

# OCF Core Specification Extension

## WiFi Easy Setup

VERSION 1.3.0 | December 2017



**OPEN** CONNECTIVITY  
FOUNDATION™

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

Copyright Open Connectivity Foundation, Inc. © 2017.  
All Rights Reserved.

## Legal Disclaimer

2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19

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 Open Connectivity Foundation, Inc. All rights reserved.

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

# CONTENTS

20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61

1	Scope .....	7
2	Normative references .....	7
3	Terms, definitions, symbols and abbreviations .....	8
3.1	Terms and definitions .....	8
3.2	Conventions .....	8
3.3	Data types .....	8
4	Document conventions and organization .....	9
5	Overview .....	10
5.1	Introduction .....	10
5.2	Architecture .....	10
5.3	Example Scenario .....	10
6	Resource model .....	11
6.1	Introduction .....	11
6.2	EasySetup Resource .....	11
6.2.1	Overview .....	11
6.2.2	Resource .....	11
6.3	WiFiConf Resource Type .....	12
6.3.1	Introduction .....	12
6.3.2	Resource Type .....	12
6.4	DevConf Resource Type .....	13
6.4.1	Introduction .....	13
6.4.2	Resource Type .....	14
7	Network and connectivity .....	15
8	Functional interactions .....	16
8.1	Onboarding, Provisioning and Configuration .....	16
8.2	Resource discovery .....	16
8.3	Retrieving and Updating Easy Setup Resources .....	16
8.4	Error Handling .....	16
8.5	Example Easy Setup Flow .....	17
9	Security .....	20
Annex A (normative)	Resource Type definitions .....	21
A.1	List of Resource Type definitions .....	21
A.2	Easy Setup Collection Baseline Interface .....	21
A.2.1	Introduction .....	21
A.2.2	Example URI .....	21
A.2.3	Resource Type .....	21
A.2.4	RAML Definition .....	21
A.2.5	Property Definition .....	24
A.2.6	CRUDN behavior .....	25
A.3	Wi-Fi Configuration Resource Baseline Interface .....	25

62	A.3.1	Introduction .....	25
63	A.3.2	Example URI .....	25
64	A.3.3	Resource Type .....	25
65	A.3.4	RAML Definition .....	25
66	A.3.5	Property Definition .....	29
67	A.3.6	CRUDN behavior .....	29
68	A.4	Device Configuration .....	29
69	A.4.1	Introduction .....	29
70	A.4.2	Example URI .....	29
71	A.4.3	Resource Type .....	29
72	A.4.4	RAML Definition .....	30
73	A.4.5	Property Definition .....	31
74	A.4.6	CRUDN behavior .....	31
75	Annex B (informative)	Swagger2.0 definitions .....	32
76	B.1	Device Configuration .....	32
77	B.1.1	Introduction .....	32
78	B.1.2	Wellknown URI .....	32
79	B.1.3	Resource Type .....	32
80	B.1.4	Swagger2.0 Definition .....	32
81	B.1.5	Property Definition .....	34
82	B.1.6	CRUDN behaviour .....	34
83	B.2	Easy Setup Collection .....	34
84	B.2.1	Introduction .....	34
85	B.2.2	Wellknown URI .....	34
86	B.2.3	Resource Type .....	34
87	B.2.4	Swagger2.0 Definition .....	34
88	B.2.5	Property Definition .....	46
89	B.2.6	CRUDN behaviour .....	49
90	B.3	Wi-Fi Configuration Resource .....	49
91	B.3.1	Introduction .....	49
92	B.3.2	Wellknown URI .....	49
93	B.3.3	Resource Type .....	49
94	B.3.4	Swagger2.0 Definition .....	49
95	B.3.5	Property Definition .....	54
96	B.3.6	CRUDN behaviour .....	54
97			
98			

99  
100  
101  
102  
103  
104  
105

## Figures

Figure 1. Easy Setup deployment architecture .....	10
--	----

## Tables

106  
107

108	Table 1. EasySetup Resource Type.....	11
109	Table 2. “oic.r.easyssetup” Resource Type definition .....	11
110	Table 3. WiFiConf Resource Type .....	13
111	Table 4. “oic.r.wificonf” Resource Type definition .....	13
112	Table 5. DevConf Resource Type .....	14
113	Table 6. “oic.r.devconf” Resource Type definition .....	14
114	Table 7. Alphabetized list of resources .....	21

115  
116

## 117 **1 Scope**

118 This specification defines functional extensions to the capabilities defined in the OCF Core  
119 Specification to meet the requirements of Wi-Fi Easy Setup. This specification specifies new  
120 Resource Types to enable the functionality and any extensions to the existing capabilities defined  
121 in the OCF Core Specification.

## 122 **2 Normative references**

123 The following documents, in whole or in part, are normatively referenced in this document and are  
124 indispensable for its application. For dated references, only the edition cited applies. For undated  
125 references, the latest edition of the referenced document (including any amendments) applies.

126 OCF Core Specification, *Open Connectivity Foundation Core Specification*, Version 1.3

127 Available at: [https://openconnectivity.org/specs/OCF\\_Core\\_Specification\\_v1.3.0.pdf](https://openconnectivity.org/specs/OCF_Core_Specification_v1.3.0.pdf)

128 Latest version available at: [https://openconnectivity.org/specs/OCF\\_Core\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Core_Specification.pdf)

129 OCF Security Specification, *Open Connectivity Foundation Security Capabilities*, Version 1.3

130 Available at: [https://openconnectivity.org/specs/OCF\\_Security\\_Specification\\_v1.3.0.pdf](https://openconnectivity.org/specs/OCF_Security_Specification_v1.3.0.pdf)

131 Latest version available at: [https://openconnectivity.org/specs/OCF\\_Security\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Security_Specification.pdf)

132 IEEE 802.11:2016, IEEE Standard for Information technology—Telecommunications and  
133 information exchange between systems Local and metropolitan area networks—Specific  
134 requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY)  
135 Specifications, December 2016

136 <https://standards.ieee.org/findstds/standard/802.11-2016.html>

137 IETF RFC 7159, *The JavaScript Object Notation (JSON) Data Interchange Format*, March 2014

138 <https://www.rfc-editor.org/info/rfc7159>

139 IETF RFC 7252, *The Constrained Application Protocol (CoAP)*, June 2014

140 <https://www.rfc-editor.org/info/rfc7252>

141 JSON Schema Validation, *JSON Schema: interactive and non-interactive validation*, January 2013

142 <http://json-schema.org/latest/json-schema-validation.html>

143 OpenAPI specification, *aka Swagger RESTful API Documentation Specification*, Version 2.0

144 <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md>

145

146 **3 Terms, definitions, symbols and abbreviations**

147 All terms and definitions as defined in the OCF Core Specification also apply to this specification.

148 **3.1 Terms and definitions**

149 As defined in the OCF Core Specification with the following additions

150 **3.1.1.**

151 **Easy Setup Enrollment**

152 Easy Setup Enrollment is a step during Easy Setup in which the Enrollee is contacted by the  
153 Mediator to configure the Enroller's information by means of accessing Easy Setup Resources.

154 **3.1.2.**

155 **Enrollee**

156 The Device that needs to be configured and connected. E.g. Air-conditioner, Printer.

157 **3.1.3.**

158 **Enroller**

159 Target network entity to which the Enrollee connects. E.g. Wi-Fi AP

160 **3.1.4.**

161 **Mediator**

162 Device that enables the Enrollee to connect to the target network (Enroller). The Mediator transfers  
163 configuration information to the Enrollee. E.g. Mobile Phone

164 **3.1.5.**

165 **Easy Setup**

166 Process of configuring an Enrollee using Mediator (by transferring of essential information to the  
167 Enrollee).

168 **3.1.6.**

169 **Soft AP**

170 Software Enabled Access Point hosted on the device which is not a dedicated Access Point.

171 **3.2 Conventions**

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

176 **3.3 Data types**

177 As defined in the OCF Core Specification.

178



#### 179 **4 Document conventions and organization**

180 In this document, features are described as required, recommended, allowed or DEPRECATED as  
181 follows:

182 Required (or shall or mandatory)(M).

- 183 • These basic features shall be implemented to comply with Core Architecture. The phrases  
184 “shall not”, and “PROHIBITED” indicate behaviour that is prohibited, i.e. that if performed  
185 means the implementation is not in compliance.

186 Recommended (or should)(S).

- 187 • These features add functionality supported by Core Architecture and should be implemented.  
188 Recommended features take advantage of the capabilities Core Architecture, usually without  
189 imposing major increase of complexity. Notice that for compliance testing, if a recommended  
190 feature is implemented, it shall meet the specified requirements to be in compliance with these  
191 guidelines. Some recommended features could become requirements in the future. The phrase  
192 “should not” indicates behaviour that is permitted but not recommended.

193 Allowed (may or allowed)(O).

- 194 • These features are neither required nor recommended by Core Architecture, but if the feature  
195 is implemented, it shall meet the specified requirements to be in compliance with these  
196 guidelines.

197 DEPRECATED.

- 198 • Although these features are still described in this specification, they should not be implemented  
199 except for backward compatibility. The occurrence of a deprecated feature during operation of  
200 an implementation compliant with the current specification has no effect on the  
201 implementation’s operation and does not produce any error conditions. Backward compatibility  
202 may require that a feature is implemented and functions as specified but it shall never be used  
203 by implementations compliant with this specification.

204 Conditionally allowed (CA)

- 205 • The definition or behaviour depends on a condition. If the specified condition is met, then the  
206 definition or behaviour is allowed, otherwise it is not allowed.

207 Conditionally required (CR)

- 208 • The definition or behaviour depends on a condition. If the specified condition is met, then the  
209 definition or behaviour is required. Otherwise the definition or behaviour is allowed as default  
210 unless specifically defined as not allowed.

211

212 Strings that are to be taken literally are enclosed in “double quotes”.

213 Words that are emphasized are printed in italic.

214

215 **5 Overview**

216 **5.1 Introduction**

217 This specification describes a way to setup and configure a new OCF Device, using an already  
218 configured OCF Device or onboarding tool.

219 The described setup and configure mechanism is optional and other mechanisms are allowed to  
220 be used.

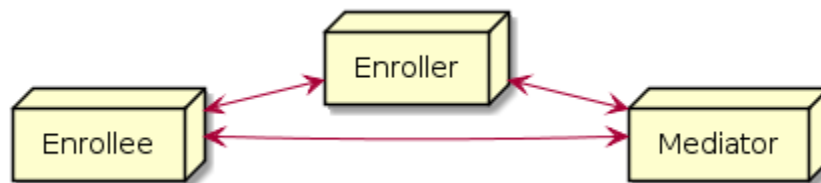
221 Specifically, this method allows the transferring of essential information to the new Device, which  
222 includes:

- 223 • Local network connection information, e.g. in case of Wi-Fi it will be Wi-Fi access point  
224 information.
- 225 • Device Configuration: Additional Device configuration information.

226 Easy Setup can be enhanced in future by incorporating other suitable technologies.

227 **5.2 Architecture**

228 Figure 1 shows the deployment architectural approach.  
229



230

231 **Figure 1. Easy Setup deployment architecture**

232 Easy Setup defines the following roles: Enrollee, Enroller, and Mediator. Please refer to Section  
233 3.1 for definitions thereof.

234 **5.3 Example Scenario**

235 The following scenario presents a typical setup case.

236 The configuration information and steps taken may vary depending on the Device's type and status.

- 237 1. The Enrollee enters Easy Setup mode (when the Device is unboxed for the first time, it may  
238 be in this mode by default).
- 239 2. The Mediator discovers and connects to the Enrollee.
- 240 3. The Mediator performs Security Provisioning of the Enrollee.
- 241 4. The Mediator transmits Wi-Fi Setting Information to the Enrollee.
- 242 5. Using the information received from the Mediator, the Enrollee connects to the Enroller  
243 (Wi-Fi AP).

244

245 **6 Resource model**

246 **6.1 Introduction**

247 Devices capable of Easy Setup shall support the following Resource Types.

- 248 1. EasySetup Resource Type
- 249 2. WiFiConf Resource Type
- 250 3. DevConf Resource Type

251 The EasySetup Resource Type is a Collection Resource and shall contain Links to instances of at  
252 least WiFiConf and DevConf. A vendor may add links to other Resource Types.

253 Note that the EasySetup Resource Type supports the batch Interface (oic.if.b) which allows for  
254 efficient data delivery with a single request rather than multiple requests to each linked Resource.



255  
256 **Figure 2. Easy Setup Resource Types**

257 **6.2 EasySetup Resource**

258 **6.2.1 Overview**

259 The EasySetup Resource stores useful information including current status of Enrollee and last  
260 error code which was produced in the process of Easy Setup.

261 **6.2.2 Resource**

262 The Easy Setup Resource Type is as defined in Table 1. EasySetup Resource Type.

263 **Table 1. EasySetup Resource Type**

Example URI	Resource Type Title	Resource Type ID ("rt" value)	Interfaces	Description	Related Functional Interaction
/example/EasySetupResourceURI	EasySetup	oic.r.easysetup, oic.wk.col	oic.if.baseline, oic.if.ll, oic.if.b	Top level Resource for Easy Setup. Indicates easy setup status. The Resource properties exposed are listed in Table 2.	

264

265 Table 2. "oic.r.easysetup" Resource Type definition defines the details for the "oic.r.easysetup"  
266 Resource Type.

267 **Table 2. "oic.r.easysetup" Resource Type definition**

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
Easy Setup Provisioning Status	ps	integer	enum		R	Yes	Easy setup provisioning status of the Device 0: Need to Setup, 1: Connecting to Enroller, 2: Connected to Enroller,

							3: Failed to Connect to Enroller, 4~254: Reserved, 255: EOF
<b>Last Error Code</b>	lec	integer	enum		R	Yes	Indicates a failure reason if it fails to connect to Enroller 0: NO error, 1: Given SSID is not found, 2: Wi-Fi password is wrong, 3: IP address is not allocated, 4: NO internet connection, 5: Timeout, 6: Wi-Fi Auth Type is not supported by the Enrollee, 7: Wi-Fi Encryption Type is not supported by the Enrollee, 8: Wi-Fi Auth Type is wrong (failure while connecting to the Enroller), 9: Wi-Fi Encryption Type is wrong (failure while connecting to the Enroller), 10~254: Reserved, 255: Unknown error.
<b>Connect</b>	cn	array of integer			RW	Yes	Array of connection types to trigger Enrollee to initiate connection: 1 : Wi-Fi, 2 : Other transport to be added in a future (e.g. BLE))
<b>Links</b>	links	array			R	Yes	Array of links that are WiFiConf and DevConf Resource.

268 Enrollee shall set the following as default values (for example, when Device is unboxed first time):

- 269 • “ps” equal to 0.
- 270 • “lec” equal to 0.
- 271 • “cn” equal to an empty array.

## 272 **6.3 WiFiConf Resource Type**

### 273 **6.3.1 Introduction**

274 The WiFiConf Resource Type stores information to help an Enrollee to connect to an existing Wi-Fi AP.

### 276 **6.3.2 Resource Type**

277 The WiFiConf Resource Type is as defined in Table 3. WiFiConf Resource Type.

**Table 3. WiFiConf Resource Type**

Example URI	Resource Type Title	Resource Type ID ("rt" value)	Interfaces	Description	Related Functional Interaction
/example/WiFiConfResURI	WiFiConf	oic.r.wificonf	oic.if.baseline, oic.if.rw	Contains Wi-Fi related properties The Resource properties exposed are listed in Table 4.	

279

280 Table 4. "oic.r.wificonf" Resource Type definition defines the details for the "oic.r.wificonf"  
281 Resource Type.

282

**Table 4. "oic.r.wificonf" Resource Type definition**

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
<b>Supported Wi-Fi Mode Type</b>	swmt	array of string	enum		R	Yes	Supported Wi-Fi modes by Enrollee. Can be multiple. ("A", "B", "G", "N", "AC")
<b>Supported Wi-Fi Frequency</b>	swf	array of string	Refer to description for valid values.		R	Yes	Supported Wi-Fi frequencies by Enrollee. Can be multiple. ("2.4G", "5G")
<b>Target Network Name</b>	tnn	string			RW	Yes	Target network name (SSID of Wi-Fi AP i.e. enrollee)
<b>Credential</b>	cd	string			RW	No	Credential information of Wi-Fi AP (Password used to connect to enrollee).
<b>Wi-Fi Auth Type</b>	wat	string	enum		RW	Yes	Wi-Fi auth type ("None", "WEP", "WPA_PSK", "WPA2_PSK")
<b>Wi-Fi Encryption Type</b>	wet	string	enum		RW	Yes	Wi-Fi encryption type ("None", "WEP_64", "WEP_128", "TKIP", "AES", "TKIP_AES")
<b>Supported Wi-Fi Auth Type</b>	swat	array of string	enum		R	Yes	Supported Wi-Fi Auth types. Can be multiple. ("None", "WEP", "WPA_PSK", "WPA2_PSK")
<b>Supported Wi-Fi Encryption Type</b>	swet	array of string	enum		R	Yes	Supported Wi-Fi Encryption types. Can be multiple. ("None", "WEP-64", "WEP_128", "TKIP", "AES", "TKIP_AES")

283

## 284 6.4 DevConf Resource Type

### 285 6.4.1 Introduction

286 The DevConf Resource Type stores Device configuration information required in Wi-Fi Easy Setup.

287 **6.4.2 Resource Type**

288 The DevConf Resource Type is as defined in Table 5. DevConf Resource Type

289 **Table 5. DevConf Resource Type**

Example URI	Resource Type Title	Resource Type ID ("rt" value)	Interfaces	Description	Related Functional Interaction
/example/DevConfResURI	DevConf	oic.r.devconf	oic.if.base line, "oic.if.r"	Stores device configuration information required in Easy Setup process The Resource properties exposed are listed in Table 6.	

290

291 Table 6. "oic.r.devconf" Resource Type definition defines the details for the "oic.r.devconf"  
292 Resource Type.

293 **Table 6. "oic.r.devconf" Resource Type definition**

Property title	Property name	Value type	Value rule	Unit	Access mode	Mandatory	Description
<b>Device Name</b>	dn	one of: string or array of object			R	Yes	Indicates a pre-configured device name in language indicated by 'dl' in /oic/con. or An array of objects where each object has a 'language' field (containing an IETF RFC 5646 language tag) and a 'value' field containing the pre-configured device name in the indicated language. The pre-configured device name is presented by enrollee to mediator during easy-setup process.

294

295 **7 Network and connectivity**

296 Both the Mediator and Enrollee communicate via a common connectivity (e.g. Wi-Fi).

297 If using Wi-Fi for Easy Setup then the Enrollee shall have capability to act as a Soft AP. A Soft AP  
298 shall support the access point requirements defined by IEEE 802.11:2016.

299

## 300 **8 Functional interactions**

### 301 **8.1 Onboarding, Provisioning and Configuration**

302 Mediator may perform Ownership Transfer on the Enrollee and may also perform ACL provisioning.  
303 If it does so, the Mediator must conform to any existing DOXS and AMS requirements respectively.  
304 For details refer to the OCF Security Specification.

305

### 306 **8.2 Resource discovery**

307 The Mediator connects to the Enrollee via a mutually supported connection.

308 When in Easy Setup phase, if using Wi-Fi as the connectivity between the Enrollee and the  
309 Mediator then the Enrollee shall make itself discoverable as a Soft AP. The Soft AP has additional  
310 availability constraints which are documented in the OCF Security Specification.

### 311 **8.3 Retrieving and Updating Easy Setup Resources**

312 The Enrollee shall expose Easy Setup Resources such that a Mediator is able to discover them  
313 using standard OCF Resource discovery methods (i.e. via a RETRIEVE on /oic/res); see the OCF  
314 Core Specification, Section 11.3.

315 Easy Setup Resources shall expose only secure Endpoints (e.g. CoAPS); see the OCF Core  
316 Specification, Section 10.

317 The Mediator may request retrieval of an Easy Setup Resource to check the Enrollee's status at  
318 any stage of Easy Setup. This applies only when the Enrollee & the Mediator are on a common  
319 network.

320 The Mediator may request that the Enrollee update its Resource Property(-ies). Upon request from  
321 the Mediator the Enrollee shall update its current Resource Property Values, and shall perform  
322 any required action. For example, if the "cn" Property of "EasySetup" Resource is updated by the  
323 Mediator, to indicate connection to Wi-Fi, the Enrollee shall start the connection to Enroller.

324 For details of Easy Setup Resources refer to Section 6.

### 325 **8.4 Error Handling**

326 The "lec" Property of the EasySetup Resource (i.e. oic.r.easyssetup) is used to indicate the error  
327 that occurred in the Easy Setup process while trying to connect to the Enroller (using the  
328 information provided by the Mediator in WiFiConf Resource):

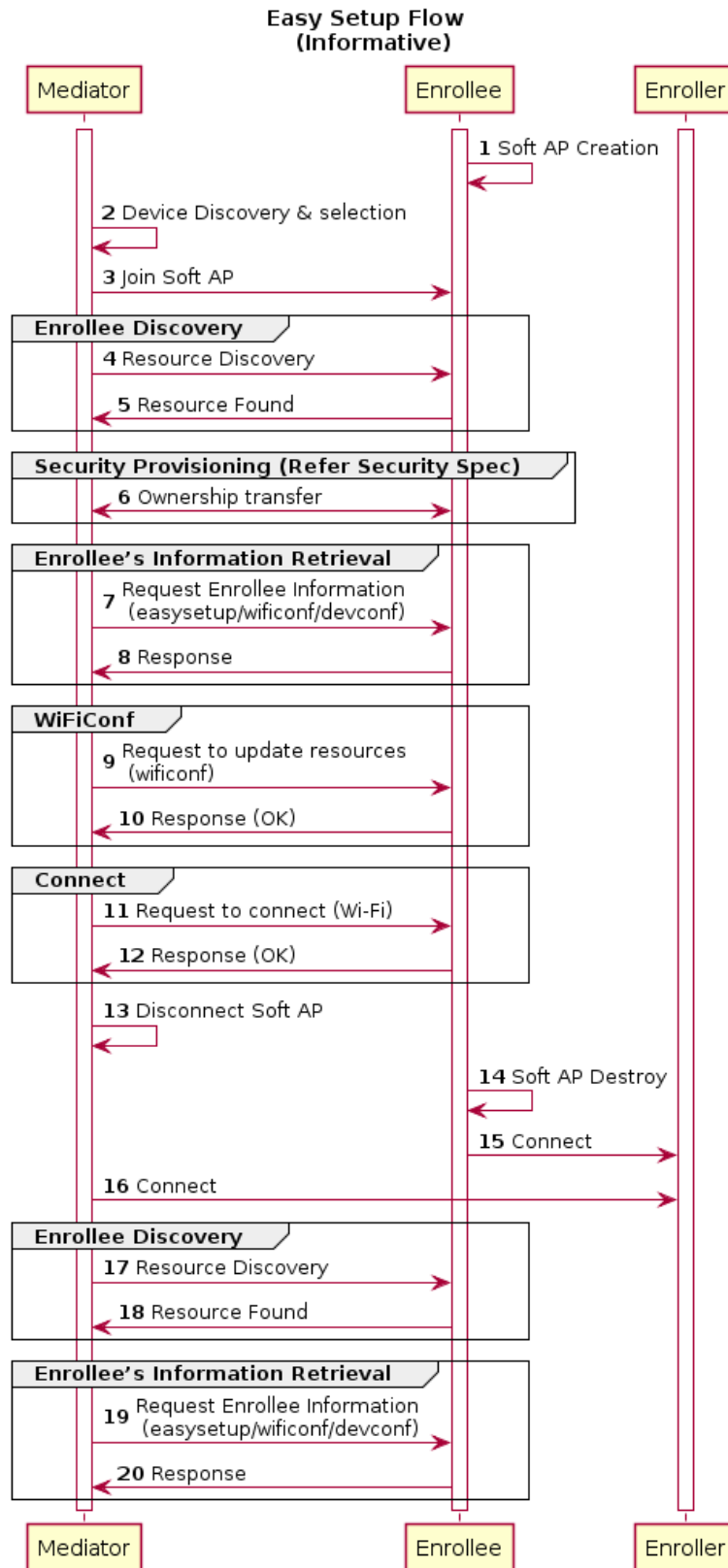
- 329 • The Enrollee shall set "lec" Property to 1, if it fails to connect because it can't find the SSID.
- 330 • The Enrollee shall set "lec" Property to 2, if it fails to connect due to wrong credential  
331 (password) information.
- 332 • The Enrollee should set "lec" Property to 6, if the Auth type is not supported by the Enrollee.
- 333 • The Enrollee should set "lec" Property to 7, if the Encryption type is not supported by the  
334 Enrollee.
- 335 • The Enrollee should set "lec" Property to 8, if it fails to connect due to wrong Auth type  
336 information (even though it's supported by the Enrollee).
- 337 • The Enrollee should set "lec" Property to 9, if it fails to connect due to wrong Encryption  
338 type information (even though it's supported by the Enrollee).



339 When using Wi-Fi as the connectivity between the Enrollee and Mediator, if the Enrollee fails to  
340 connect to the Enroller, it shall again make itself discoverable as a Soft AP (in case it destroyed  
341 its Soft AP earlier).

#### 342 **8.5 Example Easy Setup Flow**

343 The following figure shows an example Easy Setup flow for informative purposes:



346  
347  
348  
349  
350  
351

**Figure 3. Easy Setup Flow (Informative)**

The example flow above undergoes security provisioning (step 6) during Easy Setup. Alternatively security provisioning can be done before Enrollee Discovery (steps 4 and 5) if preferred. Please refer to the OCF Security Specification for more information on the different scenarios.

352 **9 Security**

353 Wi-Fi Easy Setup security requirements are captured in the OCF Security Specification.

## Annex A (normative)

### Resource Type definitions

#### A.1 List of Resource Type definitions

Table 7 contains the list of defined resources in this specification.

Table 7. Alphabetized list of resources

Friendly Name (informative)	Resource Type (rt)	Section
Easy Setup	"oic.r.easyssetup"	A.2
Wi-Fi Configuration	"oic.r.wificonf"	A.3
Device Configuration	"oic.r.devconf"	A.4

#### A.2 Easy Setup Collection Baseline Interface

##### A.2.1 Introduction

Easy Setup resource stores useful information including current status of unboxing device and last error code which are produced in a process of easy setup. Note that, Easy Setup resource is a type of collection resource, which contains links to WiFiConf, DevConf resources and may additionally contain links to other resources.

##### A.2.2 Example URI

/example/EasySetupBaselineInterfaceResURI

##### A.2.3 Resource Type

The resource type (rt) is defined as: oic.r.easyssetup.

##### A.2.4 RAML Definition

```
##RAML 0.8
title: Easy Setup Resource
version: v0.0.3-20170611

traits:
  - interface-11 :
    queryParameters:
      if:
        enum: ["oic.if.11"]
  - interface-baseline :
    queryParameters:
      if:
        enum: ["oic.if.baseline"]
  - interface-all :
    queryParameters:
      if:
        enum: ["oic.if.baseline", "oic.if.11", "oic.if.b"]
```

```

389 - interface-batch :
390   queryParameters:
391     if:
392       enum: ["oic.if.b"]
393
394 /example/EasySetupBaselineInterfaceResURI:
395   description: |
396     Easy Setup resource stores useful information including current status of
397     unboxing device and last error code which are produced in a process of
398     easy setup.
399     Note that, Easy Setup resource is a type of collection resource, which
400     contains links to WiFiConf, DevConf resources and may additionally contain
401     links to other resources.
402
403   is : ['interface-baseline']
404   get:
405     description: |
406       Retrieve useful information during easy setup process :
407       1
408       A current status in easy setup process.
409       2
410       A last error code describing reason for failure occurred at the last
411       time.
412
413     responses :
414       200:
415         body:
416           application/json:
417             schema: /
418               {
419                 "$schema": "http://json-schema.org/draft-04/schema#",
420                 "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
421 reserved.",
422                 "id": "http://www.openconnectivity.org/ocf-apis/core/schemas/oic.r.easyssetup-
423 schema.json#",
424                 "definitions": {
425                   "oic.r.easyssetup": {
426                     "type": "object",
427                     "allOf": [
428                       {
429                         "$ref": "oic.collection-schema.json#/definitions/oic.collection"
430                       },
431                       {
432                         "properties": {
433                           "rt": {
434                             "type": "array",
435                             "minItems": 2,
436                             "maxItems": 2,
437                             "uniqueItems": true,
438                             "items": {
439                               "enum": ["oic.r.easyssetup", "oic.wk.col"]
440                             }
441                           },
442                           "ps": {
443                             "type": "integer",
444                             "enum": [0, 1, 2, 3],
445                             "description": "Indicates the easy setup status of the device. (0: Need
446 to Setup, 1: Connecting to Enroller, 2: Connected to Enroller, 3: Failed to Connect to Enroller,
447 4~254: Reserved, 255: EOF)",
448                             "readOnly": true
449                           },
450                           "lec": {
451                             "type": "integer",
452                             "enum": [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 255],

```

```

453         "description": "Indicates a failure reason (0: NO error, 1: A given
454 SSID is not found, 2: Wi-Fi's password is wrong, 3: IP address is not allocated, 4: No internet
455 connection, 5: Timeout, 6: Wi-Fi Auth Type is not supported by the Enrollee, 7: Wi-Fi Encryption
456 Type is not supported by the Enrollee, 8: Wi-Fi Auth Type is wrong (failure while connecting to the
457 Enroller), 9: Wi-Fi Encryption Type is wrong (failure while connecting to the Enroller), 10-254:
458 Reserved, 255: Unknown error)",
459         "readOnly": true
460     },
461     "cn": {
462         "type": "array",
463         "description": "Indicates an array of connection types that trigger an
464 attempt to connect to the Enroller to start.",
465         "items": {
466             "type": "integer",
467             "description": "Connection type to attempt. (1 : Wi-Fi, 2 : other
468 entities / transports to be added in future (e.g. Connect to cloud / BLE))"
469         }
470     },
471     },
472     "required": ["ps", "lec", "cn"]
473 }
474 ]
475 }
476 },
477 "type": "object",
478 "allOf": [
479     { "$ref": "oic.core-schema.json#/definitions/oic.core" },
480     { "$ref": "#/definitions/oic.r.easyssetup" }
481 ]
482 }
483
484 example: /
485 {
486     "rt" : ["oic.r.easyssetup", "oic.wk.col"],
487     "if" : ["oic.if.ll", "oic.if.baseline", "oic.if.b"],
488     "ps" : 0,
489     "lec": 0,
490     "cn": [1],
491     "links": [
492         {
493             "href": "/EasySetupResURI",
494             "rt": ["oic.r.easyssetup", "oic.wk.col"],
495             "if": ["oic.if.b"],
496             "p":{"bm":3},
497             "eps": [
498                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
499             ],
500             "rel":["self", "item"]
501         },
502         {
503             "href": "/WiFiConfResURI",
504             "rt": ["oic.r.wificonf"],
505             "if": ["oic.if.baseline"],
506             "p":{"bm":3},
507             "eps": [
508                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
509             ]
510         },
511         {
512             "href": "/DevConfResURI",
513             "rt": ["oic.r.devconf"],
514             "if": ["oic.if.baseline"],
515             "p":{"bm":3},
516             "eps": [
517                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
518             ]
519         }
520     ]
521 }
522

```

## A.2.5 Property Definition

Property name	Value type	Mandatory	Access mode	Description
rt	array: see schema		Read Write	
ps	integer	yes	Read Write	Indicates the easy setup status of the device. (0: Need to Setup, 1: Connecting to Enroller, 2: Connected to Enroller, 3: Failed to Connect to Enroller, 4~254: Reserved, 255: EOF)
lec	integer	yes	Read Write	Indicates a failure reason (0: NO error, 1: A given SSID is not found, 2: Wi-Fi's password is wrong, 3: IP address is not allocated, 4: No internet connection, 5: Timeout, 6: Wi-Fi Auth Type is not supported by the Enrollee, 7: Wi-Fi Encryption Type is not supported by the Enrollee, 8: Wi-Fi Auth Type is wrong (failure while connecting to the Enroller), 9: Wi-Fi Encryption Type is wrong (failure while connecting to the Enroller), 10~254: Reserved, 255: Unknown error)
cn	array: see schema	yes	Read Write	Indicates an array of connection types that trigger an attempt to



				connect to the Enroller to start.
--	--	--	--	-----------------------------------

524 **A.2.6 CRUDN behavior**

Resource	Create	Read	Update	Delete	Notify
/example/EasySetupBaselineInterfaceResURI		get			

525 **A.3 Wi-Fi Configuration Resource Baseline Interface**

526 **A.3.1 Introduction**

527 WiFiConf resource stores essential information to help an unboxing device to connect to an  
528 existing Wi-Fi AP.

529 **A.3.2 Example URI**

530 /example/WiFiConfBaselineInterfaceResURI

531 **A.3.3 Resource Type**

532 The resource type (rt) is defined as: oic.r.wificonf.

533 **A.3.4 RAML Definition**

```
534 #%RAML 0.8
535 title: Wi-Fi Configuration Resource
536 version: v0.0.3-20170611
537 traits:
538   - interface-rw :
539     queryParameters:
540       if:
541         enum: ["oic.if.rw"]
542   - interface-baseline :
543     queryParameters:
544       if:
545         enum: ["oic.if.baseline"]
546   - interface-all :
547     queryParameters:
548       if:
549         enum: ["oic.if.baseline", "oic.if.rw"]
550
551 /example/WiFiConfBaselineInterfaceResURI:
552   description: |
553     WiFiConf resource stores essential information to help an unboxing device
554     to connect to an existing Wi-Fi AP.
555
556   is : ['interface-baseline']
557   get:
558     description: |
559       Retrieve properties of WiFiConf resource.
560       The information includes :
561       1
562       Wi-Fi SSID and password
563       2
564       Wi-Fi Security type (i.e
565       auth type and encryption type)
566       3
567       Wi-Fi hardware capability (i.e
568       supported frequencies, modes,
569       auth types and encryption types)
570
571   responses :
```

```

572     200:
573     body:
574     application/json:
575     schema: /
576     {
577     "$schema": "http://json-schema.org/draft-04/schema#",
578     "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
579 reserved.",
580     "id": "http://www.openconnectivity.org/ocf-apis/core/schemas/oic.r.wificonf-
581 schema.json#",
582     "definitions": {
583     "oic.r.wificonf": {
584     "type": "object",
585     "properties": {
586     "swmt": {
587     "type": "array",
588     "description": "Indicates supported Wi-Fi mode types. It can be multiple",
589     "readOnly": true,
590     "items":
591     {
592     "type": "string",
593     "enum": ["A", "B", "G", "N", "AC"],
594     "description": "Supported Wi-Fi Mode Type."
595     }
596     },
597     "swf": {
598     "type": "array",
599     "description": "Indicates Supported Wi-Fi frequencies by the Enrollee. Can
600 be multiple. Valid values are ('2.4G', '5G')",
601     "readOnly": true,
602     "items":
603     {
604     "type": "string",
605     "pattern": "^(2\\.4|5)G$"
606     }
607     },
608     "tnn": {
609     "type": "string",
610     "description": "Indicates Target Network Name (SSID of Wi-Fi AP)",
611     "pattern": "^.*$"
612     },
613     "cd": {
614     "type": "string",
615     "description": "Indicates credential information of Wi-Fi AP",
616     "pattern": "^.*$"
617     },
618     "wat": {
619     "type": "string",
620     "enum": ["None", "WEP", "WPA_PSK", "WPA2_PSK"],
621     "description": "Indicates Wi-Fi Auth Type"
622     },
623     "wet": {
624     "type": "string",
625     "enum": ["None", "WEP_64", "WEP_128", "TKIP", "AES", "TKIP_AES"],
626     "description": "Indicates Wi-Fi Encryption Type"
627     },
628     "swat": {
629     "type": "array",
630     "description": "Indicates supported Wi-Fi Auth types. It can be multiple",
631     "readOnly": true,
632     "items":
633     {
634     "type": "string",
635     "enum": ["None", "WEP", "WPA_PSK", "WPA2_PSK"],
636     "description": "Indicates Wi-Fi Auth Type"
637     }
638     },
639     "swet": {
640     "type": "array",

```

```

641         "description": "Indicates supported Wi-Fi Encryption types. It can be
642 multiple",
643         "readOnly": true,
644         "items":
645         {
646             "type": "string",
647             "enum": ["None", "WEP_64", "WEP_128", "TKIP", "AES", "TKIP_AES"],
648             "description": "Indicates Wi-Fi Encryption Type"
649         }
650     },
651     },
652     "required":["swmt", "swf", "swat", "swet", "tnn", "wat", "wet"]
653 },
654 },
655 "type": "object",
656 "allof": [
657     { "$ref": "oic.core-schema.json#/definitions/oic.core"},
658     { "$ref": "#/definitions/oic.r.wificonf" }
659 ]
660 }
661
662 example: /
663 {
664     "rt": ["oic.r.wificonf"],
665     "swmt" : ["A", "B", "G"],
666     "swf": ["2.4G", "5G"],
667     "tnn": "Home_AP_SSID",
668     "cd": "Home_AP_PWD",
669     "wat": "WPA2_PSK",
670     "wet": "TKIP",
671     "swat": ["WPA_PSK", "WPA2_PSK"],
672     "swet": ["TKIP", "AES", "TKIP_AES"]
673 }
674
675 post:
676 description: |
677 Deliver Wi-Fi AP's information for an unboxing device to connect to it.
678
679 body:
680 application/json:
681 schema: /
682 {
683     "$schema": "http://json-schema.org/draft-v4/schema#",
684     "description": "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
685 reserved.",
686     "id": "http://www.openconnectivity.org/ocf-apis/core/schemas/oic.r.wificonf-update-
687 schema.json#",
688     "definitions": {
689         "oic.r.wificonf": {
690             "type": "object",
691             "properties": {
692                 "tnn": {
693                     "type": "string",
694                     "description": "Indicates Target Network Name (SSID of Wi-Fi AP)",
695                     "pattern": "^.*$"
696                 },
697                 "cd": {
698                     "type": "string",
699                     "description": "Indicates credential information of Wi-Fi AP",
700                     "pattern": "^.*$"
701                 },
702                 "wat": {
703                     "enum": ["None", "WEP", "WPA_PSK", "WPA2_PSK"],
704                     "description": "Indicates Wi-Fi Auth Type"
705                 },
706                 "wet": {
707                     "enum": ["None", "WEP_64", "WEP_128", "TKIP", "AES", "TKIP_AES"],

```

```

708         "description": "Indicates Wi-Fi Encryption Type"
709     },
710 },
711     "required":["tnn", "wat", "wet"]
712 },
713 },
714     "type": "object",
715     "allOf": [
716         { "$ref": "oic.core-schema.json#/definitions/oic.core"},
717         { "$ref": "#/definitions/oic.r.wificonf" }
718     ]
719 }
720
721 example: /
722 {
723     "tnn": "Home_AP_SSID",
724     "cd": "Home_AP_PWD",
725     "wat": "WPA2_PSK",
726     "wet": "AES"
727 }
728
729 responses :
730 200:
731     body:
732     application/json:
733     schema: /
734     {
735         "$schema": "http://json-schema.org/draft-v4/schema#",
736         "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
737 reserved.",
738         "id": "http://www.openconnectivity.org/ocf-apis/core/schemas/oic.r.wificonf-update-
739 schema.json#",
740         "definitions": {
741             "oic.r.wificonf": {
742                 "type": "object",
743                 "properties": {
744                     "tnn": {
745                         "type": "string",
746                         "description": "Indicates Target Network Name (SSID of Wi-Fi AP)",
747                         "pattern": "^.*$"
748                     },
749                     "cd": {
750                         "type": "string",
751                         "description": "Indicates credential information of Wi-Fi AP",
752                         "pattern": "^.*$"
753                     },
754                     "wat": {
755                         "enum": ["None", "WEP", "WPA_PSK", "WPA2_PSK"],
756                         "description": "Indicates Wi-Fi Auth Type"
757                     },
758                     "wet": {
759                         "enum": ["None", "WEP_64", "WEP_128", "TKIP", "AES", "TKIP_AES"],
760                         "description": "Indicates Wi-Fi Encryption Type"
761                     }
762                 },
763                 "required":["tnn", "wat", "wet"]
764             }
765         },
766         "type": "object",
767         "allOf": [
768             { "$ref": "oic.core-schema.json#/definitions/oic.core"},
769             { "$ref": "#/definitions/oic.r.wificonf" }
770         ]
771     }
772
773 example: /

```

```

774     {
775         "tnn": "Home_AP_SSID",
776         "cd": "Home_AP_PWD",
777         "wat": "WPA2_PSK",
778         "wet": "AES"
779     }
780

```

### 781 A.3.5 Property Definition

Property name	Value type	Mandatory	Access mode	Description
tnn	string	yes	Read Write	Indicates Target Network Name (SSID of Wi-Fi AP)
swmt	array: schema see	yes	Read Only	Indicates supported Wi-Fi mode types. It can be multiple
swat	array: schema see	yes	Read Only	Indicates supported Wi-Fi Auth types. It can be multiple
cd	string		Read Write	Indicates credential information of Wi-Fi AP
swf	array: schema see	yes	Read Only	Indicates Supported Wi-Fi frequencies by the Enrollee. Can be multiple. Valid values are ('2.4G', '5G')
wet	string	yes	Read Write	Indicates Wi-Fi Encryption Type
wat	string	yes	Read Write	Indicates Wi-Fi Auth Type
swet	array: schema see	yes	Read Only	Indicates supported Wi-Fi Encryption types. It can be multiple

### 782 A.3.6 CRUDN behavior

Resource	Create	Read	Update	Delete	Notify
/example/WiFiConfBaselineInterfaceResURI		get	post		

## 783 A.4 Device Configuration

### 784 A.4.1 Introduction

785 Device configuration resource stores a preference of device settings like device name. Vender-  
786 specific information can be added to the resource.

### 787 A.4.2 Example URI

788 /example/DevConfResURI

### 789 A.4.3 Resource Type

790 The resource type (rt) is defined as: oic.r.devconf.

```

791 A.4.4 RAML Definition
792 #%RAML 0.8
793 title: Device Configuration Resource
794 version: v0.0.2-20170604
795 traits:
796   - interface :
797     queryParameters:
798       if:
799         enum: ["oic.if.baseline", "oic.if.r"]
800
801 /example/DevConfResURI:
802   description: |
803     Device configuration resource stores a preference of device settings like
804     device name
805     Vender-specific information can be added to the resource.
806
807   is : ['interface']
808   get:
809     description: |
810       Retrieve various settings regarding to device-specific settings
811       1
812       Device name (human-friendly name to be detected by mediator during
813       easy setup)
814
815   responses :
816     200:
817       body:
818         application/json:
819           schema: /
820             {
821               "$schema": "http://json-schema.org/draft-04/schema#",
822               "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
823 reserved.",
824               "id": "http://www.openconnectivity.org/ocf-apis/core/schemas/oic.r.devconf-
825 schema.json#",
826               "definitions": {
827                 "oic.r.devconf": {
828                   "type": "object",
829                   "oneOf": [
830                     {
831                       "properties": {
832                         "dn": {
833                           "type": "string",
834                           "description": "Indicates a pre-configured device name in language
835 indicated by 'dl' in /oic/con; presented by enrollee device to mediator device during easy-setup
836 process",
837                           "pattern": "^.*$",
838                           "readOnly": true
839                         }
840                       },
841                       "required":["dn"]
842                     },
843                     {
844                       "properties": {
845                         "dn": {
846                           "type": "array",
847                           "items": {
848                             "type": "object",
849                             "properties": {
850                               "language": {
851                                 "$ref": "oic.types-schema.json#/definitions/language-tag",
852                                 "readOnly": true,

```

```

853         "description": "An RFC 5646 language tag."
854     },
855     "value": {
856         "type": "string",
857         "description": "Pre-configured device name in the indicated
858 language.",
859         "pattern": "^.*$",
860         "readOnly": true
861     }
862 },
863 },
864 "minItems" : 1,
865 "readOnly": true,
866 "description": "Localized device name."
867 }
868 },
869 "required": ["dn"]
870 }
871 ]
872 }
873 },
874 "type": "object",
875 "allof": [
876 { "$ref": "oic.core-schema.json#/definitions/oic.core"},
877 { "$ref": "#/definitions/oic.r.devconf" }
878 ]
879 }
880
881 example: /
882 {
883     "rt": ["oic.r.devconf"],
884     "dn" : "My Refrigerator"
885 }
886

```

#### 887 A.4.5 Property Definition

Property name	Value type	Mandatory	Access mode	Description
dn	array: see schema	yes	Read Only	Localized device name.

#### 888 A.4.6 CRUDN behavior

Resource	Create	Read	Update	Delete	Notify
/example/DevConfResURI		get			

889

890

## Annex B(informative)

### Swagger2.0 definitions

891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950

#### B.1 Device Configuration

##### B.1.1 Introduction

Device configuration resource stores a preference of device settings like device name. Vender-specific information can be added to the resource. Retrieve various settings regarding to device-specific settings

1. Device name (human-friendly name to be detected by mediator during easy setup)

##### B.1.2 Wellknown URI

/example/DevConfResURI

##### B.1.3 Resource Type

The resource type (rt) is defined as: ['oic.r.devconf'].

##### B.1.4 Swagger2.0 Definition

```
{
  "swagger": "2.0",
  "info": {
    "title": "Device Configuration",
    "version": "v0.0.2-20170604",
    "license": {
      "name": "copyright 2016-2017 Open Connectivity Foundation, Inc. All rights reserved.",
      "x-description": "Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:\n
1. Redistributions of source code must retain the above copyright notice, this list of conditions and
the following disclaimer.\n
2. Redistributions in binary form must reproduce the above
copyright notice, this list of conditions and the following disclaimer in the documentation and/or
other materials provided with the distribution.\n\n
THIS SOFTWARE IS PROVIDED BY THE Open
Connectivity Foundation, INC. \AS IS\ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR
WARRANTIES OF NON-INFRINGEMENT, ARE DISCLAIMED.\n
IN NO EVENT SHALL THE Open Connectivity
Foundation, INC. OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)\n
HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.\n"
    }
  },
  "schemes": ["http"],
  "consumes": ["application/json"],
  "produces": ["application/json"],
  "paths": {
    "/example/DevConfResURI" : {
      "get": {
        "description": "Device configuration resource stores a preference of device settings
like\ndevice name. Vender-specific information can be added to the resource.\nRetrieve various
settings regarding to device-specific settings\n1. Device name (human-friendly name to be detected
by mediator during\n easy setup)\n",
        "parameters": [
          {"$ref": "#/parameters/interface"}
        ],
        "responses": {
          "200": {
            "description": "",
            "x-example": {
              "rt": ["oic.r.devconf"],
              "dn": "My Refrigerator"
            }
          }
        }
      }
    }
  }
}
```



```

951         }
952     },
953     "schema": { "$ref": "#/definitions/DevConf" }
954 }
955 }
956 }
957 }
958 },
959 "parameters": {
960     "interface": {
961         "in": "query",
962         "name": "if",
963         "type": "string",
964         "enum": ["oic.if.baseline", "oic.if.r"]
965     }
966 },
967 "definitions": {
968     "DevConf": {
969         {
970         "oneOf": [
971             {
972                 "properties": {
973                     "dn": {
974                         "description": "Indicates a pre-configured device name in language indicated by
975 'dl' in /oic/con; presented by enrollee device to mediator device during easy-setup process",
976                         "pattern": "^.*$",
977                         "readOnly": true,
978                         "type": "string"
979                     }
980                 },
981                 "required": [
982                     "dn"
983                 ]
984             },
985             {
986                 "properties": {
987                     "dn": {
988                         "description": "Localized device name.",
989                         "items": {
990                             "properties": {
991                                 "language": {
992                                     "description": "An RFC 5646 language tag.",
993                                     "pattern": "^[A-Za-z]{1,8}(-[A-Za-z0-9]{1,8})*$",
994                                     "readOnly": true,
995                                     "type": "string"
996                                 },
997                                 "value": {
998                                     "description": "Pre-configured device name in the indicated language.",
999                                     "pattern": "^.*$",
1000                                     "readOnly": true,
1001                                     "type": "string"
1002                                 }
999                             }
1000                         },
1001                         "type": "object"
1002                     },
1003                     "minItems": 1,
1004                     "readOnly": true,
1005                     "type": "array"
1006                 }
1007             },
1008             "required": [
1009                 "dn"
1010             ]
1011         }
1012     ],
1013     "type": "object"
1014 }
1015 }
1016 }
1017 }
1018 }
1019 }
1020 }
1021 }

```

1022 **B.1.5 Property Definition**

Property name	Value type	Mandatory	Access mode	Description
dn	array: see schema	yes	Read Only	Localized device name.

1023 **B.1.6 CRUDN behaviour**

Resource	Create	Read	Update	Delete	Notify
/example/DevConfResURI		get			

1024 **B.2 Easy Setup Collection**

1025 **B.2.1 Introduction**

1026 Easy Setup resource stores useful information including current status of  
 1027 unboxing device and last error code which are produced in a process of  
 1028 easy setup.  
 1029 Note that, Easy Setup resource is a type of collection resource, which  
 1030 contains links to WiFiConf, DevConf resources and may additionally contain  
 1031 links to other resources.  
 1032 Retrieve useful information during easy setup process:  
 1033 1. A current status in easy setup process.  
 1034 2. A last error code describing reason for failure occurred at the last  
 1035 time.  
 1036

1037 **B.2.2 Wellknown URI**

1038 /example/EasySetupBaselineInterfaceResURI

1039 **B.2.3 Resource Type**

1040 The resource type (rt) is defined as: ['oic.r.easyssetup', 'oic.wk.col'].

1041 **B.2.4 Swagger2.0 Definition**

```

1042 {
1043   "swagger": "2.0",
1044   "info": {
1045     "title": "Easy Setup Collection Batch Interface",
1046     "version": "v0.0.3-20170611",
1047     "license": {
1048       "name": "copyright 2016-2017 Open Connectivity Foundation, Inc. All rights reserved.",
1049       "x-description": "Redistribution and use in source and binary forms, with or without
1050 modification, are permitted provided that the following conditions are met:\n      1.
1051 Redistributions of source code must retain the above copyright notice, this list of conditions and
1052 the following disclaimer.\n      2. Redistributions in binary form must reproduce the above
1053 copyright notice, this list of conditions and the following disclaimer in the documentation and/or
1054 other materials provided with the distribution.\n\n      THIS SOFTWARE IS PROVIDED BY THE Open
1055 Connectivity Foundation, INC. \AS IS\ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
1056 LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR
1057 WARRANTIES OF NON-INFRINGEMENT, ARE DISCLAIMED.\n      IN NO EVENT SHALL THE Open Connectivity
1058 Foundation, INC. OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
1059 EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
1060 OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)\n      HOWEVER CAUSED AND
1061 ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
1062 OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
1063 OF SUCH DAMAGE.\n"
1064     }
1065   },
1066   "schemes": ["http"],
1067   "consumes": ["application/json"],
1068   "produces": ["application/json"],
1069   "paths": {
1070     "/example/EasySetupBatchInterfaceResURI" : {
1071       "get": {
1072         "description": "Easy Setup resource stores useful information including current status
  
```

```

1073 of\nunboxing device and last error code which are produced in a process of\neasy setup.\nNote that,
1074 Easy Setup resource is a type of collection resource, which\ncontains links to WiFiConf, DevConf
1075 resources and may additionally contain\nlinks to other resources.\nRetrieve useful information
1076 during easy setup process :\nl. A current status in easy setup process.\n2. A last error code
1077 describing reason for failure occurred at the last\n time.\n",
1078 "parameters": [
1079   {"$ref": "#/parameters/interface-batch"}
1080 ],
1081 "responses": {
1082   "200": {
1083     "description": "",
1084     "x-example":
1085     [
1086       {
1087         "href": "/EasySetupResURI",
1088         "rep":{
1089           "ps": 0,
1090           "lec": 0,
1091           "cn": [1]
1092         }
1093       },
1094       {
1095         "href": "/WiFiConfResURI",
1096         "rep":{
1097           "swmt": ["A", "B", "G"],
1098           "swf": ["2.4G", "5G"],
1099           "tnn": "Home_AP_SSID",
1100           "cd": "Home_AP_PWD",
1101           "wat": "WPA2_PSK",
1102           "wet": "AES"
1103         }
1104       },
1105       {
1106         "href": "/DevConfResURI",
1107         "rep":{
1108           "dn": "My Refrigerator"
1109         }
1110       }
1111     ]
1112   },
1113   "schema": { "$ref": "#/definitions/sbatch" }
1114 }
1115 },
1116 "post": {
1117   "description": "Able to deliver Wi-Fi, Device configuration and other
1118 configuration\ninformation in a batch by utilizing 'batch' interface.\nIf you want to deliver Wi-Fi
1119 and Device configuration information in a batch,\nyou can write all properties you want to send
1120 with a 'batch' interface.\nThe below example is the case to send Easy Setup and Wi-Fi
1121 configuration\n(i.e. connection type, target network, auth type information) in a batch.\n",
1122   "parameters": [
1123     {"$ref": "#/parameters/interface-batch"},
1124     {
1125       "name": "body",
1126       "in": "body",
1127       "required": true,
1128       "schema": { "$ref": "#/definitions/sbatch-update" },
1129       "x-example":
1130       [
1131         {
1132           "href": "/EasySetupResURI",
1133           "rep":{
1134             "cn": [1]
1135           }
1136         },
1137         {
1138           "href": "/WiFiConfResURI",
1139           "rep":{
1140             "tnn": "Home_AP_SSID",
1141             "cd": "Home_AP_PWD",
1142             "wat": "WPA2_PSK",

```

```

1144         "wet": "AES"
1145     }
1146 }
1147 ]
1148 }
1149 ],
1150 "responses": {
1151     "200": {
1152         "description": "",
1153         "x-example":
1154             [
1155                 {
1156                     "href": "/EasySetupResURI",
1157                     "rep": {
1158                         "ps": 0,
1159                         "lec": 0,
1160                         "cn": [1]
1161                     }
1162                 },
1163                 {
1164                     "href": "/WiFiConfResURI",
1165                     "rep": {
1166                         "swmt": ["A", "B", "G"],
1167                         "swf": ["2.4G", "5G"],
1168                         "tnn": "Home_AP_SSID",
1169                         "cd": "Home_AP_PWD",
1170                         "wat": "WPA2_PSK",
1171                         "wet": "AES"
1172                     }
1173                 },
1174                 {
1175                     "href": "/DevConfResURI",
1176                     "rep": {
1177                         "dn": "My Refrigerator"
1178                     }
1179                 }
1180             ]
1181         },
1182         "schema": { "$ref": "#/definitions/sbatch" }
1183     }
1184 }
1185 },
1186 "/example/EasySetupLLInterfaceResURI" : {
1187     "get": {
1188         "description": "Easy Setup resource stores useful information including current status
1189 of\nunboxing device and last error code which are produced in a process of\neasy setup.\nNote that,
1190 Easy Setup resource is a type of collection resource, which\ncontains links to WiFiConf, DevConf
1191 resources and may additionally contain\nlinks to other resources.\nRetrieve useful information
1192 during easy setup process :\n1. A current status in easy setup process.\n2. A last error code
1193 describing reason for failure occurred at the last\n time.\n",
1194         "parameters": [
1195             { "$ref": "#/parameters/interface-ll" }
1196         ],
1197         "responses": {
1198             "200": {
1199                 "description": "",
1200                 "x-example":
1201                     [
1202                         {
1203                             "href": "/EasySetupResURI",
1204                             "rt": ["oic.r.easyssetup", "oic.wk.col"],
1205                             "if": ["oic.if.b"],
1206                             "p":{"bm":3},
1207                             "eps": [
1208                                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
1209                             ],
1210                             "rel":["self", "item"]
1211                         },
1212                         {
1213                             "href": "/WiFiConfResURI",
1214

```

```

1215         "rt": ["oic.r.wificonf"],
1216         "if": ["oic.if.baseline"],
1217         "p":{"bm":3},
1218         "eps": [
1219             {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
1220         ]
1221     },
1222     {
1223         "href": "/DevConfResURI",
1224         "rt": ["oic.r.devconf"],
1225         "if": ["oic.if.baseline"],
1226         "p":{"bm":3},
1227         "eps": [
1228             {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
1229         ]
1230     }
1231 ]
1232 ,
1233 "schema": { "$ref": "#/definitions/slinks" }
1234 }
1235 }
1236 }
1237 },
1238 "/example/EasySetupBaselineInterfaceResURI" : {
1239     "get": {
1240         "description": "Easy Setup resource stores useful information including current status
1241 of\nunboxing device and last error code which are produced in a process of\neasy setup.\nNote that,
1242 Easy Setup resource is a type of collection resource, which\ncontains links to WiFiConf, DevConf
1243 resources and may additionally contain\nlinks to other resources.\nRetrieve useful information
1244 during easy setup process :\n 1. A current status in easy setup process.\n 2. A last error code
1245 describing reason for failure occurred at the last\n      time.\n",
1246         "parameters": [
1247             {"$ref": "#/parameters/interface-baseline"}
1248         ],
1249         "responses": {
1250             "200": {
1251                 "description": "",
1252                 "x-example":
1253                 {
1254                     "rt" : ["oic.r.easyssetup", "oic.wk.col"],
1255                     "if" : ["oic.if.ll", "oic.if.baseline", "oic.if.b"],
1256                     "ps" : 0,
1257                     "lec" : 0,
1258                     "cn": [1],
1259                     "links": [
1260                         {
1261                             "href": "/EasySetupResURI",
1262                             "rt": ["oic.r.easyssetup", "oic.wk.col"],
1263                             "if": ["oic.if.b"],
1264                             "p":{"bm":3},
1265                             "eps": [
1266                                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
1267                             ],
1268                             "rel":["self", "item"]
1269                         },
1270                         {
1271                             "href": "/WiFiConfResURI",
1272                             "rt": ["oic.r.wificonf"],
1273                             "if": ["oic.if.baseline"],
1274                             "p":{"bm":3},
1275                             "eps": [
1276                                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}
1277                             ]
1278                         },
1279                         {
1280                             "href": "/DevConfResURI",
1281                             "rt": ["oic.r.devconf"],
1282                             "if": ["oic.if.baseline"],
1283                             "p":{"bm":3},
1284                             "eps": [
1285                                 {"ep": "coaps://[fe80::b1d6]:1111", "pri": 2}

```

```

1286         ]
1287     }
1288 ]
1289 }
1290 ,
1291 "schema": { "$ref": "#/definitions/EasySetup" }
1292 }
1293 }
1294 }
1295 }
1296 },
1297 "parameters": {
1298     "interface-ll" : {
1299         "in" : "query",
1300         "name" : "if",
1301         "type" : "string",
1302         "enum" : ["oic.if.ll"]
1303     },
1304     "interface-baseline" : {
1305         "in" : "query",
1306         "name" : "if",
1307         "type" : "string",
1308         "enum" : ["oic.if.baseline"]
1309     },
1310     "interface-all" : {
1311         "in" : "query",
1312         "name" : "if",
1313         "type" : "string",
1314         "enum" : ["oic.if.baseline", "oic.if.ll", "oic.if.b"]
1315     },
1316     "interface-batch" : {
1317         "in" : "query",
1318         "name" : "if",
1319         "type" : "string",
1320         "enum" : ["oic.if.b"]
1321     }
1322 },
1323 "definitions": {
1324     "sbatch" :
1325     {
1326         "items": {
1327             "additionalProperties": true,
1328             "properties": {
1329                 "href": {
1330 anchor,
1331                 "description": "URI of the target resource relative assuming the collection URI as
1332                 "format": "uri",
1333                 "maxLength": 256,
1334                 "type": "string"
1335             },
1336             "rep": {
1337                 "oneOf": [
1338                     {
1339                         "description": "The response payload from a single resource",
1340                         "type": "object"
1341                     },
1342                     {
1343                         "description": " The response payload from a collection (batch) resource",
1344                         "type": "array"
1345                     }
1346                 ]
1347             }
1348         },
1349         "required": [
1350             "href",
1351             "rep"
1352         ],
1353         "type": "object"
1354     },
1355     "minItems": 1,
1356     "type": "array"

```

```

1357     }
1358
1359     ,
1360     "sbatch-update" :
1361     {
1362         "description": "array of resource representations to apply to the batch collection, using
1363 href to indicate which resource(s) in the batch to update. If the href property is empty,
1364 effectively making the URI reference to the collection itself, the representation is to be applied
1365 to all resources in the batch",
1366         "items": {
1367             "additionalProperties": true,
1368             "properties": {
1369                 "href": {
1370                     "description": "URI of the target resource relative assuming the collection URI as
1371 anchor",
1372                     "format": "uri",
1373                     "maxLength": 256,
1374                     "type": "string"
1375                 },
1376                 "rep": {
1377                     "oneOf": [
1378                         {
1379                             "description": "The response payload from a single resource",
1380                             "type": "object"
1381                         },
1382                         {
1383                             "description": " The response payload from a collection (batch) resource",
1384                             "type": "array"
1385                         }
1386                     ]
1387                 }
1388             },
1389             "required": [
1390                 "href",
1391                 "rep"
1392             ],
1393             "type": "object"
1394         },
1395         "minItems": 1,
1396         "type": "array"
1397     }
1398
1399     ,
1400     "slinks" :
1401     {
1402         "description": "All forms of links in a collection",
1403         "oneOf": [
1404             {
1405                 "description": "A set (array) of simple or individual OIC Links. In addition to
1406 properties required for an OIC Link, the identifier for that link in this set is also required",
1407                 "items": {
1408                     "properties": {
1409                         "anchor": {
1410                             "description": "This is used to override the context URI e.g. override the URI of
1411 the containing collection",
1412                             "format": "uri",
1413                             "maxLength": 256,
1414                             "type": "string"
1415                         },
1416                         "di": {
1417                             "description": "Unique identifier for device (UUID)",
1418                             "pattern": "[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
1419 F0-9]{12}$",
1420                             "type": "string"
1421                         },
1422                         "eps": {
1423                             "description": "the Endpoint information of the target Resource",
1424                             "items": {
1425                                 "properties": {
1426                                     "ep": {
1427                                         "description": "URI with Transport Protocol Suites + Endpoint Locator as

```

```

1428 specified in 10.2.1",
1429         "format": "uri",
1430         "type": "string"
1431     },
1432     "pri": {
1433         "description": "The priority among multiple Endpoints as specified in
1434 10.2.3",
1435         "minimum": 1,
1436         "type": "integer"
1437     }
1438 },
1439 "type": "object"
1440 },
1441 "type": "array"
1442 },
1443 "href": {
1444     "description": "This is the target URI, it can be specified as a Relative
1445 Reference or fully-qualified URI. Relative Reference should be used along with the di parameter to
1446 make it unique.",
1447     "format": "uri",
1448     "maxLength": 256,
1449     "type": "string"
1450 },
1451 "if": {
1452     "description": "The interface set supported by this resource",
1453     "items": {
1454         "enum": [
1455             "oic.if.baseline",
1456             "oic.if.ll",
1457             "oic.if.b",
1458             "oic.if.rw",
1459             "oic.if.r",
1460             "oic.if.a",
1461             "oic.if.s"
1462         ],
1463         "type": "string"
1464     },
1465     "minItems": 1,
1466     "type": "array"
1467 },
1468 "ins": {
1469     "description": "The instance identifier for this web link in an array of web
1470 links - used in collections",
1471     "oneOf": [
1472         {
1473             "description": "An ordinal number that is not repeated - must be unique in
1474 the collection context",
1475             "type": "integer"
1476         },
1477         {
1478             "description": "Any unique string including a URI",
1479             "format": "uri",
1480             "maxLength": 256,
1481             "type": "string"
1482         },
1483         {
1484             "description": "Unique identifier (UUID)",
1485             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-
1486 fA-F0-9]{12}$",
1487             "type": "string"
1488         }
1489     ]
1490 },
1491 "p": {
1492     "description": "Specifies the framework policies on the Resource referenced by
1493 the target URI",
1494     "properties": {
1495         "bm": {
1496             "description": "Specifies the framework policies on the Resource referenced
1497 by the target URI for e.g. observable and discoverable",
1498             "type": "integer"

```



```

1499         }
1500     },
1501     "required": [
1502         "bm"
1503     ],
1504     "type": "object"
1505 },
1506 "rel": {
1507     "description": "The relation of the target URI referenced by the link to the
context URI",
1508     "oneOf": [
1509         {
1510             "default": [
1511                 "hosts"
1512             ],
1513             "items": {
1514                 "maxLength": 64,
1515                 "type": "string"
1516             },
1517             "minItems": 1,
1518             "type": "array"
1519         },
1520         {
1521             "default": "hosts",
1522             "maxLength": 64,
1523             "type": "string"
1524         }
1525     ]
1526 },
1527 },
1528 "rt": {
1529     "description": "Resource Type",
1530     "items": {
1531         "maxLength": 64,
1532         "type": "string"
1533     },
1534     "minItems": 1,
1535     "type": "array"
1536 },
1537 "title": {
1538     "description": "A title for the link relation. Can be used by the UI to provide a
context",
1539     "maxLength": 64,
1540     "type": "string"
1541 },
1542 },
1543 "type": {
1544     "default": "application/cbor",
1545     "description": "A hint at the representation of the resource referenced by the
target URI. This represents the media types that are used for both accepting and emitting",
1546     "items": {
1547         "maxLength": 64,
1548         "type": "string"
1549     },
1550     "minItems": 1,
1551     "type": "array"
1552 }
1553 },
1554 },
1555 "required": [
1556     "href",
1557     "rt",
1558     "if"
1559 ],
1560 "type": "object"
1561 },
1562 "type": "array"
1563 }
1564 ]
1565 }
1566
1567 'EasySetup' :
1568 {
1569

```

```

1570         "description": "A collection is a set (array) of tagged-link or set (array) of simple links
1571 along with additional properties to describe the collection itself",
1572         "properties": {
1573             "cn": {
1574                 "description": "Indicates an array of connection types that trigger an attempt to
1575 connect to the Enroller to start.",
1576                 "items": {
1577                     "description": "Connection type to attempt. (1 : Wi-Fi, 2 : other entities /
1578 transports to be added in future (e.g. Connect to cloud / BLE))",
1579                     "type": "integer"
1580                 },
1581                 "type": "array"
1582             },
1583             "di": {
1584                 "description": "The device ID which is an UUIDv4 string; used for backward
1585 compatibility with Spec A definition of /oic/res",
1586                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
1587 9]{12}$",
1588                 "type": "string"
1589             },
1590             "drel": {
1591                 "description": "When specified this is the default relationship to use when an OIC Link
1592 does not specify an explicit relationship with *rel* parameter",
1593                 "type": "string"
1594             },
1595             "id": {
1596                 "anyOf": [
1597                     {
1598                         "description": "A number that is unique to that collection; like an ordinal number
1599 that is not repeated",
1600                         "type": "integer"
1601                     },
1602                     {
1603                         "description": "A unique string that could be a hash or similarly unique",
1604                         "type": "string"
1605                     },
1606                     {
1607                         "description": "A unique string that could be a UUIDv4",
1608                         "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
1609 9]{12}$",
1610                         "type": "string"
1611                     }
1612                 ],
1613                 "description": "ID for the collection. Can be an value that is unique to the use
1614 context or a UUIDv4"
1615             },
1616             "lec": {
1617                 "description": "Indicates a failure reason (0: NO error, 1: A given SSID is not found,
1618 2: Wi-Fi's password is wrong, 3: IP address is not allocated, 4: No internet connection, 5:
1619 Timeout, 6: Wi-Fi Auth Type is not supported by the Enrollee, 7: Wi-Fi Encryption Type is not
1620 supported by the Enrollee, 8: Wi-Fi Auth Type is wrong (failure while connecting to the Enroller),
1621 9: Wi-Fi Encryption Type is wrong (failure while connecting to the Enroller), 10~254: Reserved,
1622 255: Unknown error)",
1623                 "enum": [
1624                     0,
1625                     1,
1626                     2,
1627                     3,
1628                     4,
1629                     5,
1630                     6,
1631                     7,
1632                     8,
1633                     9,
1634                     255
1635                 ],
1636                 "readOnly": true,
1637                 "type": "integer"
1638             },
1639             "links": {
1640                 "description": "All forms of links in a collection",

```

```

1641         "oneOf": [
1642             {
1643                 "description": "A set (array) of simple or individual OIC Links. In addition to
1644 properties required for an OIC Link, the identifier for that link in this set is also required",
1645                 "items": {
1646                     "properties": {
1647                         "anchor": {
1648                             "description": "This is used to override the context URI e.g. override the
1649 URI of the containing collection",
1650                             "format": "uri",
1651                             "maxLength": 256,
1652                             "type": "string"
1653                         },
1654                         "di": {
1655                             "description": "Unique identifier for device (UUID)",
1656                             "pattern": "[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-
1657 fA-F0-9]{12}$",
1658                             "type": "string"
1659                         },
1660                         "eps": {
1661                             "description": "the Endpoint information of the target Resource",
1662                             "items": {
1663                                 "properties": {
1664                                     "ep": {
1665                                         "description": "URI with Transport Protocol Suites + Endpoint Locator
1666 as specified in 10.2.1",
1667                                         "format": "uri",
1668                                         "type": "string"
1669                                     },
1670                                     "pri": {
1671                                         "description": "The priority among multiple Endpoints as specified in
1672 10.2.3",
1673                                         "minimum": 1,
1674                                         "type": "integer"
1675                                     }
1676                                 },
1677                                 "type": "object"
1678                             },
1679                             "type": "array"
1680                         },
1681                         "href": {
1682                             "description": "This is the target URI, it can be specified as a Relative
1683 Reference or fully-qualified URI. Relative Reference should be used along with the di parameter to
1684 make it unique.",
1685                             "format": "uri",
1686                             "maxLength": 256,
1687                             "type": "string"
1688                         },
1689                         "if": {
1690                             "description": "The interface set supported by this resource",
1691                             "items": {
1692                                 "enum": [
1693                                     "oic.if.baseline",
1694                                     "oic.if.ll",
1695                                     "oic.if.b",
1696                                     "oic.if.rw",
1697                                     "oic.if.r",
1698                                     "oic.if.a",
1699                                     "oic.if.s"
1700                                 ],
1701                                 "type": "string"
1702                             },
1703                             "minItems": 1,
1704                             "type": "array"
1705                         },
1706                         "ins": {
1707                             "description": "The instance identifier for this web link in an array of web
1708 links - used in collections",
1709                             "oneOf": [
1710                                 {
1711                                     "description": "An ordinal number that is not repeated - must be unique

```

```

1712 in the collection context",
1713     "type": "integer"
1714   },
1715   {
1716     "description": "Any unique string including a URI",
1717     "format": "uri",
1718     "maxLength": 256,
1719     "type": "string"
1720   },
1721   {
1722     "description": "Unique identifier (UUID)",
1723     "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-
1724 [a-fA-F0-9]{12}$",
1725     "type": "string"
1726   }
1727 ]
1728 },
1729 "p": {
1730   "description": "Specifies the framework policies on the Resource referenced
1731 by the target URI",
1732   "properties": {
1733     "bm": {
1734       "description": "Specifies the framework policies on the Resource
1735 referenced by the target URI for e.g. observable and discoverable",
1736       "type": "integer"
1737     }
1738   },
1739   "required": [
1740     "bm"
1741   ],
1742   "type": "object"
1743 },
1744 "rel": {
1745   "description": "The relation of the target URI referenced by the link to the
1746 context URI",
1747   "oneOf": [
1748     {
1749       "default": [
1750         "hosts"
1751       ],
1752       "items": {
1753         "maxLength": 64,
1754         "type": "string"
1755       },
1756       "minItems": 1,
1757       "type": "array"
1758     },
1759     {
1760       "default": "hosts",
1761       "maxLength": 64,
1762       "type": "string"
1763     }
1764   ]
1765 },
1766 "rt": {
1767   "description": "Resource Type",
1768   "items": {
1769     "maxLength": 64,
1770     "type": "string"
1771   },
1772   "minItems": 1,
1773   "type": "array"
1774 },
1775 "title": {
1776   "description": "A title for the link relation. Can be used by the UI to
1777 provide a context",
1778   "maxLength": 64,
1779   "type": "string"
1780 },
1781 "type": {
1782   "default": "application/cbor",

```

```

1783         "description": "A hint at the representation of the resource referenced by
1784 the target URI. This represents the media types that are used for both accepting and emitting",
1785         "items": {
1786             "maxLength": 64,
1787             "type": "string"
1788         },
1789         "minItems": 1,
1790         "type": "array"
1791     }
1792 },
1793     "required": [
1794         "href",
1795         "rt",
1796         "if"
1797     ],
1798     "type": "object"
1799 },
1800 "type": "array"
1801 }
1802 ]
1803 },
1804 "ps": {
1805     "description": "Indicates the easy setup status of the device. (0: Need to Setup, 1:
1806 Connecting to Enroller, 2: Connected to Enroller, 3: Failed to Connect to Enroller, 4-254:
1807 Reserved, 255: EOF)",
1808     "enum": [
1809         0,
1810         1,
1811         2,
1812         3
1813     ],
1814     "readOnly": true,
1815     "type": "integer"
1816 },
1817 "rt": {
1818     "items": {
1819         "enum": [
1820             "oic.r.easyssetup",
1821             "oic.wk.col"
1822         ]
1823     },
1824     "maxItems": 2,
1825     "minItems": 2,
1826     "type": "array",
1827     "uniqueItems": true
1828 },
1829 "rts": {
1830     "description": "Defines the list of allowable resource types (for Target and anchors)
1831 in links included in the collection; new links being created can only be from this list",
1832     "items": {
1833         "maxLength": 64,
1834         "type": "string"
1835     },
1836     "minItems": 1,
1837     "readOnly": true,
1838     "type": "array"
1839 }
1840 },
1841 "required": [
1842     "ps",
1843     "lec",
1844     "cn"
1845 ],
1846 "type": "object"
1847 }
1848 }
1849 }
1850 }
1851

```

**B.2.5 Property Definition**

Property name	Value type	Mandatory	Access mode	Description
rep	multiple types: see schema	yes		
href	string	yes		URI of the target resource relative assuming the collection URI as anchor
cn	array: see schema	yes		Indicates an array of connection types that trigger an attempt to connect to the Enroller to start.
di	string			The device ID which is an UUIDv4 string; used for backward compatibility with Spec A definition of /oic/res
lec	integer	yes	Read Only	Indicates a failure reason (0: NO error, 1: A given SSID is not found, 2: Wi-Fi's password is wrong, 3: IP address is not allocated, 4: No internet connection, 5: Timeout, 6: Wi-Fi Auth Type is not supported by the Enrollee, 7: Wi-Fi Encryption Type is not supported by the Enrollee, 8: Wi-Fi Auth Type is wrong (failure while connecting to the Enroller), 9: Wi-Fi Encryption Type is wrong (failure while connecting to the Enroller), 10~254: Reserved, 255: Unknown error)

drel	string			When specified this is the default relationship to use when an OIC Link does not specify an explicit relationship with *rel* parameter
id	multiple types: see schema			ID for the collection. Can be an value that is unique to the use context or a UUIDv4
rt	array: see schema			
rts	array: see schema		Read Only	Defines the list of allowable resource types (for Target and anchors) in links included in the collection; new links being created can only be from this list
ps	integer	yes	Read Only	Indicates the easy setup status of the device. (0: Need to Setup, 1: Connecting to Enroller, 2: Connected to Enroller, 3: Failed to Connect to Enroller, 4~254: Reserved, 255: EOF)
links	multiple types: see schema			All forms of links in a collection
anchor	string			This is used to override the context URI e.g. override the URI of the containing collection
di	string			Unique identifier for device (UUID)
title	string			A title for the link relation. Can be used by the UI to provide a context

p	object: see schema			Specifies the framework policies on the Resource referenced by the target URI
rel	multiple types: see schema			The relation of the target URI referenced by the link to the context URI
rt	array: see schema	yes		Resource Type
ins	multiple types: see schema			The instance identifier for this web link in an array of web links - used in collections
eps	array: see schema			the Endpoint information of the target Resource
if	array: see schema	yes		The interface set supported by this resource
type	array: see schema			A hint at the representation of the resource referenced by the target URI. This represents the media types that are used for both accepting and emitting
href	string	yes		This is the target URI, it can be specified as a Relative Reference or fully-qualified URI. Relative Reference should be used along with the di parameter to make it unique.
rep	multiple types: see schema	yes		
href	string	yes		URI of the target resource relative assuming the collection URI as anchor



1853 **B.2.6 CRUDN behaviour**

Resource	Create	Read	Update	Delete	Notify
/example/EasySetupBaselineInterfaceResURI		get			

1854 **B.3 Wi-Fi Configuration Resource**

1855 **B.3.1 Introduction**

1856 WiFiConf resource stores essential information to help an unboxing device  
 1857 to connect to an existing Wi-Fi AP.  
 1858 Retrieve properties of WiFiConf resource.  
 1859 The information includes:  
 1860 1. Wi-Fi SSID and password  
 1861 2. Wi-Fi Security type (i.e. auth type and encryption type)  
 1862 3. Wi-Fi hardware capability (i.e. supported frequencies, modes,  
 1863 auth types and encryption types)  
 1864

1865 **B.3.2 Wellknown URI**

1866 /example/WiFiConfBaselineInterfaceResURI

1867 **B.3.3 Resource Type**

1868 The resource type (rt) is defined as: ['oic.r.wificonf'].

1869 **B.3.4 Swagger2.0 Definition**

```

1870 {
1871   "swagger": "2.0",
1872   "info": {
1873     "title": "Wi-Fi Configuration Resource Baseline Interface",
1874     "version": "v0.0.3-20170611",
1875     "license": {
1876       "name": "copyright 2016-2017 Open Connectivity Foundation, Inc. All rights reserved.",
1877       "x-description": "Redistribution and use in source and binary forms, with or without
1878 modification, are permitted provided that the following conditions are met:\n      1.
1879 Redistributions of source code must retain the above copyright notice, this list of conditions and
1880 the following disclaimer.\n      2. Redistributions in binary form must reproduce the above
1881 copyright notice, this list of conditions and the following disclaimer in the documentation and/or
1882 other materials provided with the distribution.\n\n      THIS SOFTWARE IS PROVIDED BY THE Open
1883 Connectivity Foundation, INC. \\"AS IS\\" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
1884 LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE OR
1885 WARRANTIES OF NON-INFRINGEMENT, ARE DISCLAIMED.\n\n      IN NO EVENT SHALL THE Open Connectivity
1886 Foundation, INC. OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
1887 EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
1888 OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)\n\n      HOWEVER CAUSED AND
1889 ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
1890 OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
1891 OF SUCH DAMAGE.\n"
1892   }
1893 },
1894 "schemes": ["http"],
1895 "consumes": ["application/json"],
1896 "produces": ["application/json"],
1897 "paths": {
1898   "/example/WiFiConfBaselineInterfaceResURI" : {
1899     "get": {
1900       "description": "WiFiConf resource stores essential information to help an unboxing
1901 device\nto connect to an existing Wi-Fi AP.\nRetrieve properties of WiFiConf resource.\nThe
1902 information includes :\n1. Wi-Fi SSID and password\n2. Wi-Fi Security type (i.e. auth type and
1903 encryption type)\n3. Wi-Fi hardware capability (i.e. supported frequencies, modes,\n auth types
1904 and encryption types)\n",
1905       "parameters": [
1906         {"$ref": "#/parameters/interface-baseline"}
1907       ],
1908       "responses": {
1909         "200": {

```

```

1910         "description" : "",
1911         "x-example":
1912         {
1913             "rt": ["oic.r.wificonf"],
1914             "swmt" : ["A", "B", "G"],
1915             "swf": ["2.4G", "5G"],
1916             "tnn": "Home_AP_SSID",
1917             "cd": "Home_AP_PWD",
1918             "wat": "WPA2_PSK",
1919             "wet": "TKIP",
1920             "swat": ["WPA_PSK", "WPA2_PSK"],
1921             "swet": ["TKIP", "AES", "TKIP_AES"]
1922         }
1923     ,
1924     "schema": { "$ref": "#/definitions/WiFiConf" }
1925 }
1926 },
1927 ],
1928 "post": {
1929     "description": "Deliver Wi-Fi AP's information for an unboxing device to connect to it.\n",
1930     "parameters": [
1931         { "$ref": "#/parameters/interface-baseline" },
1932         {
1933             "name": "body",
1934             "in": "body",
1935             "required": true,
1936             "schema": { "$ref": "#/definitions/WiFiConfUpdate" },
1937             "x-example":
1938             {
1939                 "tnn": "Home_AP_SSID",
1940                 "cd": "Home_AP_PWD",
1941                 "wat": "WPA2_PSK",
1942                 "wet": "AES"
1943             }
1944         }
1945     ],
1946     "responses": {
1947         "200": {
1948             "description": "",
1949             "x-example":
1950             {
1951                 "tnn": "Home_AP_SSID",
1952                 "cd": "Home_AP_PWD",
1953                 "wat": "WPA2_PSK",
1954                 "wet": "AES"
1955             }
1956         }
1957     },
1958     "schema": { "$ref": "#/definitions/WiFiConfUpdate" }
1959 }
1960 },
1961 ],
1962 "/example/WiFiConfRWInterfaceResURI" : {
1963     "get": {
1964         "description": "WiFiConf resource stores essential information to help an unboxing
1965 device\nto connect to an existing Wi-Fi AP.\nRetrieve properties of WiFiConf resource that can be
1966 updated by a client.\n",
1967         "parameters": [
1968             { "$ref": "#/parameters/interface-rw" }
1969         ],
1970         "responses": {
1971             "200": {
1972                 "description": "",
1973                 "x-example":
1974                 {
1975                     "tnn": "Home_AP_SSID",
1976                     "cd": "Home_AP_PWD",
1977                     "wat": "WPA2_PSK",
1978                     "wet": "AES"
1979                 }
1980             }

```

```

1981         "schema": { "$ref": "#/definitions/WiFiConfUpdate" }
1982     }
1983 }
1984 },
1985 "post": {
1986     "description": "Deliver Wi-Fi AP's information for an unboxing device to connect to it.\n",
1987     "parameters": [
1988         { "$ref": "#/parameters/interface-rw" },
1989         {
1990             "name": "body",
1991             "in": "body",
1992             "required": true,
1993             "schema": { "$ref": "#/definitions/WiFiConfUpdate" },
1994             "x-example":
1995                 {
1996                     "tnn": "Home_AP_SSID",
1997                     "cd": "Home_AP_PWD",
1998                     "wat": "WPA2_PSK",
1999                     "wet": "AES"
2000                 }
2001         }
2002     ],
2003     "responses": {
2004         "200": {
2005             "description": "",
2006             "x-example":
2007                 {
2008                     "tnn": "Home_AP_SSID",
2009                     "cd": "Home_AP_PWD",
2010                     "wat": "WPA2_PSK",
2011                     "wet": "AES"
2012                 }
2013             ,
2014             "schema": { "$ref": "#/definitions/WiFiConfUpdate" }
2015         }
2016     }
2017 }
2018 },
2019 },
2020 "parameters": {
2021     "interface-rw" : {
2022         "in" : "query",
2023         "name" : "if",
2024         "type" : "string",
2025         "enum" : ["oic.if.rw"]
2026     },
2027     "interface-baseline" : {
2028         "in" : "query",
2029         "name" : "if",
2030         "type" : "string",
2031         "enum" : ["oic.if.baseline"]
2032     },
2033     "interface-all" : {
2034         "in" : "query",
2035         "name" : "if",
2036         "type" : "string",
2037         "enum" : ["oic.if.baseline", "oic.if.rw"]
2038     }
2039 },
2040 "definitions": {
2041     "WiFiConf" :
2042     {
2043         "properties": {
2044             "cd": {
2045                 "description": "Indicates credential information of Wi-Fi AP",
2046                 "pattern": "^.*$",
2047                 "type": "string"
2048             },
2049             "swat": {
2050                 "description": "Indicates supported Wi-Fi Auth types. It can be multiple",
2051                 "items": {

```

```

2052         "description": "Indicates Wi-Fi Auth Type",
2053         "enum": [
2054             "None",
2055             "WEP",
2056             "WPA_PSK",
2057             "WPA2_PSK"
2058         ],
2059         "type": "string"
2060     },
2061     "readOnly": true,
2062     "type": "array"
2063 },
2064 "swet": {
2065     "description": "Indicates supported Wi-Fi Encryption types. It can be multiple",
2066     "items": {
2067         "description": "Indicates Wi-Fi Encryption Type",
2068         "enum": [
2069             "None",
2070             "WEP_64",
2071             "WEP_128",
2072             "TKIP",
2073             "AES",
2074             "TKIP_AES"
2075         ],
2076         "type": "string"
2077     },
2078     "readOnly": true,
2079     "type": "array"
2080 },
2081 "swf": {
2082     "description": "Indicates Supported Wi-Fi frequencies by the Enrollee. Can be multiple.
2083 Valid values are ('2.4G', '5G')",
2084     "items": {
2085         "pattern": "^(2\\.4|5)G$",
2086         "type": "string"
2087     },
2088     "readOnly": true,
2089     "type": "array"
2090 },
2091 "swmt": {
2092     "description": "Indicates supported Wi-Fi mode types. It can be multiple",
2093     "items": {
2094         "description": "Supported Wi-Fi Mode Type.",
2095         "enum": [
2096             "A",
2097             "B",
2098             "G",
2099             "N",
2100             "AC"
2101         ],
2102         "type": "string"
2103     },
2104     "readOnly": true,
2105     "type": "array"
2106 },
2107 "tnn": {
2108     "description": "Indicates Target Network Name (SSID of Wi-Fi AP)",
2109     "pattern": "^.*$",
2110     "type": "string"
2111 },
2112 "wat": {
2113     "description": "Indicates Wi-Fi Auth Type",
2114     "enum": [
2115         "None",
2116         "WEP",
2117         "WPA_PSK",
2118         "WPA2_PSK"
2119     ],
2120     "type": "string"
2121 },
2122 "wet": {

```

```

2123         "description": "Indicates Wi-Fi Encryption Type",
2124         "enum": [
2125             "None",
2126             "WEP_64",
2127             "WEP_128",
2128             "TKIP",
2129             "AES",
2130             "TKIP_AES"
2131         ],
2132         "type": "string"
2133     },
2134 },
2135 "required": [
2136     "swmt",
2137     "swf",
2138     "swat",
2139     "swet",
2140     "tnn",
2141     "wat",
2142     "wet"
2143 ],
2144 "type": "object"
2145 }
2146
2147 ,
2148 "WiFiConfUpdate" :
2149 {
2150     "properties": {
2151         "cd": {
2152             "description": "Indicates credential information of Wi-Fi AP",
2153             "pattern": "^.*$",
2154             "type": "string"
2155         },
2156         "tnn": {
2157             "description": "Indicates Target Network Name (SSID of Wi-Fi AP)",
2158             "pattern": "^.*$",
2159             "type": "string"
2160         },
2161         "wat": {
2162             "description": "Indicates Wi-Fi Auth Type",
2163             "enum": [
2164                 "None",
2165                 "WEP",
2166                 "WPA_PSK",
2167                 "WPA2_PSK"
2168             ]
2169         },
2170         "wet": {
2171             "description": "Indicates Wi-Fi Encryption Type",
2172             "enum": [
2173                 "None",
2174                 "WEP_64",
2175                 "WEP_128",
2176                 "TKIP",
2177                 "AES",
2178                 "TKIP_AES"
2179             ]
2180         }
2181     },
2182     "required": [
2183         "tnn",
2184         "wat",
2185         "wet"
2186     ],
2187     "type": "object"
2188 }
2189 }
2190 }
2191 }
2192

```

**B.3.5 Property Definition**

Property name	Value type	Mandatory	Access mode	Description
swet	array: see schema	yes	Read Only	Indicates supported Wi-Fi Encryption types. It can be multiple
wet	string	yes		Indicates Wi-Fi Encryption Type
swmt	array: see schema	yes	Read Only	Indicates supported Wi-Fi mode types. It can be multiple
swat	array: see schema	yes	Read Only	Indicates supported Wi-Fi Auth types. It can be multiple
cd	string			Indicates credential information of Wi-Fi AP
tnn	string	yes		Indicates Target Network Name (SSID of Wi-Fi AP)
wat	string	yes		Indicates Wi-Fi Auth Type
swf	array: see schema	yes	Read Only	Indicates Supported Wi-Fi frequencies by the Enrollee. Can be multiple. Valid values are ('2.4G', '5G')
tnn	string	yes		Indicates Target Network Name (SSID of Wi-Fi AP)
wet	multiple types: see schema	yes		Indicates Wi-Fi Encryption Type
wat	multiple types: see schema	yes		Indicates Wi-Fi Auth Type
cd	string			Indicates credential information of Wi-Fi AP

**B.3.6 CRUDN behaviour**

Resource	Create	Read	Update	Delete	Notify
/example/WiFiConfBaselineInterfaceResURI		get	post		