameters": [
2217         {"$ref": "#/parameters/interface-all"}
2218     ],
2219     "responses": {
2220         "200": {
2221             "description": "",
2222             "x-example": [
2223                 {
2224                     "href": "/myRes1URI",
2225                     "rt": ["x.example.widget.false"],
2226                     "if": ["oic.if.r", "oic.if.rw", "oic.if.baseline"],
2227                     "eps": [
2228                         {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2229                     ]
2230                 },
2231                 {
2232                     "href": "/myRes2URI",
2233                     "rt": ["x.example.widget.true"],
2234                     "if": ["oic.if.r", "oic.if.rw", "oic.if.baseline"],
2235                     "eps": [
2236                         {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2237                     ]
2238                 },
2239                 {
2240                     "href": "/myRes3URI",
2241                     "rt": ["x.example.widget.method1"],
2242                     "if": ["oic.if.rw", "oic.if.baseline"],
2243                     "eps": [
2244                         {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2245                     ]
2246                 },
2247                 {
2248                     "href": "/myRes4URI",
2249                     "rt": ["x.example.widget.method2"],
2250                     "if": ["oic.if.rw", "oic.if.baseline"],
2251                     "eps": [
2252                         {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2253                     ]
2254                 }
2255             ],
2256             "schema": {
2257                 "$ref": "#/definitions/slinks"
2258             }
2259         }
2260     }
2261 },
2262 {
2263     "/example/AllJoynObject?if=oic.if.baseline": {
2264         "get": {
2265             "description": "This Resource is a Collection of Resources that were all derived from the
2266 same AllJoyn object.\n",
2267             "parameters": [
2268                 {"$ref": "#/parameters/interface-all"}
2269             ],
2270             "responses": {
2271                 "200": {
2272                     "description": "",
2273                     "x-example": {
2274                         "rt": ["oic.r.alljoynobject", "oic.wk.col"],
2275                         "links": [
2276                             {
2277                                 "href": "/myRes1URI",
2278                                 "rt": ["x.example.widget.false"],
2279                                 "if": ["oic.if.r", "oic.if.rw", "oic.if.baseline"],
2280                                 "eps": [
2281                                     {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2282                                 ]
2283                             },
2284                             {
2285                                 "href": "/myRes2URI",
2286                                 "rt": ["x.example.widget.true"],

```



```

2287         "if": ["oic.if.r", "oic.if.rw", "oic.if.baseline"],
2288         "eps": [
2289             {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2290         ],
2291     },
2292     {
2293         "href": "/myRes3URI",
2294         "rt": ["x.example.widget.method1"],
2295         "if": ["oic.if.rw", "oic.if.baseline"],
2296         "eps": [
2297             {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2298         ],
2299     },
2300     {
2301         "href": "/myRes4URI",
2302         "rt": ["x.example.widget.method2"],
2303         "if": ["oic.if.rw", "oic.if.baseline"],
2304         "eps": [
2305             {"ep": "coaps://[2001:db8:a::b1d4]:11111"}
2306         ]
2307     }
2308 ],
2309 },
2310 "schema": {
2311     "$ref": "#/definitions/AllJoynObject"
2312 }
2313 }
2314 }
2315 }
2316 },
2317 "parameters": {
2318     "interface-all": {
2319         "in": "query",
2320         "name": "if",
2321         "type": "string",
2322         "enum": ["oic.if.ll", "oic.if.baseline"]
2323     }
2324 },
2325 "definitions": {
2326     "oic.oic-link": {
2327         "type": "object",
2328         "properties": {
2329             "anchor": {
2330                 "$ref":
2331 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2332 schema.json#/definitions/anchor"
2333             },
2334             "di": {
2335                 "$ref":
2336 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2337 schema.json#/definitions/di"
2338             },
2339             "eps": {
2340                 "$ref":
2341 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2342 schema.json#/definitions/eps"
2343             },
2344             "href": {
2345                 "$ref":
2346 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2347 schema.json#/definitions/href"
2348             },
2349             "ins": {
2350                 "$ref":
2351 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2352 schema.json#/definitions/ins"
2353             },
2354             "p": {
2355                 "$ref":
2356 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2357

```

```

2358 schema.json#/definitions/p"
2359     },
2360     "rel": {
2361         "$ref":
2362 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2363 schema.json#/definitions/rel_array"
2364     },
2365     "title": {
2366         "$ref":
2367 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2368 schema.json#/definitions/title"
2369     },
2370     "type": {
2371         "$ref":
2372 "https://openconnectivityfoundation.github.io/core/schemas/oic.links.properties.core-
2373 schema.json#/definitions/type"
2374     },
2375     "if": {
2376         "description": "The OCF Interfaces supported by the target Resource",
2377         "items": {
2378             "enum": [
2379                 "oic.if.baseline",
2380                 "oic.if.ll",
2381                 "oic.if.r",
2382                 "oic.if.rw"
2383             ],
2384             "type": "string",
2385             "maxLength": 64
2386         },
2387         "minItems": 1,
2388         "uniqueItems": true,
2389         "type": "array"
2390     },
2391     "rt": {
2392         "description": "Resource Type of the target Resource",
2393         "items": {
2394             "maxLength": 64,
2395             "type": "string"
2396         },
2397         "minItems": 1,
2398         "uniqueItems": true,
2399         "type": "array"
2400     }
2401 },
2402 "required": [
2403     "href",
2404     "rt",
2405     "if"
2406 ]
2407 },
2408 "slinks" : {
2409     "type": "array",
2410     "items": {
2411         "$ref": "#/definitions/oic.oic-link"
2412     }
2413 },
2414 "AllJoynObject": {
2415     "type": "object",
2416     "properties": {
2417         "id": {
2418             "$ref":
2419 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
2420 schema.json#/definitions/id"
2421         },
2422         "if": {
2423             "description": "The interface set supported by this resource",
2424             "items": {
2425                 "enum": ["oic.if.baseline", "oic.if.ll"],
2426                 "type": "string"
2427             },
2428             "minItems": 1,

```

```

2429         "readOnly": true,
2430         "type": "array"
2431     },
2432     "n": {
2433         "$ref":
2434         "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
2435         schema.json#/definitions/n"
2436     },
2437     "rt": {
2438         "items": {
2439             "enum": ["oic.r.alljoynobject", "oic.wk.col"],
2440             "type": "string"
2441         },
2442         "maxItems": 2,
2443         "minItems": 2,
2444         "uniqueItems": true,
2445         "readOnly": true,
2446         "type": "array"
2447     },
2448     "links" : {
2449         "type": "array",
2450         "description": "A set of OCF Links.",
2451         "items": {
2452             "$ref": "#/definitions/oic.oic-link"
2453         }
2454     }
2455 }
2456 }
2457 }
2458 }
2459

```

2460 **10.2.5 Property definition**

2461 Table 73 defines the Properties that are part of the "oic.r.alljoynobject, oic.wk.col" Resource Type.

2462 **Table 73 – The Property definitions of the Resource with type "rt" = "oic.r.alljoynobject,**
2463 **oic.wk.col".**

Property name	Value type	Mandatory	Access mode	Description
id	multiple types: see schema		Read Write	
links	array: see schema		Read Write	A set of OCF Links.
n	multiple types: see schema		Read Write	
rt	array: see schema		Read Only	
if	array: see schema		Read Only	The interface set supported by this resource
rel	multiple types: see schema	No	Read Write	
type	multiple types: see schema	No	Read Write	
if	array: see schema	Yes	Read Write	The OCF Interfaces supported by the target Resource
p	multiple types: see schema	No	Read Write	
anchor	multiple types: see schema	No	Read Write	

rt	array: see schema	Yes	Read Write	Resource Type of the target Resource
eps	multiple types: see schema	No	Read Write	
href	multiple types: see schema	Yes	Read Write	
ins	multiple types: see schema	No	Read Write	
title	multiple types: see schema	No	Read Write	
di	multiple types: see schema	No	Read Write	

2464 **10.2.6 CRUDN behaviour**

2465 Table 74 defines the CRUDN operations that are supported on the "oic.r.alljoynobject, oic.wk.col"
2466 Resource Type.

2467 **Table 74 – The CRUDN operations of the Resource with type "rt" = "oic.r.alljoynobject,**
2468 **oic.wk.col".**

Create	Read	Update	Delete	Notify
	get			observe

2469 **10.3 SecureMode**

2470 **10.3.1 Introduction**

2471 This Resource describes a secure mode on/off feature (on/off).

2472 A secureMode value of 'true' means that the feature is on, and any Bridged Server that cannot be
2473 communicated with securely shall not have a corresponding Virtual OCF Server, and any Bridged
2474 Client that cannot be communicated with securely shall not have a corresponding Virtual OCF
2475 Client.

2476 A secureMode value of 'false' means that the feature is off, any Bridged Server can have a
2477 corresponding Virtual OCF Server, and any Bridged Client can have a corresponding Virtual OCF
2478 Client.

2479 **10.3.2 Example URI**

2480 /example/SecureModeResURI

2481 **10.3.3 Resource type**

2482 The Resource Type is defined as: "oic.r.securemode".

2483 **10.3.4 OpenAPI 2.0 definition**

```

2484 {
2485   "swagger": "2.0",
2486   "info": {
2487     "title": "SecureMode",
2488     "version": "2019-03-19",
2489     "license": {
2490       "name": "OCF Data Model License",
2491       "url":
2492         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
2493         CENSE.md",
2494       "x-copyright": "Copyright 2016-2019 Open Connectivity Foundation, Inc. All rights reserved."
2495     },
2496     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
2497   },
2498   "schemes": ["http"],
2499   "consumes": ["application/json"],
2500   "produces": ["application/json"],
2501   "paths": {

```

```

2502     "/example/SecureModeResURI": {
2503         "get": {
2504             "description": "This Resource describes a secure mode on/off feature (on/off).\nA secureMode
2505 value of 'true' means that the feature is on, and any Bridged Server that cannot be communicated
2506 with securely shall not have a corresponding Virtual OCF Server, and any Bridged Client that cannot
2507 be communicated with securely shall not have a corresponding Virtual OCF Client.\nA secureMode value
2508 of 'false' means that the feature is off, any Bridged Server can have a corresponding Virtual OCF
2509 Server, and any Bridged Client can have a corresponding Virtual OCF Client.\n",
2510             "parameters": [
2511                 {"$ref": "#/parameters/interface"}
2512             ],
2513             "responses": {
2514                 "200": {
2515                     "description": "",
2516                     "x-example": {
2517                         "rt": ["oic.r.securemode"],
2518                         "secureMode": false
2519                     },
2520                     "schema": {
2521                         "$ref": "#/definitions/SecureMode"
2522                     }
2523                 }
2524             },
2525             "post": {
2526                 "description": "Updates the value of secureMode.\n",
2527                 "parameters": [
2528                     {"$ref": "#/parameters/interface"},
2529                     {
2530                         "name": "body",
2531                         "in": "body",
2532                         "required": true,
2533                         "schema": {
2534                             "$ref": "#/definitions/SecureMode-Update"
2535                         },
2536                         "x-example": {
2537                             "secureMode": true
2538                         }
2539                     }
2540                 ],
2541                 "responses": {
2542                     "200": {
2543                         "description": "",
2544                         "x-example": {
2545                             "secureMode": true
2546                         },
2547                         "schema": {
2548                             "$ref": "#/definitions/SecureMode"
2549                         }
2550                     }
2551                 }
2552             }
2553         }
2554     },
2555     "parameters": {
2556         "interface": {
2557             "in": "query",
2558             "name": "if",
2559             "type": "string",
2560             "enum": ["oic.if.rw", "oic.if.baseline"]
2561         }
2562     },
2563     "definitions": {
2564         "SecureMode": {
2565             "properties": {
2566                 "id": {
2567                     "$ref":
2568 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
2569 schema.json#/definitions/id"
2570                 },
2571                 "if": {
2572

```

```

2573         "description": "The interface set supported by this resource",
2574         "items": {
2575             "enum": ["oic.if.baseline", "oic.if.rw"],
2576             "type": "string",
2577             "maxLength": 64
2578         },
2579         "minItems": 1,
2580         "readOnly": true,
2581         "uniqueItems": true,
2582         "type": "array"
2583     },
2584     "n": {
2585         "$ref":
2586         "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
2587         schema.json#/definitions/n"
2588     },
2589     "rt": {
2590         "description": "Resource Type",
2591         "items": {
2592             "enum": ["oic.r.securemode"],
2593             "type": "string",
2594             "maxLength": 64
2595         },
2596         "minItems": 1,
2597         "uniqueItems": true,
2598         "readOnly": true,
2599         "type": "array"
2600     },
2601     "secureMode": {
2602         "description": "Status of the Secure Mode",
2603         "type": "boolean"
2604     }
2605 },
2606 "required": ["secureMode"],
2607 "type": "object"
2608 },
2609 "SecureMode-Update": {
2610     "properties": {
2611         "secureMode": {
2612             "description": "Status of the Secure Mode",
2613             "type": "boolean"
2614         }
2615     }
2616 }
2617 }
2618 }
2619

```

2620 10.3.5 Property definition

2621 Table 75 defines the Properties that are part of the "oic.r.securemode" Resource Type.

2622 **Table 75 – The Property definitions of the Resource with type "rt" = "oic.r.securemode".**

Property name	Value type	Mandatory	Access mode	Description
secureMode	boolean		Read Write	Status of the Secure Mode
secureMode	boolean	Yes	Read Write	Status of the Secure Mode
n	multiple types: see schema	No	Read Write	
if	array: see schema	No	Read Only	The interface set supported by this resource
rt	array: see schema	No	Read Only	Resource Type

id	multiple types: see schema	No	Read Write	
----	-------------------------------	----	------------	--

2623 **10.3.6 CRUDN behaviour**

2624 Table 76 defines the CRUDN operations that are supported on the "oic.r.securemode" Resource
2625 Type.

2626 **Table 76 – The CRUDN operations of the Resource with type "rt" = "oic.r.securemode".**

Create	Read	Update	Delete	Notify
	get	post		observe

2627

2628