

# OCF Bridging Specification

VERSION 2.1.1 | February 2020



**OPEN** CONNECTIVITY  
FOUNDATION™

CONTACT [admin@openconnectivity.org](mailto:admin@openconnectivity.org)

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

## 1 **LEGAL DISCLAIMER**

2  
3 NOTHING CONTAINED IN THIS DOCUMENT SHALL BE DEEMED AS GRANTING YOU ANY KIND  
4 OF LICENSE IN ITS CONTENT, EITHER EXPRESSLY OR IMPLIEDLY, OR TO ANY  
5 INTELLECTUAL PROPERTY OWNED OR CONTROLLED BY ANY OF THE AUTHORS OR  
6 DEVELOPERS OF THIS DOCUMENT. THE INFORMATION CONTAINED HEREIN IS PROVIDED  
7 ON AN "AS IS" BASIS, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW,  
8 THE AUTHORS AND DEVELOPERS OF THIS SPECIFICATION HEREBY DISCLAIM ALL OTHER  
9 WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, STATUTORY OR AT  
10 COMMON LAW, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF  
11 MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OPEN CONNECTIVITY  
12 FOUNDATION, INC. FURTHER DISCLAIMS ANY AND ALL WARRANTIES OF NON-  
13 INFRINGEMENT, ACCURACY OR LACK OF VIRUSES.

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

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

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

# CONTENTS

21 1 Scope..... 1

22 2 Normative references ..... 1

23 3 Terms, definitions, and abbreviated terms ..... 1

24 3.1 Terms and definitions..... 1

25 3.2 Abbreviated terms..... 3

26 4 Document conventions and organization..... 4

27 4.1 Conventions..... 4

28 4.2 Notation..... 4

29 5 Introduction ..... 5

30 5.1 Translation between OCF and Non-OCF ecosystem - primitive concept of

31 Bridging ..... 5

32 5.2 Bridge Platform..... 5

33 5.3 Symmetric vs. asymmetric bridging ..... 6

34 5.4 General requirements ..... 8

35 5.4.1 Requirements common to all Bridge Platforms..... 8

36 5.4.2 Requirements specific to Symmetric Bridge Platforms ..... 8

37 5.5 VOD List ..... 8

38 5.6 Resource discovery ..... 8

39 5.7 "Deep translation" vs. "on-the-fly" ..... 14

40 5.8 Security ..... 14

41 6 Device type definitions ..... 14

42 7 Resource type definitions ..... 14

43 7.1 List of resource types..... 14

44 7.2 VOD List ..... 15

45 7.2.1 Introduction ..... 15

46 7.2.2 Example URI ..... 15

47 7.2.3 Resource type ..... 15

48 7.2.4 OpenAPI 2.0 definition..... 15

49 7.2.5 Property definition ..... 17

50 7.2.6 CRUDN behaviour ..... 17

53

## Figures

54	Figure 1 – Server bridging to Non- OCF device.....	5
55	Figure 2 – Bridge Platform components .....	5
56	Figure 3 – Schematic overview of a Bridge Platform bridging non-OCF devices .....	6
57	Figure 4 – Asymmetric server bridge.....	7
58	Figure 5 – Asymmetric client bridge .....	7
59	Figure 6 – /oic/res example responses.....	14
60		

61

## Tables

62	Table 1 – Device type definitions .....	14
63	Table 2 – Alphabetical list of resource types .....	15
64	Table 3 – The Property definitions of the Resource with type "rt" = "oic.r.vodlist".....	17
65	Table 4 – The CRUDN operations of the Resource with type "rt" = "oic.r.vodlist". .....	17
66		

## 67 **1 Scope**

68 This document specifies a framework for translation between OCF Devices and other ecosystems,  
69 and specifies the behaviour of a Bridging Function that exposes servers in non-OCF ecosystem to  
70 OCF Clients and/or exposes OCF Servers to clients in non-OCF ecosystem. Translation per  
71 specific Device is left to other documents (deep translation). This document provides generic  
72 requirements that apply unless overridden by a more specific document.

## 73 **2 Normative references**

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

78 ISO/IEC 30118-1:2018 Information technology -- Open Connectivity Foundation (OCF)  
79 Specification -- Part 1: Core specification  
80 <https://www.iso.org/standard/53238.html>  
81 Latest version available at: [https://openconnectivity.org/specs/OCF\\_Core\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Core_Specification.pdf)

82 ISO/IEC 30118-2:2018 Information technology -- Open Connectivity Foundation (OCF)  
83 Specification -- Part 2: Security specification  
84 <https://www.iso.org/standard/74239.html>  
85 Latest version available at: [https://openconnectivity.org/specs/OCF\\_Security\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Security_Specification.pdf)

86 OpenAPI Specification, Version 2.0  
87 <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md>

## 88 **3 Terms, definitions, and abbreviated terms**

### 89 **3.1 Terms and definitions**

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

92 ISO and IEC maintain terminological databases for use in standardization at the following  
93 addresses:

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

#### 96 **3.1.1**

##### 97 **Asymmetric Client Bridge**

98 an asymmetric client bridge exposes another ecosystem clients into the OCF ecosystem as Virtual  
99 OCF Clients (3.1.22). This is equivalent to exposing OCF Servers (3.1.18) into the other ecosystem.  
100 How this is handled in each ecosystem is specified on a per ecosystem basis in this document.

#### 101 **3.1.2**

##### 102 **Asymmetric Server Bridge**

103 an asymmetric server bridge exposes another ecosystem devices into the OCF ecosystem as  
104 Virtual OCF Servers (3.1.25). How this is handled in each ecosystem is specified on a per  
105 ecosystem basis in this document.

#### 106 **3.1.3**

##### 107 **Bridge**

108 OCF Device that has a Device Type of "oic.d.bridge", provides information on the set of Virtual  
109 OCF Devices (3.1.23) that are resident on the same Bridge Platform.

110 **3.1.4**  
111 **Bridge Platform**  
112 Entity on which the Bridge (3.1.3) and Virtual OCF Devices (3.1.23) are resident

113 **3.1.5**  
114 **Bridged Client**  
115 logical entity that accesses data via a Bridged Protocol (3.1.7).

116 **3.1.6**  
117 **Bridged Device**  
118 Bridged Client (3.1.5) or Bridged Server (3.1.10).

119 **3.1.7**  
120 **Bridged Protocol**  
121 another protocol (e.g., AllJoyn) that is being translated to or from OCF protocols

122 **3.1.8**  
123 **Bridged Resource**  
124 represents an artefact modelled and exposed by a Bridged Protocol (3.1.7).

125 **3.1.9**  
126 **Bridged Resource Type**  
127 schema used with a Bridged Protocol (3.1.7).

128 **3.1.10 Bridged Server**  
129 logical entity that provides data via a Bridged Protocol (3.1.7). More than one Bridged Server can  
130 exist on the same physical platform.

131 **3.1.11**  
132 **Bridging Function**  
133 Logic resident on the Bridge Platform (3.1.4) that performs that protocol mapping between OCF  
134 and the Bridged Protocol (3.1.7); a Bridge Platform (3.1.4) may contain multiple Bridging Functions  
135 dependent on the number of Bridged Protocols (3.1.7) supported.

136 **3.1.12**  
137 **OCF Bridge Device**  
138 OCF Device (3.1.14) that can represent devices that exist on the network but communicate using  
139 a Bridged Protocol (3.1.7) rather than OCF protocols.

140 **3.1.13**  
141 **OCF Client**  
142 logical entity that accesses an OCF Resource (3.1.15) on an OCF Server (3.1.18), which might be  
143 a Virtual OCF Server (3.1.25) exposed by the OCF Bridge Device (3.1.12)

144 **3.1.14**  
145 **OCF Device**  
146 logical entity that assumes one or more OCF roles (OCF Client (3.1.13), OCF Server (3.1.18)). More  
147 than one OCF Device can exist on the same physical platform.

148 **3.1.15**  
149 **OCF Resource**  
150 represents an artefact modelled and exposed by the OCF Framework

151 **3.1.16**  
152 **OCF Resource Property**  
153 significant aspect or notion including metadata that is exposed through the OCF Resource (3.1.15)

154 **3.1.17**  
155 **OCF Resource Type**  
156 OCF Resource Property (3.1.16) that represents the data type definition for the OCF Resource  
157 (3.1.15)

158 **3.1.18**  
159 **OCF Server**  
160 logical entity with the role of providing resource state information and allowing remote control of its  
161 resources

162 **3.1.19**  
163 **Symmetric, Asymmetric Bridging**  
164 in symmetric bridging, a bridge device exposes OCF Server(s) (3.1.18) to another ecosystem and  
165 exposes other ecosystem's server(s) to OCF. In asymmetric bridging, a bridge device exposes  
166 OCF Server(s) (3.1.18) to another ecosystem or exposes another ecosystem's server(s) to OCF,  
167 but not both.

168 **3.1.20**  
169 **Virtual Bridged Client**  
170 logical representation of an OCF Client (3.1.13), which an OCF Bridge Device (3.1.12) exposes to  
171 Bridged Servers (3.1.10).

172 **3.1.21**  
173 **Virtual Bridged Server**  
174 logical representation of an OCF Server (3.1.18), which an OCF Bridge Device (3.1.12) exposes to  
175 Bridged Clients (3.1.5).

176 **3.1.22**  
177 **Virtual OCF Client**  
178 logical representation of a Bridged Client (3.1.5), which an OCF Bridge Device (3.1.12) exposes to  
179 OCF Servers (3.1.18)

180 **3.1.23**  
181 **Virtual OCF Device**  
182 Virtual OCF Client (3.1.22) or Virtual OCF Server (3.1.25).

183 **3.1.24**  
184 **Virtual OCF Resource**  
185 logical representation of a Bridged Resource (3.1.8), which an OCF Bridge Device (3.1.12) exposes  
186 to OCF Clients (3.1.13)

187 **3.1.25**  
188 **Virtual OCF Server**  
189 logical representation of a Bridged Server (3.1.10), which an OCF Bridge Device (3.1.12) exposes  
190 to OCF Clients (3.1.13).

191 **3.2 Abbreviated terms**

192 **3.2.1**  
193 **CRUDN**  
194 Create, Read, Update, Delete, and Notify

195 **3.2.2**  
196 **CSV**  
197 Comma separated value



## 198 4 Document conventions and organization

### 199 4.1 Conventions

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

### 204 4.2 Notation

205 In this document, features are described as required, recommended, allowed or DEPRECATED as  
206 follows:

207 Required (or shall or mandatory).

- 208 – These basic features shall be implemented to comply with OIC Core Architecture. The phrases  
209 “shall not”, and "PROHIBITED" indicate behaviour that is prohibited, i.e. that if performed means  
210 the implementation is not in compliance.

211 Recommended (or should).

- 212 – These features add functionality supported by OIC Core Architecture and should be  
213 implemented. Recommended features take advantage of the capabilities OIC Core Architecture,  
214 usually without imposing major increase of complexity. Notice that for compliance testing, if a  
215 recommended feature is implemented, it shall meet the specified requirements to be in  
216 compliance with these guidelines. Some recommended features could become requirements in  
217 the future. The phrase "should not" indicates behaviour that is permitted but not recommended.

218 Allowed (or allowed).

- 219 – These features are neither required nor recommended by OIC Core Architecture, but if the  
220 feature is implemented, it shall meet the specified requirements to be in compliance with these  
221 guidelines.

- 222 – Conditionally allowed (CA)The definition or behaviour depends on a condition. If the specified  
223 condition is met, then the definition or behaviour is allowed, otherwise it is not allowed.

224 Conditionally required (CR)

- 225 – The definition or behaviour depends on a condition. If the specified condition is met, then the  
226 definition or behaviour is required. Otherwise the definition or behaviour is allowed as default  
227 unless specifically defined as not allowed.

228 DEPRECATED

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

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

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

237 **5 Introduction**

238 **5.1 Translation between OCF and Non-OCF ecosystem - primitive concept of Bridging**

239 The details of Bridging may be implemented in many ways, for example, by using a Bridge Platform  
240 with an entity handler to interface directly to a Non-OCF device as shown in Figure 1.



241

242 **Figure 1 – Server bridging to Non- OCF device**

243 On start-up the Bridge Platform runs the entity handlers which discover the non-OCF systems (e.g.,  
244 Heart Rate Sensor Device) and create Resources for each Device or functionality discovered. The  
245 entity handler creates a Resource for each discovered Device or functionality and binds itself to  
246 that Resource. These Resources are made discoverable by the Bridge Platform.

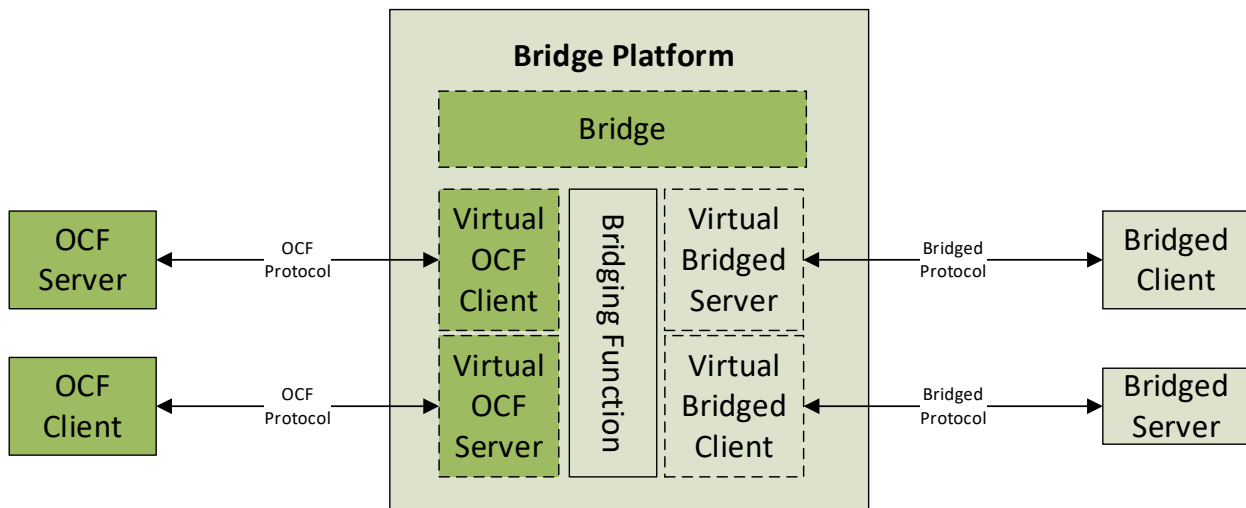
247 Once the Resources are created and made discoverable, then the Client Device can discover these  
248 Resources and operate on them using the mechanisms described in ISO/IEC 30118-1:2018. The  
249 requests to a Resource on the Bridge Platform are then interpreted by the entity handler and  
250 forwarded to the non-OCF device using the protocol supported by the non-OCF device. The  
251 returned information from the non-OCF device is then mapped to the appropriate response for that  
252 Resource.

253 Current OCF Bridging architecture implements the entity handler in the form of VOD.

254

255 **5.2 Bridge Platform**

256 This clause describes the functionality of a Bridge Platform; such a device is illustrated in Figure 2.

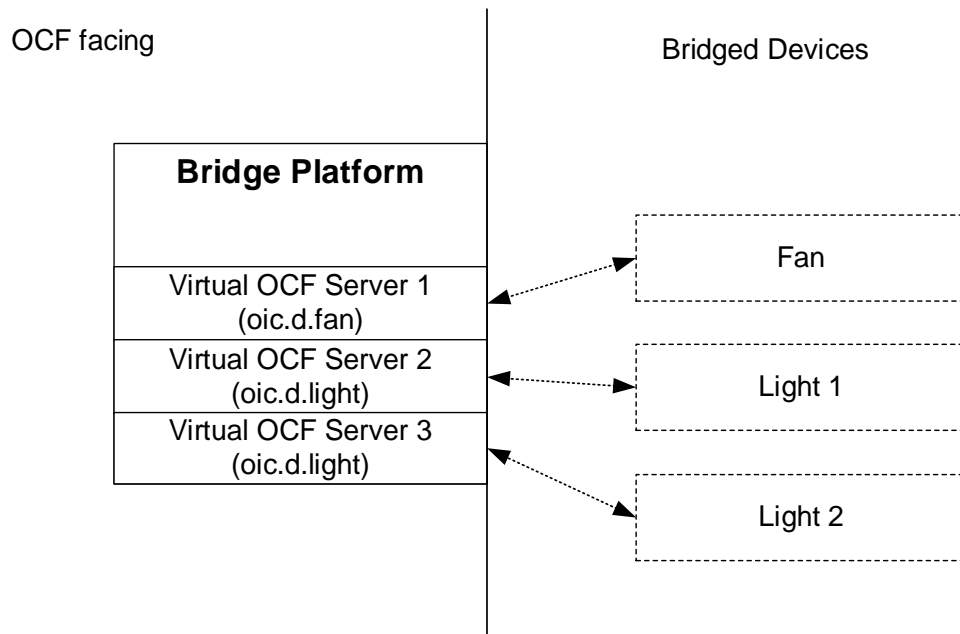


257

258 **Figure 2 – Bridge Platform components**

259 A Bridge Platform enables the representation of one or more Bridged Devices as Virtual OCF  
 260 Devices (VODs) on the network and/or enables the representation of one or more OCF Devices as  
 261 Virtual OCF Devices using another protocol on the network. The Bridged Devices themselves are  
 262 out of the scope of this document. The only difference between a native OCF Device and a VOD  
 263 from the perspective of an OCF Client is the inclusion of "oic.d.virtual" in the "rt" of "/oic/d" of the  
 264 VOD.

265 A Bridge Platform exposes a Bridge Device which is an OCF Device with a Device Type of  
 266 "oic.d.bridge". This provides to an OCF Client an explicit indication that the discovered Device is  
 267 performing a bridging function. This is useful for several reasons; 1) when establishing a home  
 268 network, the Client can determine that the bridge is reachable and functional when no bridged  
 269 devices are present, 2) allows for specific actions to be performed on the bridge considering the  
 270 known functionality a bridge supports, 3) allows for explicit discovery of all devices that are serving  
 271 a bridging function which benefits trouble shooting and maintenance actions on behalf of a user.  
 272 When such a device is discovered the exposed Resources on the OCF Bridge Device describe  
 273 other devices. For example, as shown in Figure 3.



274

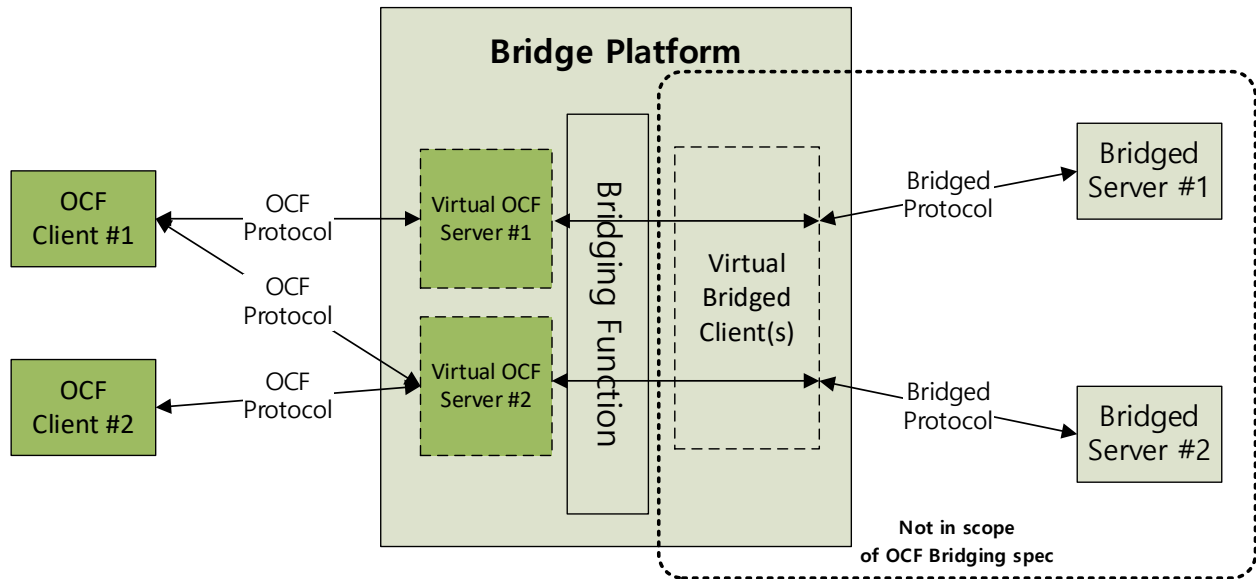
275 **Figure 3 – Schematic overview of a Bridge Platform bridging non-OCF devices**

276 It is expected that the Bridge Platform creates a set of devices during the start-up of the Bridge  
 277 Platform, these being the Bridge and any known VODs. The exposed set of VODs can change as  
 278 Bridged Devices are added or removed from the bridge. The adding and removing of Bridged  
 279 Devices is implementation dependent.

280 **5.3 Symmetric vs. asymmetric bridging**

281 There are two kinds of bridging: Symmetric, Asymmetric. In symmetric bridging, a bridge device  
 282 exposes OCF server(s) to another ecosystem and exposes other ecosystem's server(s) to OCF. In  
 283 asymmetric bridging, a bridge device exposes OCF server(s) to another ecosystem or exposes  
 284 another ecosystem's server(s) to OCF, but not both. The former case is called an Asymmetric  
 285 Server Bridge (see Figure 4), the latter case is called an Asymmetric Client Bridge (see Figure 5)

286



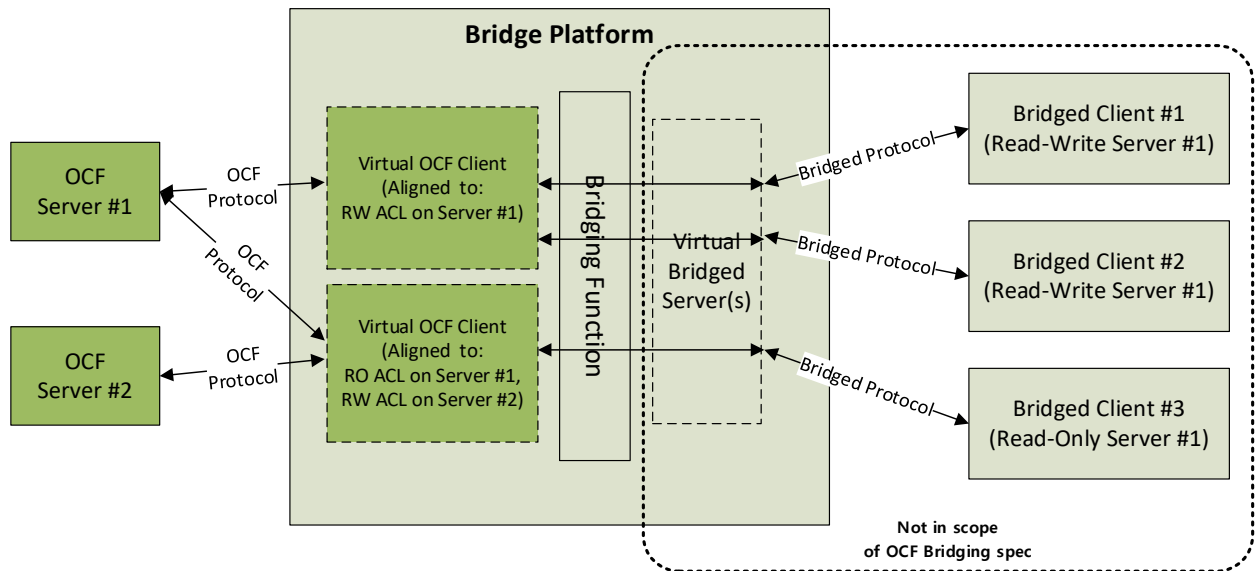
287

288

**Figure 4 – Asymmetric server bridge**

289 In Figure 4 each Bridged Server is exposed as a Virtual OCF Server to OCF side. These Virtual  
290 OCF Servers are same as normal OCF Servers except that they have additional rt value  
291 ("oic.d.virtual") for "/oic/d". The details of the Virtual Bridged Client are not in scope of this  
292 document.

293



294

295

**Figure 5 – Asymmetric client bridge**

296 Figure 5 shows that each access to the OCF Server is modelled as a Virtual OCF Client. Those  
297 accesses can be aggregated if their target OCF servers and access permissions are same,  
298 therefore a Virtual OCF Client can tackle multiple Bridged Clients.

## 299 **5.4 General requirements**

### 300 **5.4.1 Requirements common to all Bridge Platforms**

301 A VOD shall have a Device Type that contains "oic.d.virtual". This allows Bridge Platforms to  
302 determine if a device is already being translated when multiple Bridge Platforms are present or  
303 Clients to determine if corresponding Server is a VOD or not.

304 Each Bridged Device shall be exposed as a separate Virtual OCF Server or Client, with its own  
305 OCF Endpoint, and set of mandatory Resources (as defined in ISO/IEC 30118-1:2018 and ISO/IEC  
306 30118-2:2018).

307 Discovery of a VOD is the same as for an ordinary OCF Device; that is the VOD shall respond to  
308 multicast discovery requests. This allows platform-specific, device-specific, and resource-specific  
309 fields to all be preserved across translation.

310 The Bridge Introspection Device Data (IDD) provides information for the Resources exposed by the  
311 Bridge only. Each VOD shall expose an instance of "oic.wk.introspection" which provides a URL to  
312 an IDD for the specific VOD.

### 313 **5.4.2 Requirements specific to Symmetric Bridge Platforms**

314 In addition to the requirements mentioned in 5.4.1, Symmetric Bridging shall satisfy following  
315 requirements.

316 The Bridge Platform shall check the protocol-independent UUID ("piid" in OCF) of each device and  
317 shall not advertise back into a Bridged Protocol a device originally seen via that Bridged Protocol.  
318 The Bridge Platform shall stop translating any Bridged Protocol device exposed in OCF via another  
319 Bridge Platform if the Bridge Platform sees the device via the Bridged Protocol. Similarly, the Bridge  
320 Platform shall not advertise an OCF Device back into OCF, and the Bridge Platform shall stop  
321 translating any OCF device exposed in the Bridged Protocol via another Bridge Platform if the  
322 Bridge Platform sees the device via OCF. These require that the Bridge Platform can determine  
323 when a device is already being translated. A VOD shall be indicated on the OCF Security Domain  
324 with a Device Type of "oic.d.virtual". How a Bridge Platform determines if a device is already being  
325 translated on a non-OCF Security Domain is described in the relevant mapping specification for  
326 the Bridged Protocol.

327 The Bridge Platform shall detect duplicate VODs (with the same protocol-independent UUID)  
328 present in a network and shall not create more than one corresponding virtual device as it translates  
329 those duplicate devices into another network.

## 330 **5.5 VOD List**

331 For maintenance purposes, the Bridge maintains a list of VODs. This list includes Virtual OCF  
332 Servers and Virtual OCF Clients created by the Bridge Platform and subsequently on-boarded, as  
333 specified in ISO/IEC 30118-2:2018. A single instance of the Resource Type that defines the VOD  
334 list (see clause 7.2) shall be exposed by the Bridge. Please refer to ISO/IEC 30118-2:2018 for  
335 detailed operational requirements for the VOD list.

## 336 **5.6 Resource discovery**

337 A Bridge Platform shall detect devices that arrive and leave the Bridged network or the OCF  
338 Security Domain. Where there is no pre-existing mechanism to reliably detect the arrival and  
339 departure of devices on a network, a Bridge Platform shall periodically poll the network to detect  
340 the arrival and departure of devices, for example using COAP multicast discovery (a multicast  
341 RETRIEVE of "/oic/res") in the case of the OCF Security Domain. Bridge Platform implementations  
342 are encouraged to use a poll interval of 30 seconds plus or minus a random delay of a few seconds.

343 A Bridge Platform and any exposed VODs shall each respond to network discovery commands.  
344 The response to a RETRIEVE on "/oic/res" shall only include the devices that match the RETRIEVE  
345 request.

346 For example, if a Bridge exposes VODs for the fan and lights shown in Figure 3, and an OCF Client  
347 performs a discovery request with a content format of "application/vnd.ocf+cbor", there will be four  
348 discrete responses, one for the Bridge, one for the virtual fan Device, and two for the virtual light  
349 Devices. Note that what is returned is not in the JSON format but in a suitable encoding as defined  
350 in ISO/IEC 30118-1:2018.

351 Response from the Bridge:

```
352 [
353   {
354     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
355     "href": "/oic/res",
356     "rel": "self",
357     "rt": ["oic.wk.res"],
358     "if": ["oic.if.ll", "oic.if.baseline"],
359     "p": {"bm": 3},
360     "eps": [{"ep": "coap://[2001:db8:a::b1d4]:5555"},
361             {"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
362   },
363   {
364     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
365     "href": "/oic/d",
366     "rt": ["oic.wk.d", "oic.d.bridge"],
367     "if": ["oic.if.r", "oic.if.baseline"],
368     "p": {"bm": 3},
369     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
370   },
371   {
372     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
373     "href": "/oic/p",
374     "rt": ["oic.wk.p"],
375     "if": ["oic.if.r", "oic.if.baseline"],
376     "p": {"bm": 3},
377     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
378   },
379   {
380     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
381     "href": "/oic/sec/doxm",
382     "rt": ["oic.r.doxm"],
383     "if": ["oic.if.baseline"],
384     "p": {"bm": 1},
385     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
386   },
387   {
388     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
389     "href": "/oic/sec/pstat",
390     "rt": ["oic.r.pstat"],
391     "if": ["oic.if.baseline"],
392     "p": {"bm": 1},
393     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
394   },
395   {
396     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
397     "href": "/oic/sec/cred",
398     "rt": ["oic.r.cred"],
399     "if": ["oic.if.baseline"],
400     "p": {"bm": 1},
401     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
```

```

402   },
403   {
404     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
405     "href": "/oic/sec/acl2",
406     "rt": ["oic.r.acl2"],
407     "if": ["oic.if.baseline"],
408     "p": {"bm": 1},
409     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
410   },
411   {
412     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
413     "href": "/myIntrospection",
414     "rt": ["oic.wk.introspection"],
415     "if": ["oic.if.r", "oic.if.baseline"],
416     "p": {"bm": 3},
417     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
418   },
419   {
420     "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
421     "href": "/myVodlist",
422     "rt": ["oic.r.vodlist "],
423     "if": ["oic.if.r", "oic.if.baseline"],
424     "p": {"bm": 3},
425     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
426   }
427 ]

```

428  
429 Response from the Fan VOD:

```

430 [
431   {
432     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
433     "href": "/oic/res",
434     "rt": ["oic.wk.res"],
435     "if": ["oic.if.ll", "oic.if.baseline"],
436     "p": {"bm": 3},
437     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
438   },
439   {
440     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
441     "href": "/oic/d",
442     "rt": ["oic.wk.d", "oic.d.fan", "oic.d.virtual"],
443     "if": ["oic.if.r", "oic.if.baseline"],
444     "p": {"bm": 3},
445     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
446   },
447   {
448     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
449     "href": "/oic/p",
450     "rt": ["oic.wk.p"],
451     "if": ["oic.if.r", "oic.if.baseline"],
452     "p": {"bm": 3},
453     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
454   },
455   {
456     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
457     "href": "/myFan",
458     "rt": ["oic.r.switch.binary"],
459     "if": ["oic.if.a", "oic.if.baseline"],
460     "p": {"bm": 3},
461     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
462   },
463   {

```

```

464     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
465     "href": "/oic/sec/doxm",
466     "rt": ["oic.r.doxm"],
467     "if": ["oic.if.baseline"],
468     "p": {"bm": 1},
469     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
470 },
471 {
472     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
473     "href": "/oic/sec/pstat",
474     "rt": ["oic.r.pstat"],
475     "if": ["oic.if.baseline"],
476     "p": {"bm": 1},
477     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
478 },
479 {
480     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
481     "href": "/oic/sec/cred",
482     "rt": ["oic.r.cred"],
483     "if": ["oic.if.baseline"],
484     "p": {"bm": 1},
485     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
486 },
487 {
488     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
489     "href": "/oic/sec/acl2",
490     "rt": ["oic.r.acl2"],
491     "if": ["oic.if.baseline"],
492     "p": {"bm": 1},
493     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
494 },
495 {
496     "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
497     "href": "/myFanIntrospection",
498     "rt": ["oic.wk.introspection"],
499     "if": ["oic.if.r", "oic.if.baseline"],
500     "p": {"bm": 3},
501     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
502 }
503 ]
504
505 Response from the first Light VOD:
506 [
507 {
508     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
509     "href": "/oic/res",
510     "rt": ["oic.wk.res"],
511     "if": ["oic.if.ll", "oic.if.baseline"],
512     "p": {"bm": 3},
513     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
514 },
515 {
516     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
517     "href": "/oic/d",
518     "rt": ["oic.wk.d", "oic.d.light", "oic.d.virtual"],
519     "if": ["oic.if.r", "oic.if.baseline"],
520     "p": {"bm": 3},
521     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
522 },
523 {
524     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
525     "href": "/oic/p",

```



```

526     "rt": ["oic.wk.p"],
527     "if": ["oic.if.r", "oic.if.baseline"],
528     "p": {"bm": 3},
529     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
530 },
531 {
532     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
533     "href": "/myLight",
534     "rt": ["oic.r.switch.binary"],
535     "if": ["oic.if.a", "oic.if.baseline"],
536     "p": {"bm": 3},
537     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
538 },
539 {
540     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
541     "href": "/oic/sec/doxm",
542     "rt": ["oic.r.doxm"],
543     "if": ["oic.if.baseline"],
544     "p": {"bm": 1},
545     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
546 },
547 {
548     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
549     "href": "/oic/sec/pstat",
550     "rt": ["oic.r.pstat"],
551     "if": ["oic.if.baseline"],
552     "p": {"bm": 1},
553     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
554 },
555 {
556     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
557     "href": "/oic/sec/cred",
558     "rt": ["oic.r.cred"],
559     "if": ["oic.if.baseline"],
560     "p": {"bm": 1},
561     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
562 },
563 {
564     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
565     "href": "/oic/sec/acl2",
566     "rt": ["oic.r.acl2"],
567     "if": ["oic.if.baseline"],
568     "p": {"bm": 1},
569     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
570 },
571 {
572     "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
573     "href": "/myLightIntrospection",
574     "rt": ["oic.wk.introspection"],
575     "if": ["oic.if.r", "oic.if.baseline"],
576     "p": {"bm": 3},
577     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
578 }
579 ]
580
581 Response from the second Light VOD:
582 [
583 {
584     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
585     "href": "/oic/res",
586     "rt": ["oic.wk.res"],
587     "if": ["oic.if.ll", "oic.if.baseline"],

```

```

588     "p": {"bm": 3},
589     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
590   },
591   {
592     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
593     "href": "/oic/d",
594     "rt": ["oic.wk.d", "oic.d.light", "oic.d.virtual"],
595     "if": ["oic.if.r", "oic.if.baseline"],
596     "p": {"bm": 3},
597     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
598   },
599   {
600     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
601     "href": "/oic/p",
602     "rt": ["oic.wk.p"],
603     "if": ["oic.if.r", "oic.if.baseline"],
604     "p": {"bm": 3},
605     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
606   },
607   {
608     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
609     "href": "/myLight",
610     "rt": ["oic.r.switch.binary"],
611     "if": ["oic.if.a", "oic.if.baseline"],
612     "p": {"bm": 3},
613     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
614   },
615   {
616     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
617     "href": "/oic/sec/doxm",
618     "rt": ["oic.r.doxm"],
619     "if": ["oic.if.baseline"],
620     "p": {"bm": 1},
621     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
622   },
623   {
624     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
625     "href": "/oic/sec/pstat",
626     "rt": ["oic.r.pstat"],
627     "if": ["oic.if.baseline"],
628     "p": {"bm": 1},
629     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
630   },
631   {
632     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
633     "href": "/oic/sec/cred",
634     "rt": ["oic.r.cred"],
635     "if": ["oic.if.baseline"],
636     "p": {"bm": 1},
637     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
638   },
639   {
640     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
641     "href": "/oic/sec/acl2",
642     "rt": ["oic.r.acl2"],
643     "if": ["oic.if.baseline"],
644     "p": {"bm": 1},
645     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
646   },
647   {
648     "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
649     "href": "/myLightIntrospection",

```

```

650     "rt": ["oic.wk.introspection"],
651     "if": ["oic.if.r", "oic.if.baseline"],
652     "p": {"bm": 3},
653     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:4444"}]
654 }
655 ]

```

656 **Figure 6 – /oic/res example responses**

657 **5.7 "Deep translation" vs. "on-the-fly"**

658 When translating a service between a Bridged Protocol (e.g., AllJoyn) and OCF protocols, there  
659 are two possible types of translation. Bridge Platforms are expected to dedicate most of their logic  
660 to "deep translation" types of communication, in which data models used with the Bridged Protocol  
661 are mapped to the equivalent OCF Resource Types and vice-versa, in such a way that a compliant  
662 OCF Client or Bridged Client would be able to interact with the service without realising that a  
663 translation was made.

664 "Deep translation" is out of the scope of this document, as the procedure far exceeds mapping of  
665 types. For example, clients on one side of a Bridge Platform may decide to represent an intensity  
666 as an 8-bit value between 0 and 255, whereas the devices on the other may have chosen to  
667 represent that as a floating-point number between 0.0 and 1.0. It's also possible that the procedure  
668 may require storing state in the Bridge Platform. Either way, the programming of such translation  
669 will require dedicated effort and study of the mechanisms on both sides.

670 The other type of translation, the "on-the-fly" or "one-to-one" translation, requires no prior  
671 knowledge of the device-specific schema in question on the part of the Bridge Platform. The burden  
672 is, instead, on one of the other participants in the communication, usually the client application.  
673 That stems from the fact that "on-the-fly" translation always produces Bridged Resource Types and  
674 OCF Resource Types as vendor extensions.

675 **5.8 Security**

676 Please refer to ISO/IEC 30118-2:2018 for security specific requirements as they pertain to a Bridge  
677 Platform. These security requirements include both universal requirements applicable to all Bridged  
678 Protocols, and additional security requirements specific to each Bridged Protocol.

679 **6 Device type definitions**

680 The required Resource Types are listed in Table 1.

681 **Table 1 – Device type definitions**

Device Name (informative)	Device Type ("rt") (Normative)	Required Resource name	Required Resource Type
Bridge	oic.d.bridge	VOD List	oic.r.vodlist
Virtual Device	oic.d.virtual	Device	oic.wk.d

682 **7 Resource type definitions**

683 **7.1 List of resource types**

684 Table 2 lists the Resource Types defined in this document.

**Table 2 – Alphabetical list of resource types**

Friendly Name (informative)	Resource Type (rt)	Clause
VOD List	oic.r.vodlist	10.4

686

**7.2 VOD List****7.2.1 Introduction**

This Resource describes the VODs that have been onboarded on the Bridge Platform.

**7.2.2 Example URI**

/VODListResURI

**7.2.3 Resource type**

The Resource Type is defined as: "oic.r.vodlist".

**7.2.4 OpenAPI 2.0 definition**

```

695 {
696   "swagger": "2.0",
697   "info": {
698     "title": "VOD List",
699     "version": "2019-05-16",
700     "license": {
701       "name": "OCF Data Model License",
702       "url":
703         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
704         CENSE.md",
705       "x-copyright": "Copyright 2019 Open Connectivity Foundation, Inc. All rights reserved."
706     },
707     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
708   },
709   "schemes": ["http"],
710   "consumes": ["application/json"],
711   "produces": ["application/json"],
712   "paths": {
713     "/VODListResURI" : {
714       "get": {
715         "description": "This Resource describes the VODs that have been onboarded on the Bridge
716         Platform.\n",
717         "parameters": [
718           {"$ref": "#/parameters/interface"}
719         ],
720         "responses": {
721           "200": {
722             "description" : "Example response payload",
723             "x-example":
724             {
725               "rt": ["oic.r.vodlist"],
726               "vods": [
727                 {
728                   "n": "Smoke sensor",
729                   "di": "54919CA5-4101-4AE4-595B-353C51AA1234",
730                   "econame": "Z-Wave"
731                 },
732                 {
733                   "n": "Thermostat",
734                   "di": "54919CA5-4101-4AE4-595B-353C51AA5678",
735                   "econame": "Zigbee"
736                 }
737               ]
738             },
739           "schema": { "$ref": "#/definitions/vodlist" }
740         }
741       }
742     }

```

```

741     }
742   }
743 }
744 },
745 "parameters": {
746   "interface": {
747     "in": "query",
748     "name": "if",
749     "type": "string",
750     "enum": ["oic.if.r", "oic.if.baseline"]
751   }
752 },
753 "definitions": {
754   "vodentry": {
755     "description": "Information for a VOD created by the Bridge",
756     "type": "object",
757     "properties": {
758       "n": {
759         "$ref":
760 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
761 schema.json#/definitions/n"
762       },
763       "di": {
764         "$ref": "https://openconnectivityfoundation.github.io/core/schemas/oic.types-
765 schema.json#/definitions/uuid"
766       },
767       "econame": {
768         "description": "Ecosystem Name of the Bridged Device which is exposed by this VOD",
769         "type": "string",
770         "enum": [ "BLE", "oneM2M", "UPlus", "Zigbee", "Z-Wave" ],
771         "readOnly": true
772       }
773     },
774     "required": ["n", "di", "econame"]
775   },
776   "vodlist": {
777     "type": "object",
778     "properties": {
779       "n": {
780         "$ref":
781 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
782 schema.json#/definitions/n"
783       },
784       "rt": {
785         "description": "Resource Type",
786         "items": {
787           "maxLength": 64,
788           "type": "string",
789           "enum": ["oic.r.vodlist"]
790         },
791         "minItems": 1,
792         "uniqueItems": true,
793         "readOnly": true,
794         "type": "array"
795       },
796       "id": {
797         "$ref":
798 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
799 schema.json#/definitions/id"
800       },
801       "if": {
802         "description": "The OCF Interface set supported by this Resource",
803         "items": {
804           "enum": [
805             "oic.if.baseline",
806             "oic.if.r"
807           ],
808           "type": "string"
809         },
810         "minItems": 2,

```

```

811         "uniqueItems": true,
812         "readOnly": true,
813         "type": "array"
814     },
815     "vods": {
816         "description": "Array of information per VOD created by the Bridge",
817         "type": "array",
818         "minItems": 0,
819         "uniqueItems": true,
820         "readOnly": true,
821         "items": {
822             "$ref": "#/definitions/vodentry"
823         }
824     }
825 },
826 "required": ["vods"]
827 }
828 }
829 }
830

```

### 831 7.2.5 Property definition

832 Table 3 defines the Properties that are part of the "oic.r.vodlist" Resource Type.

833 **Table 3 – The Property definitions of the Resource with type "rt" = "oic.r.vodlist".**

Property name	Value type	Mandatory	Access mode	Description
if	array: see schema	No	Read Only	The OCF Interface set supported by this Resource
vods	array: see schema	Yes	Read Only	Array of information per VOD created by the Bridge
id	multiple types: see schema	No	Read Write	
n	multiple types: see schema	No	Read Write	
rt	array: see schema	No	Read Only	Resource Type
econame	string	Yes	Read Only	Ecosystem Name of the Bridged Device which is exposed by this VOD
n	multiple types: see schema	Yes	Read Write	
di	multiple types: see schema	Yes	Read Write	

### 834 7.2.6 CRUDN behaviour

835 Table 4 defines the CRUDN operations that are supported on the "oic.r.vodlist" Resource Type.

836 **Table 4 – The CRUDN operations of the Resource with type "rt" = "oic.r.vodlist".**

Create	Read	Update	Delete	Notify
	get			observe

837

