# OCF Bridging Specification

VERSION 1.3.0


CONTACT admin@openconnectivity.org

# 법적 고지 사항

이 표준에 기재된 내용 중 그 어느 것도 명시적 또는 암시적으로 기재 내용에 있어서 어떠한 형태의 사용 허가를 부여하거나 이 표준의 작성자 또는 개발자 중 어느 누구가 소유 또는 관할하는 어떠한 지식재산에 대해 어떠한 형태의 사용 허가도 부여하는 것을 의미하지 않습니다. 여기에 포함된 정보는 “있는 그대로” 제공되며, 적용 가능한 법에 의해 허용되는 최대 한도까지 이 시방서의 작성자 및 개발자는 특정한 목적을 위한 판매 적격성 또는 적합성의 암시적 보증을 포함하지만 이에 한정되지 않는 명시적 또는 암시적인 성문법 또는 불문법 상의 기타 모든 보증 및 조건에 대해 일절 책임을 지지 않습니다. OPEN CONNECTIVITY FOUNDATION, INC.는 비침해, 정확성, 또는 바이러스 비 감염에 대한 모든 보증에 대해서도 일절 책임을 지지 않습니다.

OCF 로고는 미국 및 다른 국가에서 Open Connectivity Foundation, Inc 의 상표입니다. *그 밖의 명칭 및 상표는 해당하는 소유자의 자산일 수 있습니다.



For Translation to Local Language

- 이 OCF 표준의 번역본은 OCF 기반의 제품 개발을 장려하고 이에 도움이 되도록 영문 원본으로부터 작성되었습니다. 영문 시방서의 정확한 번역을 위한 모든 노력을 기울이기는 하였지만 이 번역본을 규정으로 간주해서는 안 됩니다. OCF 인증 프로그램은 명백하게 영문 시방서를 기준으로 개발되어야 하며, 어떠한 면제 또는 면책 요구도 영문 시방서의 문구를 기준으로 평가되어야 합니다.
- 최신 영문판 시방서의 공개로부터 번역본의 공개까지는 소정의 지연이 있을 수 있습니다.
- OCF 시방서의 최신 영문판 및 해당 번역본에 관해서는 https://openconnectivity.org/developer/specifications를 참조하여 주십시오.

# 목차



 

 

# 그림 목차



 

# 표 목차



 

# 1 적용범위

이 표준은 OCF device 와 그 밖의 생태계 기술 간의 변환을 위한 framework 를 규정하고, OCF client 에 AllJoyn producer application 을 노출시키고 OCF server 를 AllJoyn consumer application 에 노출시키는 변환기의 동작을 규정한다. 특정 AllJoyn interface 에서 특정 OCF resource type 으로의 변환, 혹은 특정 OCF resource type 에서 특정 AllJoyn interface 로의 변환은 다른 시방서에 규정된다. AllJoyn 이외의 프로토콜의 변환은 이 시방서의 향후 버전에 규정될 예정이다. 이 표준은 더 상세한 문서에 재 정의되지 않는 한 일반적으로 적용되는 요구 사항을 제공한다.

# 2 인용표준

다음의 인용표준은 전체 또는 부분적으로 이 표준의 적용을 위해 필수적이다. 발행연도가 표기된 인용표준은 인용된 판만을 적용한다. 발행연도가 표기되지 않은 인용표준은 최신판(모든 추록을 포함)을 적용한다.

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

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

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

IEEE 754, *IEEE Standard for Floating-Point Arithmetic*, August 2008
http://ieeexplore.ieee.org/servlet/opac?punumber=4610933

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

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

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

IETF RFC 7049, *Concise Binary Object Representation (CBOR)*, October 2013
https://www.rfc-editor.org/info/rfc7049

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

JSON Schema Core, *JSON Schema: core definitions and terminology*, January 2013
http://json-schema.org/latest/json-schema-core.html

JSON Schema Validation, *JSON Schema: interactive and non interactive validation*, January 2013
http://json-schema.org/latest/json-schema-validation.html

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

OCF 1.0 Core Specification, *Open Connectivity Foundation Core Specification*, Version 1.3
Available at: https://openconnectivity.org/specs/OCF_Core_Specification_v1.3.0.pdf
Latest version available at: https://openconnectivity.org/specs/OCF_Core_Specification.pdf

OCF Security Specification, *Open Connectivity Foundation Security Specification*, Version 1.3
https://openconnectivity.org/specs/OCF_Security_Specification_v1.3.0.pdf
Latest version available at: https://openconnectivity.org/specs/OCF_Security_Specification.pdf

OCF Resource to AllJoyn Interface Mapping Specification, *Open Connectivity Foundation Resource to AllJoyn Interface Mapping Specification*, Version 1.3
Available at:
https://openconnectivity.org/specs/OCF_Resource_to_AllJoyn_Interface_Mapping_v1.3.0.pdf
Latest version available at:
https://openconnectivity.org/specs/OCF_Resource_to_AllJoyn_Interface_Mapping.pdf

OIC Core Specification, *Open Interconnect Consortium Core Specification*, Version 1.1
https://openconnectivity.org/specs/OIC_Core_Specification_v1.1.2.pdf

RAML Specification, ***RESTful** API Modeling Language,* Version 0.8.
https://github.com/raml-org/raml-spec/blob/master/versions/raml-08/raml-08.md

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

# 3 용어, 정의, 기호 및 약어

## 3.1 용어와 정의

### 3.1.1

**OCF Bridge Device**

네트워크에 존재하지만 OCF 프로토콜이 아닌 Bridged Protocol 을 사용해서 통신하는 OCF Device.

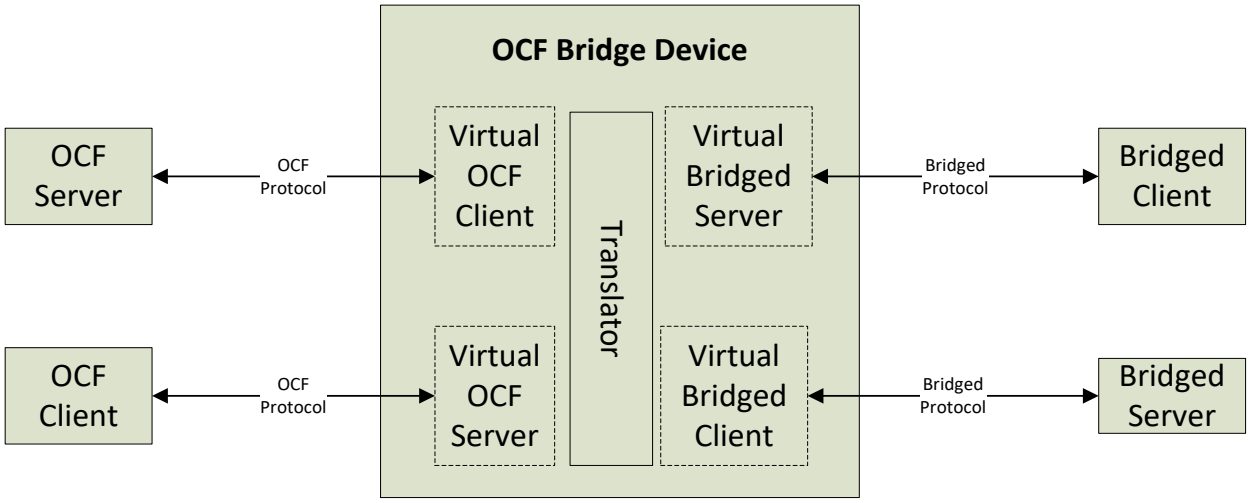


**그림 1. OCF Bridge Device 구성 요소**

### 3.1.2

**Bridged Protocol**

OCF 프로토콜로 또는 OCF 프로토콜로부터 변환되는 다른 프로토콜 (예: AllJoyn).

### 3.1.3

**Translator**

특정 Bridged Protocol 로 또는 특정 Bridged Protocol 로부터의 변환을 담당하는 OCF Bridge Device 구성 요소. 다른 Bridged Protocol 에 대해 동일한 OCF Bridge Device 상에 복수의 변환기가 존재할 수 있다.

### 3.1.4

**OCF Client**

OCF Server 의 OCF Resource 를 액세스하는 논리 개체로, 이 OCF Server 는 OCF Bridge Device 에 의해 노출되는 Virtual OCF Server 일 수 있다.

### 3.1.5

**Bridged Client**

Bridged Protocol 을 통해 데이터에 액세스하는 논리 개체. 예를 들어, AllJoyn Consumer application 은 Bridged Client 이다.

### 3.1.6

**Virtual OCF Client**

OCF Bridge Device 가 OCF Server 에 노출시키는 Bridged Client 의 논리적 표현.

 

### 3.1.7

**Virtual Bridged Client**

OCF Bridge Device 가 Bridged Server 에 노출시키는 OCF Client 의 논리적 표현.

### 3.1.8

**OCF Device**

하나 이상의 OCF role (OCF Client, OCF Server)을 가정하는 논리 개체. 동일한 물리적 platform 상에 복수의 OCF Device 가 존재할 수 있다.

### 3.1.9

**Virtual OCF Server**

OCF Bridge Device 가 OCF Client 에 노출시키는 Bridged Server 의 논리적 표현.

### 3.1.10

**Bridged Server**

Bridged Protocol 을 통해 데이터를 제공하는 논리 개체. 예를 들어, AllJoyn Producer 는 Bridged Server 이다. 동일한 물리적 platform 상에 복수의 Bridged Server 가 존재할 수 있다.

### 3.1.11

**Virtual Bridged Server**

OCF Bridge Device 가 Bridged Client 에 노출시키는 OCF Server 의 논리적 표현.

### 3.1.12

**OCF Resource**

OCF Framework 에 의해 모델링 되고 노출되는 구조결과물.

### 3.1.13

**Virtual OCF Resource**

OCF Bridge Device 가 OCF Client 로 노출시키는 Bridged Resource 의 논리적 표현.

### 3.1.14

**Bridged Resource**

ridged Protocol 에 의해 모델링 되고 노출되는 구조결과물. 예를 들어, AllJoyn object 는 Bridged Resource 이다.

### 3.1.15

**OCF Resource Property**

OCF Resource 를 통해 노출되는 메타데이터를 포함하는 주요 양상 또는 개념.

197 **3.1.16**

198 **OCF Resource Type**

199 OCF Resource 를 위한 data type 정의를 나타내는 OCF Resource Property.

200 **3.1.17**

201 **Bridged Resource Type**

202 Bridged Protocol 과 함께 사용되는 schema. 예를 들어, AllJoyn Interface 는 Bridged Resource
203 Type 이다.

204 **3.1.18**

205 **OCF Server**

206 Resource 상태 정보를 제공하고 resource 의 원격 제어를 허용하는 role 을 갖는 논리 개체.

207 **3.1.19**

208 **Onboarding Tool**

209 네트워크 내에서 특정 device 에 대한 소유권을 설정하고 device 가 동작 상태로 되도록 지원하는
210 특정 IoT network 내의 논리 개체로 OCF Security 시방서에 의해 정의.

211 **3.1.20**

212 **Bridged Device**

213 Bridged Client 또는 Bridged Server.

214 **3.1.21**

215 **Virtual OCF Device**

216 Virtual OCF Client 또는 Virtual OCF Server.

217 **3.2 기호와 약어**

218 **3.2.1**

219 **CRUDN**

220 Create Read Update Delete Notify
221 resource 에 대해 가능한 동작을 가리킨다.

222 **3.2.2**

223 **CSV**

224 Comma Separated Value List
225 콤마를 사용해서 하나의 string 내에 더 많은 field 를 포함하기 위한 구성. 값에 콤마가 포함되어
226 있을 때 그 앞에 "\"를 추가하면 콤마를 뺄 수 있다.

### 3.2.3

**OCF**

Open Connectivity Foundation

이들 시방서를 제작한 기구.

### 3.2.4

**RAML**

RESTful API Modeling Language

사실적으로 RESTful API 를 기술하는 간단명료한 방식 (RAML 규격) 참조.

## 3.3 규약

이 시방서에서, 다수의 용어, 조건, 메커니즘, 시이퀀스, 파라미터, 이벤트, 상태, 또는 유사한 용어는 각 단어의 첫 번째 문자를 대문자로 표기하고 나머지는 소문자로 표기한다 (예: Network Architecture). 이러한 단어가 소문자로 표기되었을 때는 일반적인 기술적 영어의 의미를 갖는다.

# 4 문서 규약 및 구성

이 표준의 목적을 위하여 용어와 정의는 OCF 1.0 Core 시방서에서 주어진다.

## 4.1 표기법

이 시방서에서 기능은 다음과 같이 필수(Required), 권고(Recommended), 허가(Allowed), 또는 기피(DEPRECATED)로 분류된다.

필수 (강제 또는 의무적)

> 이러한 기본 기능은 이 시방서를 준수하도록 구현되어야 한다. "하지 않는 것이 좋다"나 "금지된다" 등의 구절은 금지되는, 즉, 수행하는 경우 구현이 시방서를 준수하지 않음을 의미하는 행위를 나타낸다.

권고 (또는 제안)

> 이러한 기능은 이 시방서에 의해 지원되는 기능을 부가하며 구현되어야 한다. 권고 기능은, 통상적으로 중대한 복잡성의 증가 없이 이 시방서의 기능을 이용한다. 규정 준수 테스트를 위해 권고 기능이 구현된다면 이 가이드라인에 따르는 특정 요건을 만족해야 한다. 일부 권고 기능은 추후에 필수 요건이 될 수 있다. "하지 않는 것이 좋다"라는 표현은 허용되지만 권고하지 않는 작용을 나타낸다.

허가 (또는 허용)

> 이러한 기능은 필수적이지도 않을 뿐더러 권고되지도 않지만 기능이 구현된다면 이 가이드라인에 따르는 특정 요건을 만족해야 한다.

조건부 허용 (CA)

정의 또는 행위는 조건에 의존한다. 특정 조건이 만족되면 정의 또는 행위가 허용되고 그렇지 않으면 허용되지 않는다.

조건부 필수 (CR)

정의 또는 행위는 조건에 의존한다. 특정 조건이 만족되면 정의 또는 행위가 필수로 된다. 그렇지 않으면 특별한 기재가 없는 한 default 로 허용된다.

기피

이에 해당하는 기능은 이 시방서에서 설명은 하고 있지만 역 호환성을 제외하고는 구현되어서는 안 된다. 현재 시방서에 따르는 동작 동안 기피된 기능의 발생은 구현 동작에 어떤 영향도 끼치지 않으며 어떠한 에러 상태도 생성하지 않는다. 역 호환성은 기능이 구현되고 특정된 대로 기능할 것을 요구하지만 이 시방서에 따르는 구현에 의해 사용되어서는 안 된다.

문자 그대로 해석되는 String 은 "인용부호"를 사용한다.

강조하는 단어는 *이탤릭체*로 표기한다.

### 4.2 Data type

Data type 은 OCF 1.0 Core 시방서에 정의된다.

### 4.3 문서 구조

섹션 5 에서는 동작 시나리오를 논한다. 섹션 6 에서는 모든 OCF Bridge 에 대한 일반 요구 사항을 다루고, 섹션 7 에서는 AllJoyn 으로/으로부터 변환하는 브리지에 대한 특정 요구 사항을 다룬다. 이러한 사항들은 향후에 다른 프로토콜로의 변환을 정의하는 작업이 용이하도록 별도로 다뤄진다.

## 5 작동 시나리오

전반적인 목표는 다음과 같다.

1. Bridged Server 가 토종 OCF server 인것처럼 OCF client 에 보여지도록 한다.
2. OCF server 가 토종 non-OCF server 인것처럼 Bridged Client 에 보여지도록 한다.

### 5.1 "Deep translation" vs. "on-the-fly"

Bridged Protocol(예: AllJoyn)과 OCF protocol 간의 서비스 변환 시에는 두 가지의 가능한 변환 유형이 있다. 변환기가 "Deep Translator"를 사용하여 통신하는 경우에 Birded Protocol 에 사용되는 데이터 모델은 OCF Resource 와 연동을 위해 매핑되고, OCF Client 혹은 Bridged Client 는 변환됨을 인식 못하고 상호운영 될 수 있어야 한다.

"딥 변환"은 절차가 유형의 매핑을 크게 넘어서므로 이 표준의 적용범위에서 벗어난다. 예를 들어, 변환기의 한 쪽의 client 는 0 에서 255 사이의 8 비트 값으로 강도를 나타내고자 결정하는 반면, 다른 쪽의 device 는 0.0 에서 1.0 사이의 부동소수점 숫자의 표현을 선택한 경우가 있다. 절차가 변환기 내에 상태를 저장할 것을 요구하는 경우도 있을 수 있다. 어느 경우에도 그러한 변환의 프로그래밍은 상당한 노력과 양쪽의 메커니즘의 연구를 필요로 한다.

변환의 다른 유형인 "on-the-fly" 또는 "일대일" 변환은 변환기 측에서 대상 device 고유의 schema 에 대한 사전 지식을 필요로 하지 않는다. 대신에 통신 대상의 한쪽, 일반적으로 client 애플리케이션 쪽이 부담을 지게 된다. 이것은 "on-the-fly" 변환이 항상 제조사 확장으로 Bridged Resource Type 및 OCF Resource Type 을 생성하는 데서 기인한다.

AllJoyn 의 경우, 딥 변환은 OCF ASA 매핑에 규정되고, 임시 변환은 이 표준의 섹션 7.2 에서 다뤄진다.

## 5.2 introspection 의 사용

가능한 한 변환 코드는 메시지의 송신자와 수신자가 무엇을 예측하는지를 나타내는 메타데이터를 사용해야 한다. 예를 들어, AllJoyn Certified device 는 노출하는 각각의 객체와 interface 에 대해 introspection 데이터를 전달해야 한다. OIC 1.1 Core 시방서에는 그러한 요구 사항이 없지만 OCF 1.0 Core 시방서에는 있다. 메타데이터를 사용할 수 있으면 변환기는 착신 페이로드를 정확하게 수신자가 예측하는 형식으로 변환하고 응답 변환 시에 정보를 사용하여 더 유용한 메시지를 작성해야 한다.

예를 들어, AllJoyn 의 경우, 예측되는 상호 작용 목록은 아래와 같다.

| 메시지 유형 | 송신자 | 수신자 | 메타데이터 |
|---|---|---|---|
| Request | AllJoyn 16.10 | OIC 1.1 | 사용 불가 |
| Request | AllJoyn 16.10 | OCF 1.0 | 사용 가능 |
| Request | OIC 1.1 or OCF 1.0 | AllJoyn 16.10 | 사용 가능 |
| Response | AllJoyn 16.10 | OIC 1.1 or OCF 1.0 | 사용 가능 |
| Response | OIC 1.1 | AllJoyn 16.10 | 사용 불가 |
| Response | OCF 1.0 | AllJoyn 16.10 | 사용 가능 |

## 5.3 안정성 및 데이터 손실

이 표준에 규정된 변화 프로세스를 거치면서 동일한 원래 메시지가 재생되지는 않는다. 하지만, 프로세스를 통해 메시지의 데이터 또는 정확도가 손실되지는 않는다. OCF 와 AllJoyn payload 형식은 이 표준에서 고려되지 않은 향후의 확장을 허용한다는 점에 주의하기 바란다.

단, 세 번째 라운드의 변환은 동일한 정보가 제공되는 한 이전에 생성된 것과 동일한 메시지를 재생해야 한다. 즉, 위의 체인에서 페이로드 2 와 4 및 3 과 5 는 동일해야 한다.

# 6 OCF Bridge Device

이 섹션에서는 그림 2 에 도시된 device 와 같은 OCF Bridge Device 의 기능을 설명한다.

OCF Bridge Device 는 네트워크 상의 하나 이상의 Bridged Device 를 Virtual OCF Device 로 표현하거나 OCF Device 를 네트워크 상의 다른 프로토콜을 사용하는 Virtual Device 로 표현한다. 브리지되는 Device 는 이 표준의 적용범위에서 벗어난다. 토종 OCF Device 와 Virtual Bridged Device 간의 차이점은 OCF Bridge Device 내에서 어떻게 encapsulation 되는지에만 있다.

OCF Bridge Device 는 OCF 상에서 "oic.d.bridge"라는 Device Type 으로 나타내야 한다. 이것은 OCF Client 에게 발견된 Device 가 브리징 기능을 수행함을 명백하게 알려준다. 이것은 다음과 같은 점에서 유용하다: 1) 홈 네트워크 구축 시에 Bridged Device 가 존재하지 않을 때 브리지가 연결 가능하고 기능을 하는지를 Client 가 판단할 수 있다; 2) 브리지가 지원하는 기존의 기능을 고려해서 브리지 상에서 특정 액션을 수행할 수 있도록 한다; 3) 사용자 대신에 문제해결 및 유지보수에 유용한 브리징 기능을 제공하는 모든 device 를 명확하게 발견할 수 있도록 한다. 그러한 device 가 발견되면 OCF Bridge Device 상의 노출된 Resource 가 다른 device 를 기술한다. 예를 들면 그림 2 에 보이는 바와 같다.

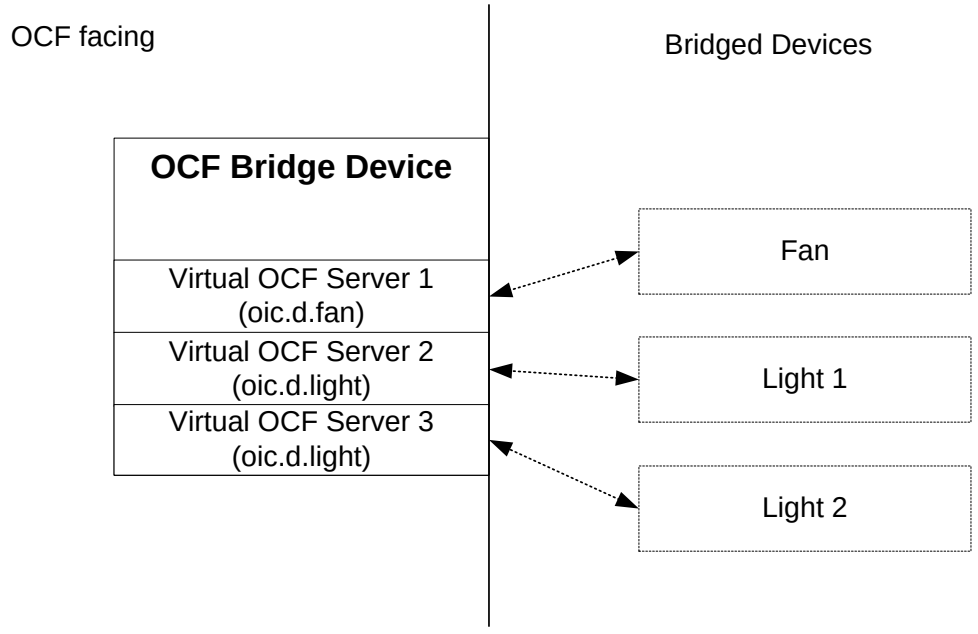


**그림 2: non-OCF device 를 브리징 하는 OCF Bridge Device 의 개념도**

OCF Bridge Device 는 시작하는 동안 device 집합을 생성할 것으로 예측된다. Bridged Device 가 브리지에 추가되거나 브리지로부터 제거됨에 따라 노출된 Virtual OCF Device 집합은 변경될 수 있다. 이때 Bridged Device 의 추가 또는 제거는 구현에 따른다. OCF Bridge Device 가 노출한 Virtual OCF Device 집합이 변경되면 "/oic/res"을 subscribe 하는 모든 OCF Client 에게 변경 내용을 통지해야 한다.

## 6.1 Resource Discovery

OCF Bridge Device 는 Bridged 네트워크 또는 OCF 네트워크에 들어오거나 나가는 device 를 검출해야 한다. 네트워크 상의 device 의 출입을 검출할 수 있는 기존의 신뢰할 만한 메커니즘이 없을 때는 OCF Bridge Device 가 주기적인 네트워크의 폴링을 통해 device 의 출입을 검출한다. 예를 들어, OCF network 의 경우, COAP multicast discovery ("/oic/res"의 multicast RETRIEVE)를 사용한다. OCF Bridge Device 구현에는 30 초 플러스 마이너스 수 초의 랜덤 지연의 폴링 간격을 사용하는 것이 바람직하다.

OCF Bridge Device 는 노출된 Bridged device 대신에 network discovery command 에 응답해야 한다. Resource 를 가지고 있는 모든 Bridged device 는 Bridge 에 존재하는 "/oic/res"에 나열되어야 한다. "/oic/res" 상의 RETRIEVE 에 대한 응답은 RETRIEVE 요청에 매칭되는 device 만 포함해야 한다.

브리지 상의 "/oic/res"에 의해 노출되는 각 Link 로부터 결정되는 resource reference 는 고유해야 한다. "/oic/res" 내의 Property 와 Link parameter 의 개체에 대해 브리지는 OCF 1.0 Core 시방서에 정의된 요구 사항을 만족해야 한다.

예를 들어, 그림 2 에 보이는 바와 같이 OCF Bridge Device 가 fan 과 light 에 대한 Virtual OCF Server 를 노출시키면, 브리지는 기존 OIC 1.1 client 가 "/oic/res"에 대한 RETRIEVE 를 수행한 JSON 에 대응하는 정보를 리턴 할 수 있다. (어떤 것이 리턴 되는지는 JSON 형식에 기술되지 않고 OCF 1.0 Core 시방서에 정의된 적절한 인코딩 내에 포함된다.)

```
[
  {
    "di": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "links": [
      {
        "href": "coap://[2001:db8:a::b1d4]:55555/oic/res",
        "rel": "self",
        "rt": ["oic.wk.res"],
        "if": ["oic.if.ll", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 11111}
      },
      {
        "href": "/oic/d",
        "rt": ["oic.wk.d", "oic.d.bridge"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 11111}
      },
```

 

```
      {
        "href": "/oic/p",
        "rt": ["oic.wk.p"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 11111}
      },
      {
        "href": "/mySecureMode",
        "rt": ["oic.r.securemode"],
        "if": ["oic.if.rw", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 11111}
      },
      {
        "href": "/oic/sec/doxm",
        "rt": ["oic.r.doxm"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 11111}
      },
      {
        "href": "/oic/sec/pstat",
        "rt": ["oic.r.pstat"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 11111}
      },
      {
        "href": "/oic/sec/cred",
        "rt": ["oic.r.cred"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 11111}
      },
      {
        "href": "/oic/sec/acl2",
        "rt": ["oic.r.acl2"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 11111}
      },
      {
        "href": "/myIntrospection",
        "rt": ["oic.wk.introspection"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 11111}
      }
    ]
  },
  {
    "di": "88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "links": [
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/res",
        "rt": ["oic.wk.res"],
        "if": ["oic.if.ll", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/d",
        "rt": ["oic.wk.d", "oic.d.fan", "oic.d.virtual"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/p",
        "rt": ["oic.wk.p"],
        "if": ["oic.if.r", "oic.if.baseline"],
```

 

```
        "p": {"bm": 3, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/myFan",
        "rt": ["oic.r.switch.binary"],
        "if": ["oic.if.a", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/sec/doxm",
        "rt": ["oic.r.doxm"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/sec/pstat",
        "rt": ["oic.r.pstat"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/sec/cred",
        "rt": ["oic.r.cred"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/oic/sec/acl2",
        "rt": ["oic.r.acl2"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 22222}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:22222/myFanIntrospection",
        "rt": ["oic.wk.introspection"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 22222}
      }
    ]
  },
  {
    "di": "dc70373c-1e8d-4fb3-962e-017eaa863989",
    "links": [
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/res",
        "rt": ["oic.wk.res"],
        "if": ["oic.if.ll", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/d",
        "rt": ["oic.wk.d", "oic.d.light", "oic.d.virtual"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/p",
        "rt": ["oic.wk.p"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/myLight",
```

 

```
        "rt": ["oic.r.switch.binary"],
        "if": ["oic.if.a", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/sec/doxm",
        "rt": ["oic.r.doxm"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/sec/pstat",
        "rt": ["oic.r.pstat"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/sec/cred",
        "rt": ["oic.r.cred"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/oic/sec/acl2",
        "rt": ["oic.r.acl2"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 33333}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:33333/myLightIntrospection",
        "rt": ["oic.wk.introspection"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 33333}
      }
    ]
  },
  {
    "di": "8202138e-aa22-452c-b512-9ebad02bef7c",
    "links": [
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/res",
        "rt": ["oic.wk.res"],
        "if": ["oic.if.ll", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/d",
        "rt": ["oic.wk.d", "oic.d.light", "oic.d.virtual"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/p",
        "rt": ["oic.wk.p"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/myLight",
        "rt": ["oic.r.switch.binary"],
        "if": ["oic.if.a", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 44444}
      },
```

 

```
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/sec/doxm",
        "rt": ["oic.r.doxm"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/sec/pstat",
        "rt": ["oic.r.pstat"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/sec/cred",
        "rt": ["oic.r.cred"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/oic/sec/acl2",
        "rt": ["oic.r.acl2"],
        "if": ["oic.if.baseline"],
        "p": {"bm": 1, "sec": true, "port": 44444}
      },
      {
        "href": "coaps://[2001:db8:a::b1d4]:44444/myLightIntrospection",
        "rt": ["oic.wk.introspection"],
        "if": ["oic.if.r", "oic.if.baseline"],
        "p": {"bm": 3, "sec": true, "port": 44444}
      }
    ]
  }
]
```

위의 예는 각 Virtual OCF Server 가 브리지에 의해 노출된 "di" 및 endpoint 를 갖고, 각 Virtual OCF Server 에 대해 "/oic/p"와 "/oic/d"가 사용 가능함을 보여준다.

OCF Client 가 "application/vnd.ocf+cbor"의 컨텐츠 형식을 요구하면 동일한 브리지가 아래와 같은 JSON 에 해당하는 정보를 리턴 한다. (어떤 것이 리턴 되는지는 JSON 형식에 기술되지 않고 OCF 1.0 Core 시방서에 정의된 적절한 인코딩 내에 포함된다.)

```
[
 {
   "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
   "href": "/oic/res",
   "rel": "self",
   "rt": ["oic.wk.res"],
   "if": ["oic.if.ll", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coap://[2001:db8:a::b1d4]:55555"},
           {"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
 },
 {
   "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
   "href": "/oic/d",
   "rt": ["oic.wk.d", "oic.d.bridge"],
   "if": ["oic.if.r", "oic.if.baseline"],
   "p": {"bm": 3},
```

 

```
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/oic/p",
    "rt": ["oic.wk.p"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/mySecureMode",
    "rt": ["oic.r.securemode"],
    "if": ["oic.if.rw", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/oic/sec/doxm",
    "rt": ["oic.r.doxm"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/oic/sec/pstat",
    "rt": ["oic.r.pstat"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/oic/sec/cred",
    "rt": ["oic.r.cred"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/oic/sec/acl2",
    "rt": ["oic.r.acl2"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },
  {
    "anchor": "ocf://e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
    "href": "/myIntrospection",
    "rt": ["oic.wk.introspection"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]
  },


  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/res",
```

 

```
    "rt": ["oic.wk.res"],
    "if": ["oic.if.ll", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/d",
    "rt": ["oic.wk.d", "oic.d.fan", "oic.d.virtual"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/p",
    "rt": ["oic.wk.p"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/myFan",
    "rt": ["oic.r.switch.binary"],
    "if": ["oic.if.a", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/sec/doxm",
    "rt": ["oic.r.doxm"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/sec/pstat",
    "rt": ["oic.r.pstat"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/sec/cred",
    "rt": ["oic.r.cred"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
    "href": "/oic/sec/acl2",
    "rt": ["oic.r.acl2"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },
  {
    "anchor": "ocf://88b7c7f0-4b51-4e0a-9faa-cfb439fd7f49",
```

 

```
    "href": "/myFanIntrospection",
    "rt": ["oic.wk.introspection"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:22222"}]
  },

  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/oic/res",
    "rt": ["oic.wk.res"],
    "if": ["oic.if.ll", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/oic/d",
    "rt": ["oic.wk.d", "oic.d.light", "oic.d.virtual"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/oic/p",
    "rt": ["oic.wk.p"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/myLight",
    "rt": ["oic.r.switch.binary"],
    "if": ["oic.if.a", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/oic/sec/doxm",
    "rt": ["oic.r.doxm"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/oic/sec/pstat",
    "rt": ["oic.r.pstat"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
  {
    "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
    "href": "/oic/sec/cred",
    "rt": ["oic.r.cred"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
  },
```

 

```
 {
   "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
   "href": "/oic/sec/acl2",
   "rt": ["oic.r.acl2"],
   "if": ["oic.if.baseline"],
   "p": {"bm": 1},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
 },
 {
   "anchor": "ocf://dc70373c-1e8d-4fb3-962e-017eaa863989",
   "href": "/myLightIntrospection",
   "rt": ["oic.wk.introspection"],
   "if": ["oic.if.r", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:33333"}]
 },

 {
   "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
   "href": "/oic/res",
   "rt": ["oic.wk.res"],
   "if": ["oic.if.ll", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
 },
 {
   "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
   "href": "/oic/d",
   "rt": ["oic.wk.d", "oic.d.light", "oic.d.virtual"],
   "if": ["oic.if.r", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
 },
 {
   "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
   "href": "/oic/p",
   "rt": ["oic.wk.p"],
   "if": ["oic.if.r", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
 },
 {
   "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
   "href": "/myLight",
   "rt": ["oic.r.switch.binary"],
   "if": ["oic.if.a", "oic.if.baseline"],
   "p": {"bm": 3},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
 },
 {
   "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
   "href": "/oic/sec/doxm",
   "rt": ["oic.r.doxm"],
   "if": ["oic.if.baseline"],
   "p": {"bm": 1},
   "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
 },
 {
   "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
   "href": "/oic/sec/pstat",
   "rt": ["oic.r.pstat"],
   "if": ["oic.if.baseline"],
   "p": {"bm": 1},
```

 

```
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
  },
   {
    "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
    "href": "/oic/sec/cred",
    "rt": ["oic.r.cred"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
  },
  {
    "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
    "href": "/oic/sec/acl2",
    "rt": ["oic.r.acl2"],
    "if": ["oic.if.baseline"],
    "p": {"bm": 1},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
  },
  {
    "anchor": "ocf://8202138e-aa22-452c-b512-9ebad02bef7c",
    "href": "/myLightIntrospection",
    "rt": ["oic.wk.introspection"],
    "if": ["oic.if.r", "oic.if.baseline"],
    "p": {"bm": 3},
    "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:44444"}]
  }
 ]
```

## 6.2 일반 요구 사항

변환기는 각 device 의 프로토콜 독립적인 UUID (OCF 의 “piid”)를 확인하고 해당하는 Bridged Protocol 을 통해 처음으로 보여진 device 를 Bridged Protocol 내에 다시 공고하지 않는다. 변환기가 Bridged Protocol 을 통해 다른 변환기를 통해 OCF 내에 노출된 Bridged Protocol device 를 관측하면 해당 device 의 변환을 중지한다. 마찬가지로, 변환기는 OCF 내에 OCF Device 를 다시 공고하지 않으며, OCF 를 통해 다른 변환기에 의해 Bridged Protocol 에 노출된 OCF device 를 관측하면 해당 device 의 변환을 중지한다. 이를 위해 변환기는 device 가 이미 변환된 것인지 판단할 수 있어야 한다. Virtual OCF Device 는 OCF network 상에 “oic.d.virtual”이라는 Device Type 으로 나타내야 한다. 이렇게 함으로써 복수의 변환기가 존재할 때 device 가 이미 변환된 것인지를 변환기가 판단할 수 있게 된다. 변환기가 non-OCF network 상에서 device 가 이미 변환된 것인지를 판단하는 방법에 관해서는 아래의 프로토콜 특정 섹션에서 설명한다.

변환기는 반드시 중복된 virtual device(protocol 과 무관한 같은 UUID 를 갖는)가 네트워크에 있는지 검출해야 하고, 다른 네트워크로 중복된 device 를 변환하지 않도록 복수의 해당하는 virtual device 를 생성하지 않아야 한다.

각 Bridged Server 는 자신의 endpoint 와 “/oic/d” 및 “/oic/p”를 갖고 독립된 Virtual OCF Server 로서 노출되어야 한다. Virtual OCF Server 의 “/oic/res” resource 는 resource directory 를 사용하는 일반적인 OCF Server 의 것과 동일하다. 즉, multicast discovery 요청에 응답하지 않고 (OCF Bridge Device 가 대신 응답하므로), unicast query 가 “rel”=“hosts” 관계와 OCF Bridge Device 가 아님을

 

나타내는 적절한 “anchor”로 resource 를 나열하는 응답을 유도한다. 이렇게 함으로써 변환 전반에 걸쳐서 모든 platform-specific, device-specific 및 resource-specific 필드가 보존된다.

변환기에 의해 제공되는 introspection 데이터는 그 시점에서 변환기에 의해 노출되는 모든 virtual device (및 해당하는 resource)에 대한 정보를 포함해야 한다. 이것은 신규 virtual device 가 노출되기 전후의 introspection 데이터가 다름을 의미한다.

### 6.3 보안

OCF Bridge Device 는 다른 onboardee 와 마찬가지로 OCF 소유권 양도를 거친다. 별도로, OCF Bridge Device 는 통상적으로 다른 onboardee 와 마찬가지로 Bridged Protocol 의 소유권 양도 메커니즘 (예: AllJoyn 주장)을 거친다.

OCF Bridge Device 는 필드 업데이트가 가능해야 한다. (본 요구 사항은 테스트를 필요로 하지 않지만 제조사 신고를 통해 보증되어야 한다.)

관리자가 허용을 동의하지 않는 한 (섹션 9.2 참조), 변환기는 안전한 연결을 보장할 수 없는 device 와 연결을 노출시켜서는 안 된다.

각 Virtual OCF Device 에는 OCF Onboarding tool 에 의한 보안이 제공되어야 한다. 각 Virtual Bridged Device 는 필요에 따라 Bridged 생태계 내에서 프로비저닝 되어야 한다. 다시 말하면, Virtual Device 는 물리적 Device 와 동일하게 취급된다. Virtual Device 는 해당하는 네트워크 내에서 프로비저닝 되어야 하는 독립체이다.

섹션 6.2 에 규정된 바와 같이, 변환기는 non-OCF Device 와 이에 대응하는 Virtual OCF Device 를 관련 짓는 “piid” 값을 제공해야 한다. Onboarding Tool 은 Virtual OCF Device 에 대해 대응하는 non-OCF Device 의 보안 설정과 동등한 보안 설정을 자동으로 생성해서 사용자 입력의 필요를 없애거나 줄임으로써 Onboarding user 의 편의를 향상시킬 수 있다. Onboarding 에 관한 자세한 정보는 OCF Security 시방서를 참조하기 바란다.

각 Virtual Device 는 연결된 생태계의 보안 요구 사항을 구현해야 한다. 예를 들어, 각 Virtual OCF Device 는 OCF 1.0 Core 시방서 및 OCF Security 시방서에 의해 요구되는 작용을 구현해야 한다. 각 Virtual OCF Device 는 Onboarding Tool 로부터 수신한 보안 설정에 따라 인증, 액세스 제어, 및 암호화를 수행해야 한다.

변환기의 아키텍처에 따라서는 인증 및 액세스 제어가 각 생태계 내에서 이루어질 수 있지만 변환기 내에서는 이루어지지 않는다. 예를 들어, OCF Client 가 Virtual OCF Server 로 요청을 전송할 때,

- OCF Client 로부터 요청 수신 시에 Virtual OCF Server 에 의해 인증 및 액세스 제어를 수행할 수 있다.

 

- 요청이 변환기를 거쳐 해당하는 Virtual Bridged Client 로 전달될 때 변환기는 인증 또는 액세스 제어를 수행하지 않을 수 있다.
- Bridged 생태계의 보안 모델에 따라, Virtual Bridged Client 로부터 요청 수신 시에 target Bridged Server 에 의해 인증 및 액세스 제어를 수행할 수 있다.

변환기는 Virtual Bridged Device 를 통해 Bridged Client 로부터 비 암호화 데이터를 수신할 수 있다. 변환된 메시지는, OCF Device 가 암호화를 요구하면, target OCF Device 에 전송되기 전에 대응하는 Virtual OCF Client 에 의해 암호화 되어야 한다.

변환기는 Virtual OCF Server 를 통해 OCF Client 로부터 비 암호화 데이터를 수신할 수 있다. 변환 후에, Bridged Server 가 암호화를 요구하면, 이 데이터는 target Bridged Server 로 전송되기 전에 대응하는 Virtual Bridged Client 에 의해 암호화 되어야 한다.

변환기는 변환기를 통해 Virtual Client 와 Virtual Server 간에 데이터가 전송되는 동안 전송되는 데이터를 보호해야만 한다. 예를 들어, 변환기가 네트워크를 통해 데이터를 전송하는 경우, 변환기는 이 통신에 관여하는 모든 피어 간에 적절한 인증 및 액세스 제어를 수행하고 데이터를 암호화 해야 한다.

### 6.3.1 OCF 생태계와 Bridged Device 의 통신 차단

OCF Onboarding Tool 은 Bridge Device 의 "oic.r.securemode" Resource 를 사용해서 안전하게 통신하지 않는 모든 Bridged Device 와 모든 OCF Device 의 통신을 차단할 수 있다.

뿐만 아니라, OCF Onboarding Tool 은, 임의의 OCF Device 와 마찬가지로, 특정 Virtual OCF Client 의 모든 OCF Server 와의 통신을 차단하거나, 모든 OCF Client 의 특정 Virtual OCF Server 와의 통신을 차단할 수 있다. 소프트 리셋 상태에 관한 자세한 정보는 OCF Security 시방서의 섹션 8.5 를 참조하기 바란다.

# 7 AllJoyn 변환

## 7.1 AllJoyn Translator 에 고유한 요구 사항

변환기는 AllJoyn Router Node 가 된다. (이것은 사용자가 다른 device 의 추가 구입 없이 인증된 OCF Bridge Device 와 임의의 AllJoyn device 간의 통신할 수 있도록 하기 위한 요구 사항이다.)

이 섹션의 요구 사항은 알고리즘 변환 시에 적용되고, 딥 변환 관련 시방서에서 별도로 규정되지 않는 한 딥 변환에 대해 default 로 적용된다.

### 7.1.1 AllJoyn producer device 의 OCF Client 에의 노출

OCF Security 시방서에 규정된 바와 같이, OCF Device (Virtual OCF Device 포함)의 "di" property 값은 해당하는 Virtual OCF Device 의 Onboarding 의 일부로 설정되어야 한다.

 

각 AllJoyn object 는 하나 이상의 Virtual OCF Resource 에 매핑 된다. 모든 AllJoyn interface 가 동일한 resource 의 resource type 으로 변환되는 경우 (아래와 같이), 단일 Virtual OCF Resource 가 존재하고 Virtual OCF Resource 의 URI 경로 구성이 AllJoyn object 경로가 되어야 한다 (여기서 AllJoyn object 경로 내의 "_h"는 각각 "-" (하이픈)으로 변환되고, AllJoyn object 경로 내의 "_d"는 각각 "." (도트)로 변환되고, AllJoyn object 경로 내의 "_t"는 각각 "~" (물결표)로 변환되고 AllJoyn object 경로 내의 "_u"는 각각 "_" (밑줄)로 변환된다). 그렇지 않으면, 해당하는 경로를 가진 Resource 가 link 의 Collection 인 ["oic.wk.col", "oic.r.alljoynobject"]의 Resource type 으로 존재하게 된다. "oic.r.alljoynobject"는 섹션 9.3 에 정의되고, collection 내의 항목은 아래와 같이 변환된 Resource Type 을 갖는 Resource 이다.

각 Virtual OCF Device 에 대한 "/oic/d"의 "piid" property 값은, AllJoyn About Announce 신호 내의 OCF 정의 AllJoyn field "org.openconnectivity.piid"가 존재하는 경우 이 값이 되고, 그렇지 않으면 다음과 같이 변환기에 의해 산출된다.

- AllJoyn device 가 보안을 지원하면 "piid" property 값은 피어 GUID 가 된다.
- AllJoyn device 가 보안을 지원하지 않지만 device 가 어떠한 형태로든 브리지 되어 있으면 (섹션 9.2 참조), "piid" property 값은 DeviceId 값 (null 종료 제외)과 AppId property (About 데이터 내)로부터 유도되고, DeviceID 값( null 종료를 포함하지 않음)과 Appid 바이트를 연결하고 SHA-1 을 해시 알고리즘을 사용하여 IETF 4122 섹션 4.3 에 규정된 알고리즘에서 사용되는 "name"으로 그 결과를 사용한다. 8f0e4e90-79e5-11e6-bdf4-0800200c9a66 을 name space ID 로 사용한다. (이것은 독립된 OCF Bridge Device 를 통해 노출되는 AllJoyn device 의 중복 제거가 가능한 문제를 해결하기 위한 것이다.)

변환기 구현에는 임의의 AllJoyn interface name 에 매칭되는 AllJoyn About Announce 신호를 수신하는 것이 바람직하다. 이렇게 함으로써 이러한 신호로부터 수신한 정보의 cache 를 유지하고, OCF Client 로부터의 "/oic/res" query 를 신속하게 처리하기 위해 (query 처리 시에 Announce 신호를 기다리지 않고) cache 를 사용할 수 있다.

변환기 구현에는 Virtual AllJoyn Device 상의 대응하는 resource 에 관계된 client 가 존재할 때만 그 밖의 신호 (property 의 EmitsChangedSignal)를 수신하는 것이 바람직하다.

AllJoyn interface 에는 다음과 같이 복수 개의 유형이 있다.

- AllJoyn interface 가 AllJoyn 과 OCF 양측에 표준 형식이 존재하는 제대로 정의된 (OCF ASA 매핑 또는 아래의 섹션 7.1.1.1 에 정의된) interface 집합 내에 존재하면,
    a. 변환기는 해당하는 interface 를 특정해서 변환하기 위한 시방서를 따르거나
    b. 변환기는 AllJoyn interface 를 변환하지 않는다.
- AllJoyn interface 가 제대로 정의된 집합 내에 존재하지 않으면,
    a. 변환기는 AllJoyn interface 를 변환하지 않거나

 

b. 변환기는 AllJoyn interface name 을 OCF resource type name 으로 변환함으로써 섹션 7.2 에 규정된 바와 같이 AllJoyn interface 를 커스텀/제조사 정의 Resource Type 으로 알고리즘적으로 매핑한다.

AllJoyn interface name 은 다음과 같이 Device Type 또는 하나 이상의 OCF Resource Type 의 집합으로 변환된다.

1) AllJoyn interface 가 member 를 갖고 있으면 아래에 설명된 접미어 ".<seeBelow>"를 덧붙인다.
2) 전체 string 내에 존재하는 각 대문자를 하이픈과 이에 뒤따르는 소문자로 대체한다 (예를 들어, "A"를 "-a"로 변환한다).
3) 밑줄 뒤에 소문자 또는 하이픈이 뒤따르면 밑줄을 두 개의 하이픈으로 대체한다 (예를 들어, "_a"를 "--a"로 대체하고, "_-a"를 "---a"로 변환한다).
4) 남아있는 각각의 밑줄을 하이픈으로 대체한다 (예를 들어, "_1"를 "-1"로 변환한다).
5) 접두어 "x."를 덧붙인다.

아래의 표에 몇 가지 예를 보인다. 처음 세 개는 특이한 OCF name 으로 변환된 일반적인 AllJoyn name 을 나타낸다. 마지막 세 개는 일반적인 OCF name 으로 변환된 특이한 AllJoyn name 을 나타낸다. ("xn--"는 Internationalized Domain Name 의 Punycode-encoded 형식에 대한 일반적인 도메인 명칭 접두어이므로 일반적인 제조사 고유의 OCF name 으로 나타난다.)

| **AllJoyn 명칭으로부터** | **OCF 명칭으로** |
|---|---|
| example.Widget | x.example.-widget |
| example.my_widget | x.example.my—widget |
| example.My_Widget | x.example.-my---widget |
| xn_p1ai.example | x.xn--p1ai.example |
| xn__90ae.example | x.xn--90ae.example |
| example.myName_1 | x.example.my-name-1 |

member 를 가지고 있고 알고리즘적인 매핑을 사용하는 각각의 AllJoyn interface 는 다음과 같이 하나 이상의 Resource Type 에 매핑된다.

- 동일한 EmitsChangedSignal 값을 갖는 AllJoyn Property 는 <seeBelow> 라벨의 값이 EmitsChangedSignal 의 값인 동일한 Resource Type 에 매핑된다. EmitsChangedSignal 값이 "const" 또는 "false"인 AllJoyn Property 는 관측 가능하지 않은 Resource 에 매핑되고, EmitsChangedSignal 값이 "true" 또는 "invalidates"인 AllJoyn Property 는 관측 가능한 Resource 에 매핑된다. AllJoyn interface 의 Version property 는 introspection XML 에 특정되지 않더라도 항상 "const"의 EmitsChangedSignal 값을 갖는 것으로 간주된다.

 

Resource Type 내의 각각의 property 명칭은 “<ResourceType>.<AllJoynPropertyName>” 이어야 한다 (여기서<AllJoynPropertyName> 내의 “_d”는 각각 “.” (도트)로 변환되고 <AllJoynPropertyName> 내의 “_h”는 각각 “-” (하이픈)으로 변환된다).

- 액세스 “readwrite”를 가진 AllJoyn Property 를 매핑하는 Resource Type 은 “oic.if.rw” Interface 를 지원해야 한다. 액세스 “read”를 가진 AllJoyn Property 를 매핑하는 Resource Type 은 “oic.if.r” Interface 를 지원해야 한다. “oic.if.rw”와 “oic.if.r” Interface 를 지원하는 Resource Type 은 “oic.if.r”를 default Interface 로 선택한다.
- 각각의 AllJoyn Method 는 독립된 Resource Type 에 매핑되고, 여기서 <seeBelow> 라벨의 값은 AllJoyn Method 이름이 된다. Resource Type 은 “oic.if.rw” Interface 를 지원해야 한다. AllJoyn Method 의 각 인자는 Resource Type 상의 독립된 Property 에 매핑되어야 하고, 여기서 해당하는 Property 의 이름은 동일한 Resource 상의 모든 Resource Type 에 걸쳐서 고유함을 갖도록 “<ResourceType>arg<#>”가 접두어로 붙는다 (여기서, <#>은 AllJoyn introspection xml 내 인자의 0 인덱스 위치). 그러므로, AllJoyn argument 이름이 지정되지 않을 때는 property 이름은 “<ResourceType>arg<#>”로 된다 (여기서, <#>은 AllJoyn introspection XML 내 인자의 0 인덱스 위치). 뿐만 아니라, 해당하는 Resource Type 은 property 의 나머지가 유효한 값을 갖는지를 나타내는 추가 <ResourceType>validity” property 를 갖는다. 값이 UPDATE 응답의 일부로 전송되면, 유효한 property 가 참이 되고 모든 다른 property 가 유효한 값을 갖는다. RETRIEVE (GET 또는 관련된 transport binding 내에서 동등한) 응답에서는 유효한 property 가 거짓이며 모든 다른 property 가 의미 없는 값을 가질 수 있다. UPDATE 요청 내에 유효한 property 가 출현하면 값은 참이 되어야 한다 (거짓 값이면 error response 를 초래한다).
- 각 AllJoyn Signal (sessionless, sessioncast, 또는 unicast)은 Observable Resource 상의 독립된 Resource Type 에 매핑된다. 여기서, <seeBelow> 라벨의 값은 AllJoyn Signal 이름이 된다. Resource Type 은 “oic.if.r” Interface 를 지원해야 한다. AllJoyn Signal 의 각 인자는 Resource Type 의 독립된 Property 에 매핑된다. 여기서 해당하는 Property 의 이름은 동일한 Resource 상의 모든 Resource Type 에 걸쳐서 고유함을 갖도록 “<ResourceType>arg<#>”이 접두어로 붙는다 (여기서, <#>은 AllJoyn introspection xml 내 인자의 0 인덱스 위치). 그러므로, AllJoyn argument 이름이 지정되지 않을 때는 property 의 이름은 “<ResourceType>arg<#>”로 된다 (여기서, <#>은 AllJoyn introspection XML 내 인자의 0 인덱스 위치). 뿐만 아니라, 해당하는 Resource Type 은 property 의 나머지가 유효한 값을 갖는지를 나타내는 추가 <ResourceType>validity” property 를 갖는다. 값이 NOTIFY response 의 일부로 전송되면, 유효한 property 가 참이 되고 모든 다른 property 가 유효한 값을 갖는다. RETRIEVE (GET 또는 관련된 transport binding 내에서 동등한) 응답에서는 유효한 property 가 거짓이며 리턴되는 모든 다른

 

property 가 의미 없는 값을 가질 수 있다. 이것은 AllJoyn 에서 신호가 순간적인 이벤트이며 해당하는 메시지의 수명을 넘어서 의미를 가질 필요가 없기 때문이다. AllJoyn 에는 신호를 저장 및 전송할 수 있도록 TTL 필드가 있지만 OCF 1.0 에서는 그러한 지원이 필요 없다는 점에 주의할 필요가 있다. 향후에는 TTL 이 TTL 내의 RETRIEVE 에 대해 유효한 값을 응답하도록 하는데 사용될 수 있으리라 예측된다.

알고리즘적인 매핑 사용 시에는 섹션 7.2 에 따라 AllJoyn data type 이 OCF property type 으로 매핑된다.

AllJoyn 동작이 실패하면 변환기는 OCF client 로 적절한 OCF error response 를 전송해야 한다. AllJoyn error name 이 사용 가능하고 "org.openconnectivity.Error.Code" 접두어를 포함하지 않으면, AllJoyn error name 과 AllJoyn error message (존재하면)로부터 "<error name>: <error message>" 형식을 사용해서 적절한 OCF error message (예: CoAP 를 사용하는 경우 진단 페이로드)를 구성해야 한다. 이 때, <error name>은 AllJoyn error name 필드로부터 취해지고, <error message>는 AllJoyn error message 로부터 취해지고, CoAP 에러 코드는 적절한 값으로 설정된다 (CoAP 사용 시). AllJoyn error name 이 사용 가능하고 "org.openconnectivity.Error.Code" 접두어를 포함하면, OCF error message (예: CoAP 를 사용하는 경우 진단 페이로드)는 AllJoyn error message (존재하면)로부터 취하는 것이 좋다. 이 때, CoAP error code (CoAP 사용 시)는 다음과 같이 도출된 값으로 설정된다: "org.openconnectivity.Error.Code" 접두어를 삭제하고, 그 결과 얻어지는 error name 이 <#>이 소수점 없는 에러 코드인 (예: "404") "<#>" 형식이면 CoAP 에러 코드는 “<#>”에 의해 가리켜지는 error code 이어야 한다. 예를 들어, "org.openconnectivity.Error.Code404"는 "404"로 되어 CoAP transport 에 대해 error 4.04 를 출력한다.

#### 7.1.1.1 Virtual OCF Server 로서의 AllJoyn producer application 의 노출

표 1 은 OCF 1.0 Core 시방서의 표 20 에 규정된 OCF Device property 가 전형적으로 AllJoyn About Interface 시방서 및 AllJoyn Configuration Interface 시방서에 규정된 필드로부터 어떻게 도출되는지를 보여준다.

AllJoyn About 또는 Config data 필드가 (아래의 표 1, 표 2, 표 3, 및 표 4 와 같이) 정의된 매핑 규칙을 가지면, 필드 명칭은 해당 매핑 규칙에 따라 변환되어야 한다. 그렇지 않고 AllJoyn About 또는 Config data 필드가 전체 명칭 (<domain> 접두어 (“com.example”, “org.alljoyn” 등)을 갖는)을 가지면 필드 명칭은 AllJoyn 필드를 매핑하기 위한 섹션 7.1.1 에 규정된 규칙에 따라 변환되어야 한다. 그렇지 않으면, 올바르지 않거나 (에러) 유효한 매핑 (daemonRealm 및 passCode 등)이 없으므로 필드가 변환되어서는 안 된다.

 

1094 **표 1: oic.wk.d resource type 정의**

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| (Device) Name | n | 인간이 인식할 수 있는 명칭<br>예: "Bob'의 Thermostat" | 예 | AppName<br>(정확하게 동등한 것은 존재하지 않는다) | 앱 제조자(개발자 또는 OEM).에 의해 할당된 애플리케이션 명칭. | 예 |
| Spec Version | icv | device가 구현되는 core 시방서의 시방서 버전.<br>구문은 "core.major.minor" | 예 | (없음) | 변환기는 자신의 값을 리턴한다. | |
| Device ID | di | Device 고유의 식별자. 이 값은 DeviceID에 대해 [OCF Security]에서 정의된 것이어야 한다. | 예 | (없음) | OCF Security 시방서에 정의된 대로 사용 | |

 

| **OCF Property title 로** | **OCF Property name** | **OCF 설명** | **OCF 필수?** | **AJ Field name 으로부터** | **AJ 설명** | **AJ 필수?** |
|---|---|---|---|---|---|---|
| Protocol-Independent ID | piid | OCF Device의 고유 식별자 (UUID) | 예 | 존재하는 경우, org.openconnectivity.piid, 그렇지 않은 경우, 인증되면 "피어 GUID" (About 내에 존재하지 않으나 프로토콜에 의해 노출됨), 그렇지 않으면 Hash(DeviceId,AppId). 여기서, Hash 는 Device Id (null 종료를 포함하지 않음) 와 AppId 를 관련 짓고 SHA-1 와. IETF RFC 4122 섹션 4.3 의 알고리즘을 사용해서 이루어진다.<br><br>이것은 RFC 6973 에서 논의되는 프라이버시 문제를 경감시키기 위해 인증 전후에 resource 가 읽혀지면 di 값이 변경될 수 있음을 의미한다. | 피어 GUID: 피어 GUID 는 피어에 대해 유일하게 지속되는 id 이다. 피어 GUID 는 원격 애플리케이션 인스턴스 를 고유하게 식별하기 위해 인증 메커니즘에 의해 사용된다. 원격 피어를 위한 피어 GUID 는 원격 피어가 인증되었을 때만 사용할 수 있다.<br><br>DeviceId: platform-고유 수단에 의해 설정된 Device 식별자<br><br>AppId: 애플리케이션을 위한 128 비트 전역 고유 식별자. AppId 는 IETF RFC 4122 에 규정된 범용 고유 식별자이어야 한다. | 피어 GUID: 조건부 예<br><br>DeviceId: 예<br><br>AppId: 예 |

 

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| Data Model Version | dmv | device data model이 구현되는 vertical 시방서의 시방서 버전. 구문은 콤마로 분리된 "<vertical>.major.minor"] 목록이다. <vertical>은 (즉, Smart Home에 대해 sh) vertical의 명칭이다. | 예 | About 의 Announce signal 의 objectDescription 인자 내에 나열된 각 interface 의 Version property 값의 콤마로 분리된 목록. OCF Core 시방서에 규정된 필수 값에 더불어 추가적인 값들은 "x.<interface name>.<Version property value>" 형식으로 표현된다. | 이 시방서에서는 Version property 의 값이 AllJoyn introspection XML 내의 interface 의 "org.gtk.GDBus.Since" annotation 의 값과 동일한 것으로 간주한다. 따라서, Version property 의 값은 introspection 만을 통해 결정될 수 있다.<br><br>"org.gtk.GDBus.Since" annotation 이 없으면 AllJoyn 은 기본 값을 1 로 지정한다. | 아니오 단, 모든 표준 interface 에 대해 IRB 에 의해 요구되고, 부재는 동의를 암시적으로 의미한다 (예: 0) |
| Localized Descriptions | ld | 하나 이상의 언어로 된 Device의 상세 설명. 이 property는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 device 설명을 포함하는 'value' 필드를 갖는 객체의 배열이다. | 아니오 | 설명 | RFC 5646 의 언어 태그로 표현된 상세 설명. | 예 |
| Software Version | sv | device software의 버전 | 아니오 | SoftwareVersion | 앱의 소프트웨어 버전 | 예 |

 

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| Manufacturer Name | dmn | 하나 이상의 언어로 된 Device 제조자의 명칭. 이 property는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 제조자 명칭을 포함하는 'value' 필드를 갖는 객체의 배열이다. | 아니오 | Manufacturer | 앱의 제조자 명칭 | 예 |
| Model Number | dmno | 제조자에 의해 지정된 모델 번호. | 아니오 | ModelNumber | 앱 모델 번호 | 예 |

뿐만 아니라, AllJoyn About data 내의 모든 추가 제조사 정의 필드는 AllJoyn 필드 명칭에 "x."를 접두어로 붙여서 명명한 property name 을 갖는 OCF Device resource "/oic/d" ("oic.wk.d" resource type 을 구현하는) 내의 제조사 정의 property 에 매핑된다.

표 2 는 OCF 1.0 Core 시방서의 표 15 에 규정된 OCF Device Configuration property 가 어떻게 도출되는지를 보여준다.

**표 2: oic.wk.con resource type 정의**

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| (Device) Name | n | 인간이 인식할 수 있는 명칭 예: "Bob'의 Thermostat" | 예 | AppName (정확하게 동등한 것은 존재하지 않는다) | 앱 제조자(개발자 또는 OEM).에 의해 할당된 애플리케이션 명칭. | 예 |
| Location | loc | 사용 가능한 곳에 위치 정보를 제공한다. | 아니오 | org.openconnectivity.loc (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |

 

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| Location Name | locn | 인간이 인식할 수 있는 위치 명칭<br>예: "Living Room". | 아니오 | org.openconnectivity.locn (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |
| Currency | c | 금전 거래에 사용되는 통화를 가리킨다. | 아니오 | org.openconnectivity.c (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |
| Region | r | device 가 위치한 현재 지역을 나타내는 자유 형식의 텍스트. 자유 형식의 텍스트는 따옴표 (")로 시작할 수 없다. | 아니오 | org.openconnectivity.r (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |

 

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| Localized Names | ln | 하나 이상의 언어로 된 인간이 인식할 수 있는 Device 의 명칭. 이 property 는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 device 명칭을 포함하는 'value' 필드를 갖는 객체의 배열이다. 이 property 와 Device Name (n) property 가 둘 다 지원되면 Device Name (n) 값을 이 배열에 포함해야 한다. | 아니오 | AppName | 앱 제조자(개발자 또는 OEM).에 의해 할당된 애플리케이션 명칭 | 아니오 |
| Default Language | dl | Device 에 의해 지원되는 기본 언어, RFC 5646 언어 태그로 지정. 기본 설정에 의해, property 가 별도로 지정하지 않는 한, client 는 모든 string property 를 이 언어로 취급할 수 있다. | 아니오 | DefaultLanguage | device 에 의해 지원되는 기본 언어. RFC 5646 에 나열된 IETF 언어 태그로 지정. | 예 |

 

뿐만 아니라, AllJoyn Configuration data 내의 모든 추가 제조사 정의 필드는 AllJoyn 필드 명칭에 “x.”를 접두어로 붙여서 명명한 property name 을 갖는 OCF Configuration resource (“oic.wk.con” resource type 및 선택적으로 “oic.wk.con.p” resource type 을 구현하는) 내의 제조사 정의 property 에 매핑된다.

표 3 은 OCF 1.0 Core 시방서의 표 21 에 규정된 OCF Platform property 가 전형적으로 AllJoyn About Interface 시방서 및 AllJoyn Configuration Interface 시방서에 규정된 필드로부터 어떻게 도출되어야 하는지를 보여준다.

 

1109 **표 3: oic.wk.p Resource Type 정의**

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| Platform ID | pi | 물리적 platform의 고유 식별자 (UIUID); IETF RFC 4122에 준거한 UUID. UUID는 RFC 고유의 random generation scheme (버전 4 UUID)을 사용해서 생성하는 것이 바람직하다. | 예 | UUID 이면 DeviceId 이고 그렇지 않으면 DeviceId 값 (null 종료 포함하지 않음)을 사용해서 DeviceId 로부터 명칭 기반의 UUID 를 해시 알고리즘으로 SHA-1 과 함께 IETF RFC 4122 섹션 4.3 에 규정된 알고리즘 내에서 사용될 "name"으로 생성하고, 8f0e4e90-79e5-11e6-bdf4-0800200c9a66 를 name space ID 로 생성한다. | platform-고유 수단 (Linux 및 Android 등)에 의해 설정된 device 의 명칭 | 예 |
| Manufacturer Name | mnmn | 제조자 명칭 (16 문자를 초과하지 않는다) | 예 | Manufacturer (DefaultLanguage 로 16 문자로 잘림) | 앱의 제조자 명칭. | 예 |
| Manufacturer Details Link (URL) | mnml | 제조자 URL (32 문자를 초과하지 않는다) | 아니오 | org.openconnectivity.mnml (if it exists, else property shall be absent) | | 아니오 |
| Model Number | mnmo | 제조자에 의해 지정된 모델 번호 | 아니오 | ModelNumber | 앱 모델 번호. | 예 |
| Date of Manufacture | mndt | device의 제조 날짜 | 아니오 | DateOfManufacture | YYYY-MM-DD 형식 (XML DateTime 형식)을 사용한 제조 날짜. | 아니오 |
| Platform Version | mnpv | platform의 버전 – string (제조자에 의해 정의) | 아니오 | org.openconnectivity.mnpv (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |

 

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| OS Version | mnos | platform 상주 OS의 버전 – string (제조자에 의해 정의) | 아니오 | org.openconnectivity.mnos (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |
| Hardware Version | mnhw | platform 하드웨어의 버전 | 아니오 | HardwareVersion | 앱이 실행되는 device 의 하드웨어 버전 | 아니오 |
| Firmware version | mnfv | device 펌웨어의 버전 | 아니오 | org.openconnectivity.mnfv (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |
| Support URL | mnsl | 제조자로부터의 지원 정보 URL | 아니오 | SupportUrl | 지원 URL (제조자에 의해 기재) | 아니오 |
| SystemTime | st | device에 대한 참조 시간 | 아니오 | org.openconnectivity.st (존재하는 경우, 다른 property 는 존재하지 않아야 한다) | | 아니오 |
| Vendor ID | vid | platform에 대한 제조사 정의 string. string은 자유 형식이며 사용하는 텍스트는 제조사가 결정한다. | 아니오 | DeviceId | platform-고유 수단 (Linux 및 Android 등)에 의해 설정된 device 의 명칭. | 예 |

표 4 는 OCF 1.0 Core 시방서의 표 16 에 규정된 OCF Platform Configuration property 가 어떻게 도출되는지를 보여준다.

 

1113 **표 4: oic.wk.con.p Resource Type 정의**

| OCF Property title 로 | OCF Property name | OCF 설명 | OCF 필수? | AJ Field name 으로부터 | AJ 설명 | AJ 필수? |
|---|---|---|---|---|---|---|
| Platform Names | Mnpn | Platform 식별자 | 아니오 | DeviceName | platform-고유 수단 (Linux 및 Android 등)에 의해 설정된 device 의 명칭 | 사용자에 의해 할당된 Device 명칭. device 명칭은 UI 상에 인식 가능한 명칭으로 표시된다. |

1114 뿐만 아니라, “oic.wk.mnt” properties Factory_Reset (“fr”) 및 Reboot (“rb”)는 각각 AllJoyn
1115 Configuration methods FactoryReset 및 Restart 에 매핑된다.

1116 ### 7.1.2 AllJoyn consumer application 에의 OCF resource 의 노출

1117 별도로 지정되지 않는 한, 각 OCF resource 는 독립된 AllJoyn object 에 매핑된다.

1118 각 OCF Server 는 자신의 About data 를 갖고 독립된 AllJoyn producer application 으로 노출되어야
1119 한다. 이를 통해 변환 전반에 걸쳐 platform-specific, device-specific, 및 resource-specific 필드를
1120 보존할 수 있다. 단, 이것은 producer application 의 AllJoyn Claiming 이 사용자의 상호 작용을
1121 요구하지 않는 식으로 해결되어야 할 필요가 있지만, 이는 구현 상의 문제로 남아 있다.

1122 AllJoyn producer application 은 “oic.d.virtual” AllJoyn interface 를 구현해야 한다. 이렇게 함으로써
1123 변환기는 복수의 변환기가 존재할 때 device 가 이미 변환된 것인지를 판단할 수 있게 된다.
1124 “oic.d.virtual” interface 는 다음과 같이 정의된다.

1125 <interface name="oic.d.virtual"/>

1126 구현은 경로 “/oic/d”에서 AllJoyn object 에 의해 이 interface 를 구현하는 것을 선택할 수 있다.

1127 AllJoyn peer ID 는 OCF device ID (“di”)로 된다.

1128 별도로 지정되지 않는 한, AllJoyn object 경로는 OCF URI 경로로 되어야 한다 (여기서, OCF URI
1129 경로 내의 “-” (하이픈)은 각각 “_h”로 변환되고, OCF URI 경로 내의 “.” (도트)는 각각 “_d”로
1130 변환되고, OCF URI 경로 내의 “~” (물결표)는 각각 “_t”로 변환되고, OCF URI 경로 내의 “_” (밑줄)은
1131 각각 “_u”로 변환된다).

 

AllJoyn About data 는 아래의 표 5 와 같이 채워진다.

변환기 구현은 AllJoyn 측으로부터의 WhoImplements query 를 처리하기 위해 OCF resource 의 cache 를 유지하고, OCF Server 마다 Announce Signal 을 방출하는 것이 바람직하다. 구체적으로, 변환기는 "/oic/res" 변화를 상시 관측하고 Virtual AllJoyn Device 상의 세션을 갖는 client 가 있을 때만 그 밖의 resource 를 관측할 수 있으면 된다.

resource 에는 복수의 유형이 있으며, 각각 다음과 같이 처리되어야 한다.

- Resource Type 이 AllJoyn 과 OCF 양측에 표준 형식이 정의된 (OCF ASA 매핑 또는 아래의 섹션 7.1.2.1 에 정의된) resource type 집합 내에 존재하면,
    a. 변환기는 해당하는 resource type 을 특정해서 변환하기 위한 시방서를 따르거나
    b. 변환기는 Resource Type 을 변환하지 않는다.
- Resource Type 이 제대로 정의된 집합 내에 존재하지 않으면 (그러나 Device Type 이 아니면),
    a. 변환기는 Resource Type 을 변환하지 않거나
    b. 변환기는 OCF Resource Type name 을 AllJoyn Interface name 으로 변환함으로써 섹션 7.2 에 규정된 바와 같이 Resource Type 을 커스텀/제조사 정의 AllJoyn interface 로 알고리즘적으로 매핑한다.

OCF Resource Type 또는 Device Type 은 다음과 같이 AllJoyn interface name 으로 변환되어야 한다.

1) 접두어 "x."가 존재하면 제거한다.
2) 존재하는 각 하이픈마다 (string 내의 왼쪽에서 오른쪽 순으로):
    a. 하이픈 뒤에 문자가 뒤따르면 하이픈과 문자를 해당하는 문자의 대문자로 대체한다 (예를 들어, "-a"를 "A"로 변환한다).
    b. 하이픈 뒤에 문자나 하이픈이 뒤따르는 또 다른 하이픈이 뒤따르면 두 개의 하이픈을 밑줄로 대체한다 (예를 들어, "--a"를 "_a"로 변환하고, "---"를 "_-"로 변환한다).
    c. 그 밖의 하이픈을 밑줄로 대체한다 (예를 들어, "-"를 "_"로 변환한다).

아래의 표에 몇 가지 예를 보인다. 처음 세 개는 일반적인 AllJoyn name 으로 변환된 특이한 OCF name 을 나타낸다. 마지막 세 개는 특이한 AllJoyn name 으로 변환된 일반적인 OCF name 을 나타낸다. ("xn--"는 Internationalized Domain Name 의 Punycode-encoded 형식에 대한 일반적인 도메인 명칭 접두어이므로 일반적인 제조사 고유의 OCF name 으로 나타난다.)

 

| OCF 명칭으로부터 | AllJoyn 명칭으로 |
|---|---|
| x.example.-widget | example.Widget |
| x.example.my--widget | example.my_widget |
| x.example.-my---widget | example.My_Widget |
| x.xn--p1ai.example | xn_p1ai.example |
| x.xn--90ae.example | xn__90ae.example |
| x.example.my-name-1 | example.myName_1 |

OCF Device Type 은 member 를 갖지 않는 AllJoyn interface 로 매핑된다.

별도로 지정되지 않는 한, 각 OCF Resource Type 은 다음과 같이 AllJoyn interface 로 매핑된다.

- 각 OCF property 는 해당하는 interface 내의 AllJoyn property 로 매핑된다 (여기서, OCF property 내의 "." (도트)는 각각 "_d"로 변환되고, OCF property 내의 "-" (하이픈)은 각각 "_h"로 변환된다).
- 각 AllJoyn property 의 EmitsChangedSignal 값은 resource 가 NOTIFY 를 지원하면 "참"으로 설정되고, 그렇지 않으면 "거짓"으로 설정된다. (이 값은 개념 상 현재 OCF 내에서 표현할 수 없으므로 "const" 또는 "invalidates"로 설정되지 않는다.)
- 각 AllJoyn property 의 "access" 속성은 OCF property 가 읽기 전용이면 "read"이고, OCF property 가 읽기-쓰기이면 "readwrite"로 된다.
- resource 가 DELETE 를 지원하면 interface 내에 Delete() 메소드가 나타난다.
- resource 가 CREATE 를 지원하면 생성되는 resource 의 각 property 의 입력 인자와 함께 interface 내에 Create() 메소드가 나타난다. (그러한 정보는 OIC 1.1 에서는 알고리즘적으로 사용할 수 없지만 OCF 1.0 에서는 introspection 을 통해 결정할 수 있다.) 그러한 정보를 사용할 수 없을 때는 입력 인자를 취하지 않는 CreateWithDefaultValues() 메소드가 나타난다. 어떠한 경우에도 출력 인자는 생성되는 resource 의 경로를 포함하는 OBJECT_PATH 가 된다.
- resource 가 UPDATE (즉, "oic.if.rw" 또는 "oic.if,a" interface)를 지원하면 AllJoyn property 설정 동작 (즉, org.freedesktop.DBus.Properties.Set() 메소드 call)이 해당하는 OCF property 를 갖는 Partial UPDATE (예: CoAP 내의 POST)에 매핑된다.
- Resource 가 Resource Type "oic.r.alljoynobject"를 가지면 collection 내의 각 Resource 를 자신의 AllJoyn object 로 개별적으로 변환되지 않고 대신에 collection 내의 모든 Resource 가 객체 경로가 collection 의 OCF URI 경로인 단일 AllJoyn object 로 변환된다.

OCF property type 은 섹션 7.2 에 따라 AllJoyn data type 으로 매핑된다.

 

OCF 동작이 실패하면 변환기는 AllJoyn consumer 에 적절한 AllJoyn error response 를 전송해야 한다. OCF response 내에 error message 가 존재하면 error message (예: CoAP 를 사용하는 경우 진단 페이로드)가 패턴 "<error name>: <error message>"에 들어맞게 된다. 여기서, <error name>은 AllJoyn error name 구문 요구 사항을 준수하고 AllJoyn error name 및 AllJoyn error message 는 error message 로부터 추출되어야 한다. 그렇지 않으면, AllJoyn error name 은 <#>이 OCF 응답에서 소수점 없는 에러 코드(예: CoAP error code)인 (예: "404") "org.openconnectivity.Error.Code<#>""이 되어야 하고, AllJoyn error message 는 OCF 응답 내의 error message 이다.

#### 7.1.2.1 Virtual AllJoyn Producer 로서의 OCF server 의 노출

About interface 에서 리턴되는 객체에 관한 기술은 AllJoyn About Interface 시방서에 규정된 형태로 되어야 하며, 표 5 는 “oic.wk.d”, “oic.wk.con”, “oic.wk.p”, 및 “oic.wk.con.p” 내의 property 를 토대로 AllJoyn About Interface 필드가 어떻게 도출되어야 하는지를 보여준다.

**표 5: AllJoyn About Data fields**

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| AppId | 애플리케이션을 위한 128 비트 전역 고유 식별자. The AppId 는 RFC 4122 에 규정된 범용 고유 식별자이어야 한다. | 예 | Device ID (정확하게 동등한 것은 존재하지 않는다) | di | OCF Device의 고유 식별자 (UUID) | 예 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| DefaultLanguage | Device 에 의해 지원되는 기본 언어, RFC 5646 에 나열된 IETF 언어 태그로 지정. | 예 | Default Language | dl | Device 에 의해 지원되는 기본 언어, RFC 5646 언어 태그로 지정. By default, 기본 설정에 의해, property 가 별도로 지정하지 않는 한, client 는 모든 string property 를 이 언어로 취급할 수 있다.<br><br>부재 시에는 변환기가 상수, 예를 들어, 빈 string 을 리턴한다. | 아니오 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| DeviceName (per supported language) | platform-고유 수단 (Linux 및 Android 등)에 의해 설정된 device 의 명칭 | 아니오 | Platform Names | mnpn | Platform 의 인식 가능한 명칭. 이 property 는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 platform 명칭을 포함하는 'value' 필드를 갖는 객체의 배열이다.<br><br>예:<br>[{"language":"en",<br>"value":"Dave's Laptop"}] | 아니오 |
| DeviceId | platform 고유 수단에 의해 설정된 Device 식별자 | 예 | Platform ID | pi | Platform 식별자 | 예 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| AppName (per supported language) | 앱 제조자(개발자 또는 OEM).에 의해 할당된 애플리케이션 명칭. | 예 | 존재하는 경우, Localized Names, 그렇지 않으면 (Device) Name | ln or n | 하나 이상의 언어로 된 인간이 인식할 수 있는 Device 의 명칭. 이 property 는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 device 명칭을 포함하는 'value' 필드를 갖는 객체의 배열이다. 이 property 와 Device Name (n) property 가 둘 다 지원되면 Device Name (n) 값을 이 배열에 포함해야 한다. | 아니오 (ln), 예 (n) |
| Manufacturer (per supported language) | 앱의 제조자 명칭 | 예 | Manufacturer Name | dmn | 하나 이상의 언어로 된 Device 제조자의 명칭. 이 property는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 제조자 명칭을 포함하는 'value' 필드를 갖는 객체의 배열이다. | 아니오 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| ModelNumber | 앱 모델 번호 | 예 | Model Number | dmno | 제조자에 의해 지정된 모델 번호. | 아니오 |
| SupportedLanguages | 지원되는 언어 목록 | 예 | Localized Name 의 language field | ln | 지원되면 각 배열 요소의 언어 필드 값 목록을 리턴하고, 그렇지 않으면 빈 배열을 리턴한다. | 아니오 |
| Description (per supported language) | RFC 5646 의 언어 태그로 표현된 상세 설명 | 예 | Localized Description | ld | 하나 이상의 언어로 된 Device 의 상세 설명. 이 property 는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 device 설명을 포함하는 'value' 필드를 갖는 객체의 배열이다. | 아니오 |
| DateOfManufacture | YYYY-MM-DD 형식 (XML DateTime 형식)을 사용한 제조 날짜. | 아니오 | 제조 날짜 | mndt | device의 제조 날짜 | 아니오 |
| SoftwareVersion | 앱의 소프트웨어 버전 | 예 | 소프트웨어 버전 | sv | device의 소프트웨어 버전 | 아니오 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| AJSoftwareVersion | 애플리케이션에 의해 사용되는 AllJoyn SDK 의 현잰 버전 | 예 | (없음) | | 변환기는 자신의 값을 리턴해야 한다. | |
| HardwareVersion | 앱이 실행되는 device 의 하드웨어 버전 | 아니오 | 하드웨어 버전 | mnhw | platform 하드웨어의 버전 | 아니오 |
| SupportUrl | 지원 URL (제조자에 의해 기재). | 아니오 | 지원 URL | mnsl | 제조자로부터의 지원 정보 URL | 아니오 |
| org.openconnectivity.mnml | | 아니오 | 제조자 상세 Link (URL) | mnml (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | 제조자 URL (32 문자를 초과하지 않는다) | 아니오 |
| org.openconnectivity.mnpv | | 아니오 | Platform 버전 | mnpv (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | platform의 버전 – string (제조자에 의해 정의) | 아니오 |
| org.openconnectivity.mnos | | 아니오 | OS 버전 | mnos (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | platform 상주 OS의 버전 – string (제조자에 의해 정의) | 아니오 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| org.openconnectivity.mnfv | | 아니오 | 펌웨어 버전 | mnfv (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | device 펌웨어의 버전 | 아니오 |
| org.openconnectivity.st | | 아니오 | SystemTime | st (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | device에 대한 참조 시간 | 아니오 |
| org.openconnectivity.piid | | 아니오 | 프로토콜에 의존하지 않는 ID | piid | 고유하고 불변의 Device 식별자. Client는 Device가 지원하는 모든 프로토콜에 대해 단일 Protocol 독립 ID 값을 사용하는 것을 발견하면 단일 Device가 복수의 통신 프로토콜을 지원하는 것으로 검출할 수 있다. | 예 |

AllJoyn 필드 “org.openconnectivity.piid”는 발표되어야 하지만 로컬라이즈 되어서는 안되며, D-Bus type 서명은 “s”이어야 한다. “org.openconnectivity.”를 접두어로 갖는 표 5 내에 나열된 그 밖의 모든 AllJoyn 필드는 발표되어서도 로컬라이즈 되어서도 안되며, D-Bus type 서명은 “s”이어야 한다.

뿐만 아니라, OCF Device resource “/oic/d” (“oic.wk.d” resource type 을 구현하는) 및 OCF Platform resource “/oic/p” (“oic.wk.p” resource type 을 구현하는) 내의 모든 추가 제조사 정의

 

1206 property 는 property name 으로부터 앞에 있는 “x.”를 제거해서 생성한 field name 을 갖는 AllJoyn
1207 About data 내의 제조사 정의 필드에 매핑된다.

1208 표 6 은 “oic.wk.con” 및 “oic.wk.con.p” 내의 property 를 토대로 AllJoyn Configuration Interface
1209 field 가 어떻게 도출되는지를 보여준다.

1210

 

1211 **표 6: AllJoyn Configuration Data fields**

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| DefaultLanguage | Default device 에 의해 지원되는 기본 언어. | 아니오 | Default Language | dl | Device에 의해 지원되는 기본 언어, RFC 5646 언어 태그로 지정. 기본 설정에 의해, property가 별도로 지정하지 않는 한, client는 모든 string property를 이 언어로 취급할 수 있다. | 아니오 |
| DeviceName | 사용자에 의해 할당된 Device 명칭. device 명칭은 UI 상에 인식 가능한 명칭으로 표시된다.. | 아니오 | PlatformNames | mnpn | Platform 의 인식 가능한 명칭. 이 property 는 각 객체가 'language' 필드 (RFC 5646 언어 태그 포함)와 지시된 언어로 platform 명칭을 포함하는 'value' 필드를 갖는 객체의 배열이다.<br><br>예:<br>[{"language":"en", "value":"Dave's Laptop"}] | 아니오 |
| org.openconnectivity.loc | | 아니오 | Location | loc (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | 사용 가능한 곳에 위치 정보를 제공한다. | 아니오 |

 

| AJ Field name 으로 | AJ 설명 | AJ 필수? | OCF Property title 로부터 | OCF Property name | OCF 설명 | OCF 필수? |
|---|---|---|---|---|---|---|
| org.openconnectivity.locn | | 아니오 | Location Name | locn (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | 인간이 인식할 수 있는 위치 명칭<br>예: "Living Room". | 아니오 |
| org.openconnectivity.c | | 아니오 | Currency | c (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | 금전 거래에 사용되는 통화를 가리킨다. | 아니오 |
| org.openconnectivity.r | | 아니오 | Region | r (존재하는 경우, 다른 필드는 존재하지 않아야 한다) | device 가 위치한 현재 지역을 나타내는 자유 형식의 텍스트. 자유 형식의 텍스트는 따옴표 (")로 시작할 수 없다. | 아니오 |

AllJoyn 필드 "org.openconnectivity.loc"는 발표되어서도 로컬라이즈 되어서도 안되며, D-Bus type 서명은 "ad"이어야 한다. "org.openconnectivity."를 접두어로 갖는 표 5 내에 나열된 그 밖의 모든 AllJoyn 필드는 발표되어서도 로컬라이즈 되어서도 안되며, D-Bus type 서명은 "s"이어야 한다.

뿐만 아니라, Configuration methods FactoryReset 및 Restart 는 각각 "oic.wk.mnt" properties Factory_Reset ("fr") 및 Reboot ("rb")에 매핑되고, OCF Configuration resource ("oic.wk.con" resource type 및 선택적으로 "oic.wk.con.p" resource type 을 구현하는) 내의 추가 제조사 정의 property 는 property name 으로부터 앞에 있는 "x."를 제거함으로써 명명한 field name 을 갖는 AllJoyn Configuration data 의 제조사 정의 필드에 매핑된다.

### 7.2 D-Bus 와 OCF payload 로부터의 On-the-Fly 변환

"dbus1" 페이로드 형식은 D-Bus 시방서에 규정되고 AllJoyn 은 D-Bus 프로토콜을 채택해서 네트워크를 통해 배포되도록 하였다. 형식에 대한 AllJoyn 의 변형은 패킷의 헤더 부분에 행해지고 data 페이로드 자체에는 행해지지 않아서 "dbus1"에 그대로 호환된다. Linux community 에 의해

 

제안된 프로토콜의 변형 (“GVariant” 및 “kdbus” 페이로드)은 약간의 비호환성을 포함하고 있으므로 여기서의 논의와는 관계가 없다.

### 7.2.1 introspection 의 지원 없는 변환

이 섹션에서는 실제 device 로부터의 introspection metadata 가 없는 경우 변환기가 두 개의 페이로드 형식 간의 메시지 변환을 어떻게 하는지를 설명한다. 이러한 상황은 다음과 같은 경우에 발생한다.

- OIC 1.1 device 에게 요청
- OIC 1.1 device 로부터의 응답
- “D-Bus VARIANT” 형식의 AllJoyn property 의 내부 페이로드와 같이 introspection 에 의해 기술되지 않는 내용

introspection 을 사용할 수 없으므로 변환기는 rich JSON sub-type 을 알 수 없고 근본적인 CBOR type 으로부터만 JSON generic type 을 추론한다. 따라서, 변환은 아래와 같이 그러한 generic type 면에서 지정된다.

#### 7.2.1.1 Boolean

Boolean 변환은 양쪽에서 이 유형을 지원하므로 일반적이다.

| D-Bus 형식 | JSON 형식 |
|---|---|
| “b” – BOOLEAN | boolean (참 또는 거짓) |

#### 7.2.1.2 Numeric type

숫자 형식의 변환은 정보 손실이 있으며 이는 JSON generic type 의 표현 상의 제한으로 인해 피할 수 없다. 이것은 introspection 을 통해서만 해결 가능하다.

숫자 형식의 변환은 방향성을 갖는다.

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| “y” - BYTE (unsigned 8-bit) | number |
| “n” - UINT16 (unsigned 16-bit) | |
| “u” - UINT32 (unsigned 32-bit) | |
| “t” - UINT64 (unsigned 64-bit)(1) | |
| “q” - INT16 (signed 16-bit) | |
| “”” - INT32 (signed 32-bit) | |

 

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| "x" - INT64 (signed 64-bit)(1) | |
| "d" - DOUBLE (IEEE 754 double precision) | |

1243

| JSON 형식으로부터 | D-Bus 형식으로 |
|---|---|
| number | "d" - DOUBLE(2) |

1244 주석 및 근거:

1245 1. "t" (UINT64)와 "x" (INT64) 유형의 D-Bus payload IEEE 754 배정밀도 부동 소수점으로 완벽하게
1246 표현할 수 없는 값을 포함한다. JSON 을 관장하는 RFC 는 그러한 숫자를 금지하지는 않지만 많은
1247 구현이 제대로 처리할 수 없는 경우가 있다고 주의를 주고 있다. OCF 는 JSON 대신에 CBOR 을
1248 사용해서 페이로드를 전송하여 그러한 숫자를 충실하게 표현할 수 있다. 단, OCF 1.0 Core
1249 시방서은 $-2^{53} \leq x \leq 2^{53}$ 범위를 벗어나는 정수를 허용하지 않는다.

1250 2. 가장 예측 가능한 결과를 제공하기 위해 OCF 로부터 AllJoyn 으로의 모든 변환은 "d" DOUBLE
1251 (IEEE 754 배정밀도) 유형의 값을 생성한다..

1252 #### 7.2.1.3 문자 string

| D-Bus 형식 | JSON 형식 |
|---|---|
| "s" - STRING | string |

1253 D-Bus 와 JSON string 간의 변환은 둘 다 컨텐츠가 유효한 Unicode 이어야 함을 요구하므로
1254 간단하게 수행할 수 있다. 예를 들어, 양쪽의 프로토콜이 데이터 인코딩으로 UTF-8 을 지정하므로
1255 direct byte copy 를 수행할 수 있으며 주어진 정규화 형식에 데이터를 제약하지도 않을 뿐더러 사적인
1256 문자 또는 비 문자를 불허해야 하는지를 지정하지도 않는다.

1257 D-Bus string 의 길이는 상시 알려진 값이므로 변환기는 CBOR indeterminate text string (first byte
1258 0x7f)을 사용하지 않는 것이 바람직하다.

1259 #### 7.2.1.4 Byte array

1260 바이트 어레이의 변환은 방향성을 갖는다.

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| "ay" - ARRAY of BYTE | (base64-encoded) string |

1261 base64url 인코딩은 IETF RFC 4648 섹션 5 에 규정되어 있다.

 

#### 7.2.1.5 D-Bus Variant

| D-Bus 형식 | JSON 형식 |
|---|---|
| "v" - VARIANT | *아래 참조* |

D-Bus 에는 다른 D-Bus type 을 둘러싸는 VARIANT ("v")라고 불리는 유형이 있다. 이것은 유형 시스템이 유형을 삭제하기 위한 하나의 방법이다. 반면에, JSON 에서는 유형이 보장되지 않으며, 이는 기술적으로 모든 JSON 값이 variant 임을 의미한다. D-Bus variant 의 JSON 으로의 변환은 해당하는 variant 를 입력하고 이 표준에 기재된 규칙에 따라 내부의 유형을 인코딩 함으로써 이루어진다.

D-Bus variant 는 variant 자체를 포함할 수 있으므로 알고리즘은 회귀적이어야 한다.

#### 7.2.1.6 D-Bus Object 경로 및 서명

D-Bus object 경로 및 서명의 변환은 단방향성을 갖는다 (이들로의 매핑은 없고 이들로부터의 매핑만 있을 뿐이다). 반대 방향으로는, "s"가 가장 일반적으로 사용되는 string type 으로 간주되므로, 섹션 7.2.1.3 는 항상 OBJECT_PATH 또는 SIGNATURE 가 아닌 D-Bus STRING 으로 변환한다.

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| "o" - OBJECT_PATH | string |
| "g" – SIGNATURE | |

D-Bus object 경로와 D-Bus type 서명은 둘 다 D-Bus 시방서에서 볼 수 있는 특정 형식을 갖는 US-ASCII string 이다. 이들은 대단히 드물게 사용되며 introspection 의 지원 없이 변환되는 resource 내에서는 보기 힘들다.

#### 7.2.1.7 D-Bus 구조

다음 유형의 변환은 방향성을 갖는다.

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| "r" - STRUCT | array, length > 0 |

D-Bus 구조는 각 요소에 대해 소정의 유형 목록을 포함하는 고정 길이의 배열로 해석할 수 있다. 이러한 구조는 D-Bus 구조의 정확한 요소인 이종 컨텐츠의 배열로서 구조에 나타나는 순서로 JSON 에 매핑된다.

#### 7.2.1.8 배열

다음과 같은 유형의 변환은 양방향성을 갖는다.

 

| D-Bus 형식 | JSON 형식 |
|---|---|
| "ay" - ARRAY of BYTE | (base64-encoded) string – 섹션 7.2.1.4 참조 |
| "ae" - ARRAY of DICT_ENTRY | object – 섹션 7.2.1.9 참조 |

1284 다음과 같은 유형의 변환은 방향성을 갖는다.

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| "a" – ARRAY of anything else not specified above | array |

1285

| JSON 형식으로부터 | 조건 | D-Bus 형식으로 |
|---|---|---|
| array | length=0 | "av" – ARRAY of VARIANT |
| array | length>0, all elements of same type | "a" – ARRAY |
| array | length>0, elements of different types | "r" – STRUCT |

1286 각각 JSON string 과 객체에 매핑되는 byte 의 배열 및 사전 항목의 배열 외의 JSON 내의 배열은
1287 단일 유형 에 제한되지 않는다 (즉, 이종 배열). 이러한 이유로, 엄밀하게 말하면 byte 의 배열 및 사전
1288 항목 이외의 모든 D-Bus array 는 먼저 "av"의 Variant 형 배열로 변환한 다음에 JSON 으로 변환할 수
1289 있다.

1290 variant 형 D-Bus array 의 변환은 위에 기술한 variant 형 변환을 사용하며, 이는 주어진 값을
1291 포함하는 variant 형과 Variant 형 이외에 해당하는 값 간의 차이를 간단하게 제거한다. 다시 말하면,
1292 D-Bus array 의 요소는 이 표준의 기타 규칙에 따라 JSON array 의 요소로 추출되어 전송된다.

1293 **7.2.1.9 사전 / 객체**

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

1294 "dictionary of STRING to VARIANT"는 페이로드 내에서 볼 수 있는 가장 일반적인 유형의
1295 dictionary 이고 D-Bus 내의 모든 가능한 dictionary 의 거의 완벽한 확대집합 이기에 선택된다. 더욱이,
1296 이는 OCF 가 데이터 모델 내에서 사용하는 표현인 JSON Object 를 충실하게 표현할 수 있으며,
1297 따라서 그러한 D-Bus dictionary 는 현재 사용되는 모든 OCF JSON Object 를 충실하게 전달할 수
1298 있음을 의미한다.

1299 String 형에서 Variant 형으로 매핑하지 않는 D-Bus dictionary 는 먼저 그러한 제약으로 변환된
1300 다음에 CBOR 로 인코딩된다.

 

1301 #### 7.2.1.10 변환 불가 형식

| D-Bus 형식 |
| --- |
| "h" – UNIX_FD (Unix file descriptor) |

| JSON 형식 |
| --- |
| null |
| undefined (not officially valid JSON, but some implementations permit it) |

1302 위의 유형은 변환할 수 없으며 변환기는 착신 메시지를 폐기해야 한다. 위의 유형 중 어느 것도 현재
1303 AllJoyn, OIC 1.1 또는 향후의 OCF 1.0 device 에 의해 사용되지 않으므로 이러한 유형을 변환할 수
1304 없는 것은 문제가 되지 않는다.

1305 #### 7.2.1.11 예

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

 

| | |
|---|---|
| SIGNATURE("s") | "s" |
| VARIANT(INT32(0)) | 0 |
| VARIANT(VARIANT(INT32(0))) | 0 |
| VARIANT(STRING("Hello")) | "Hello" |

1306

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

1307

 

비고:

1. 이 값은 정보의 손실 없이 IEEE754 배정밀도 부동 소수점으로 표현할 수 없다. 이는 또한 OCF 내에 현재 허용되는 범위에서 벗어난다.

### 7.2.2 introspection 지원을 통한 변환

introspection 이 사용 가능하면, 변환기는 상대방에 양질의 응답을 노출시키는 서비스를 제공하는 쪽에서 제공하는 여분의 메타데이터를 사용할 수 있다. 이 장에서는 메타데이터가 있을 때 앞에서 설명한 변환을 어떻게 변형하는지를 상세하게 설명한다.

Introspection metadata 는 서비스에 대한 요청 및 해당하는 서비스로부터의 응답을 둘 다 변환하는데 사용할 수 있다. 요청의 변환에 사용될 때는 변환기가 서비스가 예측하는 것에 정확하게 따라야 하므로 introspection 은 “제한적”이 된다. 응답의 변환에 사용될 때는 introspection 이 “이완적”이지만 수신자에게 향후 어떠한 가능한 값이 출현할지를 알려주는데 사용할 수 있다.

OCF introspection 은 JSON 유형, media 속성, 및 format 속성을 사용하고 CBOR 인코딩을 사용하지 않음에 주의하기 바란다. 각 JSON 유형의 실제 인코딩은 OCF 1.0 Core 시방서의 섹션 12.3 에서 다뤄지고, JSON format 속성 값은 JSON Schema Validation 에 정의되며, JSON media 속성 값은 JSON Hyper-Schema 에 정의된다.

#### 7.2.2.1 introspection 자체의 변환

OCF 1.0 및 AllJoyn 은 둘 다 노출된 모든 서비스가 introspection metadata 를 포함할 것을 요구하며, 이는 변환기가 발견하는 각각의 OCF resource 도는 AllJoyn producer 에 대해 introspection 정보를 on-the-fly 변환해야 함을 의미한다. 변환기는 변환된 형식으로 표현 가능한 만큼의 원래 정보를 보존해야 한다. 이는 설명 및 문서 텍스트와 같이 기계 상호 작용에 사용되는 정보와 사용자 상호 작용에 사용되는 정보를 둘 다 포함한다.

#### 7.2.2.2 introspection 데이터의 유효성

Introspection data 는 상수가 아니므로 변환기는 향후의 서비스 검색 시에 이전에 마주친 D-Bus interface 또는 OCF Resource Type 이 이전과는 다른 것을 알게 되는 경우가 있다. 변환기는 introspection 의 변화에 대해 destination 측이 어떻게 반응하는지에 관심을 가질 필요가 있다.

AllJoyn 서비스에 의해 사용되는 D-Bus interface 는 새로운 버전으로 업데이트될 수 있으며, 이는 주어진 유형의 서비스가 동일한 interface 의 다른 두 개의 버전에 의해 제공될 수 있음을 의미한다. 표준 interface 에 대한 업데이트는 AllSeen Interface Review Board 에 의해 작성된 엄격한 가이드라인을 따라야 하며, 각 버전의 다른 OCF Resource Type 에의 매핑은 큰 어려움 없이 가능해야 한다. 단, 제조사-specific 확장이 이러한 요구 사항을 따른다는 보장은 없다. 사실 상, 두 가지 버전의 제품이 동일한 명칭과 버전 번호를 갖지만 완전하게 호환되지 않는 interface 를 포함하는 것을 방지할 수 있는 방법은 없다.

 

반면에 규칙은 상당히 느슨하다. OCF 는 Resource Type 에 대해 옵션 property 를 규정하므로, AllJoyn consumer application 이 기대하는 것과 같이 버전 번호를 단순하게 증가시킬 수는 없다.

그러나, 변환기가 “on-the-fly” 변환에 의해 생성한 서비스는 일반적인 client 애플리케이션에 의해서만 액세스 됨에 주의해야 한다. 전용 애플리케이션은 “딥 바인딩” 변환만 사용하게 된다.

#### 7.2.2.3 숫자 형식

숫자의 경우, 모든 D-Bus 및 JSON numeric type 은 소스와 동등하게 취급되면 모두 상대방의 임의의 유형으로 변환될 수 있다. 서비스에 대한 요청의 변환 시에, 변환기는 source 에서 destination 으로 변환할 때 정보의 손실이 발생했는지만 검증하면 된다. 예를 들어, 숫자 1.5 를 JSON integer 또는 D-Bus integral type 중 하나로 변환할 때 정보의 손실이 발생하며, 이 경우 변환기는 착신 메시지를 거부하게 된다. 마찬가지로, 값 1,234,567 은 D-Bus byte, 16-bit signed 또는 unsigned integer 의 범위 내에 포함되지 않는다.

서비스로부터의 응답 변환 시에는, 변환기는 다음과 같은 규칙을 사용해야 한다.

아래의 표는 JSON type 으로부터 대응하는 D-Bus type 으로 변환하는 방법을 보여주며, 첫 번째 매칭되는 행이 사용된다. JSON schema JSON integer 의 최소값을 가리키지 않으면, 0 이 default 로 설정된다. JSON schema 가 JSON integer 의 최대값을 가리키지 않으면, $2^{32} - 1$ 이 default 로 설정된다. 결과적으로 얻어지는 AllJoyn introspection XML 은 JSON 값의 최소값 또는 최대값이 the natural minimum or maximum of the D-Bus type 의 자연 최소값 또는 최대값과 다를 때마다 각각 “org.alljoyn.Bus.Type.Min” 및 “org.alljoyn.Bus.Type.Max” annotation 을 포함하게 된다.

| JSON 형식으로부터 | 조건 | D-Bus 형식으로 |
|---|---|---|
| integer | minimum ≥ 0 AND maximum < $2^8$ | “y” (BYTE) |
| | minimum ≥ 0 AND maximum < $2^{16}$ | “q” (UINT16) |
| | minimum ≥ $-2^{15}$ AND maximum < $2^{15}$ | “n” (INT16) |
| | minimum ≥ 0 AND maximum < $2^{32}$ | “u” (UINT32) |
| | minimum ≥ $-2^{31}$ AND maximum < $2^{31}$ | “i” (INT32) |
| | minimum ≥ 0 | “t” (UINT64) |
| | | “x” (INT64) |
| number | | “d” (DOUBLE) |
| string | pattern = “^0\|([1-9][0-9]{0,19})$” | “t” (UINT64) |
| | pattern = “^0\|(-?[1-9][0-9]{0,18})$” | “x” (INT64) |

 

1360 아래의 표는 D-Bus type 으로부터 대응하는 JSON type 으로 변환하는 방법을 보여준다.

| D-Bus 형식으로부터 | JSON 형식으로 | 비고 |
|---|---|---|
| "y" (BYTE) | integer | JSON schema 의 "minimum" 및 "maximum"은, 존재하는 경우, 각각 "org.alljoyn.Bus.Type.Min" 및 "org.alljoyn.Bus.Type.Max" annotation 으로 설정되거나, 그러한 annotation 이 존재하지 않으면, or to the min and max values of the D-Bus type 범위의 최소값 및 최대값으로 설정된다. |
| "n" (UINT16) | | |
| "q" (INT16) | | |
| "u" (UINT32) | | |
| "i" (INT32) | | |
| "t" (UINT64) | org.alljoyn.Bus.Type.Max ≤ $2^{53}$ 의 경우 integer, 그렇지 않으면 JSON pattern 속성 "^0\|([1-9][0-9]{0,19})$"를 갖는 string | IETF RFC 7159 섹션 6 은 더 큰 JSON integer 는 상호 운용 가능하지 않음을 기술한다. |
| "x" (INT64) | (org.alljoyn.Bus.Type.Min ≥ -$2^{53}$ AND org.alljoyn.Bus.Type.Max ≤ $2^{53}$)의 경우 integer, 그렇지 않으면 JSON format 속성 "^0\|(-?[1-9][0-9]{0,18})$"를 갖는 string | IETF RFC 7159 섹션 6 은 그 밖의 JSON integer 는 상호 운용 가능하지 않음을 기술한다. |
| "d" (double) | number | |

1361 #### 7.2.2.4 문자 string 및 바이트 어레이

| D-Bus 형식 | JSON 형식 | JSON 매체 속성, binaryEncoding property |
|---|---|---|
| "s" – STRING | string | (none) |
| "ay" - ARRAY of BYTE | string | base64 |

1362 이전 섹션과 비교해서 text string 과 byte 배열의 변환에는 차이가 없다. 이 섹션에서는 발생된 OCF
1363 introspection 에 대한 JSON 등가 유형을 단순히 나열한다.

1364 또한, 다음의 JSON Type 의 매핑은 방향성을 갖는다.

| JSON 형식으로부터 | 조건 | D-Bus 형식으로 |
|---|---|---|
| string | pattern = "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$" | "ay" – ARRAY of BYTE |

1365 위의 표에 없는 다른 format 값 (예: date-time, uri 등) 또는 pattern 값을 갖는 JSON string 은 단순히
1366 값을 D-Bus string 으로 매핑함으로써 해당하는 format 이나 pattern 속성이 없는 것과 동일하게
1367 취급된다.

 

#### 7.2.2.5 D-Bus Variant

| D-Bus 형식 | JSON 형식 |
|---|---|
| "v" – VARIANT | *아래 참조* |

AllJoyn producer 의 introspection 이 요청 내의 값이 D-Bus VARIANT 이어야 함을 가리키면, 변환기는 이 표준의 나머지에 기술된 규칙에 따라 그러한 variant 를 생성하고 착신 값을 변수의 페이로드로 인코딩해야 한다.

#### 7.2.2.6 D-Bus Object 경로 및 서명

| D-Bus 형식으로부터 | JSON 형식으로 |
|---|---|
| "o" – OBJECT_PATH | string |
| "g" – SIGNATURE | |

AllJoyn producer 의 introspection 이 요청 내의 값이 D-Bus Object Path 또는 D-Bus Signature 이어야 함을 가리키면, 변환기는 착신 CBOR Text String 의 유효성 체크를 수행해야 한다. 착신 데이터가 이 체크를 통과하지 못하면 메시지를 거절해야 한다.

#### 7.2.2.7 D-Bus 구조

D-Bus 구조 member 는 "org.alljoyn.Bus.Struct.*StructureName.*Field.*fieldName.*Type" 주석으로 introspection XML 내에 기술된다. 변환기는 다음과 같이 Bridged AllJoyn 프로듀서로부터 취득한 About data 의 AJSoftwareVersion 필드를 사용한다. Bridged Device 에 구현된 AllJoyn 의 버전이 v16.10.00 이상이고 member 주석이 존재하면, 변환기는 각 member 를 해당하는 명칭의 entry 에 매핑하여 JSON object 를 사용해서 구조를 표현한다. 변환기는 D-Bus 구조와 비교해서 착신 CBOR payload 가 필드 순서를 변경했을 가능성이 있음을 알아야 한다. Bridged Device 에 구현된 AllJoyn 의 버전이 v16.10.00 미만이면, 변환기는 introspection data 의 지원 없이 D-Bus 구조를 변환하기 위한 규칙을 따라야 한다.

#### 7.2.2.8 배열 및 사전

AllJoyn interface 의 introspection 이 array 가 BYTE ARRAY ("ay") 또는 VARIANT ARRAY ("av")가 아니거나 사전이 STRING 을 VARIANT ("a{sv}")로 매핑하지 않는 것을 가리키면, 변환기는 다른 섹션에 규정된 제한적인 또는 느슨한 규칙을 적용해야 한다.

마찬가지로, OCF introspection 이 동종 array type 을 가리키면, 배열의 element type 에 관한 정보를 VARIANT ("v") 대신에 D-Bus array type 으로 사용해야 한다.

#### 7.2.2.9 그 밖의 JSON 형식 속성 값

JSON format 속성은 그 밖의 커스텀 속성 유형을 포함할 수 있다. 현 시점에서는 알려지지 않고 있으나, 이러한 유형들은 이들의 type 및 representation 단독에 의해 처리될 것으로 예측된다.

 

1394 **7.2.2.10 예**

| AllJoyn Source | AllJoyn Introspection Notes | Translated JSON Payload | OCF Introspection Notes |
|---|---|---|---|
| UINT32 (0) | | 0 | JSON schema 는 다음을 가리켜야 한다.<br>“type”: “integer”,<br>“minimum”: 0,<br>“maximum”: 4294967295 |
| INT64 (0) | | 0 | AllJoyn 에 Min/Max annotation 이 존재하지 않으므로, JSON schema 는 다음을 가리켜야 한다.<br>“type”: “string”,<br>“pattern”: “^0\|(-?[1-9][0-9]{0,18})$” |
| UINT64 (0) | | ”0” | AllJoyn 에 Max annotation 이 존재하지 않으므로, JSON schema 는 다음을 가리켜야 한다.<br>“type”: “string”,<br>“pattern”: ^0\|([1-9][0-9]{0,19})$” |
| STRING(“Hello”) | | ”Hello” | JSON schema 는 다음을 가리켜야 한다.<br>“type”: “string” |
| OBJECT_PATH(“/”) | | ”/” | JSON schema 는 다음을 가리켜야 한다.<br>“type”: “string” |
| SIGNATURE(“g”) | | ”g” | JSON schema 는 다음을 가리켜야 한다.<br>“type”: “string” |
| ARRAY<BYTE>(0x48, 0x65, 0x6c, 0x6c, 0x6f) | | ”SGVsbG8” | JSON schema 는 다음을 가리켜야 한다.<br>“type”: “string”<br>“media binaryEncoding”: “base64” |
| VARIANT(*anything)* | | *?* | JSON schema 는 다음을 가리켜야 한다.<br>“type”: [ "boolean", "object", "array", "number", "string", "integer" ] |
| ARRAY<INT32>() | | [] | JSON schema 는 다음을 가리켜야 한다.<br>“type”: “array”,<br>"items": { "type": "integer" } |

 

| AllJoyn Source | AllJoyn Introspection Notes | Translated JSON Payload | OCF Introspection Notes |
|---|---|---|---|
| ARRAY<INT64>() | | [] | JSON schema 는 다음을 가리켜야 한다.<br>"type": "array",<br>"items": { "type": "string",<br>"pattern": "^0\|([1-9][0-9]{0,18})$" } |
| STRUCT< INT32, INT32>(<br>0, 1<br>) | AllJoyn introspection 은 다음과 같은 annotation 으로 인자를 규정한다.<br><struct name="Point"><br><field name="x" type="i"/><br><field name="y" type="i"/><br></struct> | [“x”: 0, “y”: 1] | JSON schema 는 다음을 가리켜야 한다.<br>"type": "object", "properties": { "x": { "type": "integer" }, "y": { "type": "integer" } } |

1395

| CBOR Payload | OCF Introspection Notes | Translated AllJoyn | AllJoyn Introspection Notes |
|---|---|---|---|
| 0 | "type": "integer" | INT32(0) | |
| 0 | "type": "integer, "minimum": -$2^{40}$, "maximum": $2^{40}$ | INT64(0) | org.alljoyn.Bus.Type.Min = -$2^{40}$<br>org.alljoyn.Bus.Type.Max = $2^{40}$ |
| 0 | "type": "integer", "minimum": 0, "maximum": $2^{48}$ | UINT64(0) | org.alljoyn.Bus.Type.Max = $2^{48}$ |
| 0.0 | “type”: “number” | DOUBLE(0.0) | |
| [1] | JSON schema indicates:<br>"type": "array",<br>"items": { "type": "integer",<br>"minimum": 0"maximum": $2^{46}$ } | ARRAY<UINT64>(1) | org.alljoyn.Bus.Type.Max = $2^{46}$ |

1396

1397

 

# 8 Device Type 정의

요구되는 Resource Type 을 아래의 표에 나열한다.

| Device Name (단순 정보) | Device Type ("rt") (규범) | 필수 Resource name | 필수 Resource Type |
|---|---|---|---|
| Bridge | oic.d.bridge | Secure Mode | oic.r.securemode |
| Virtual Device | oic.d.virtual | Device | oic.wk.d |

# 9 Resource Type 정의

## 9.1 resource type 목록

**표 7 resource type 의 알파벳순 목록**

| 명칭 (단순 정보) | Resource Type (rt) | 섹션 |
|---|---|---|
| Secure Mode | oic.r.securemode | 9.2 |
| AllJoyn Object | oic.r.alljoynobject | 9.3 |

## 9.2 보안 모드

### 9.2.1 개요

이 resource 는 보안 모드 on/off 기능 (on/off)을 기술한다.

secureMode 값이 true 이면 기능이 on 되었으며, 다음을 의미한다.

- 안전하게 통신할 수 없는 Bridged Server 는 대응하는 Virtual OCF Server 를 가질 수 없으며,
- 안전하게 통신할 수 없는 Bridged Client 는 대응하는 Virtual OCF Client 를 가질 수 없다.

"secureMode 값이 false 이면 기능이 off 되었으며, 다음을 의미한다.

- 모든 Bridged Server 가 대응하는 Virtual OCF Server 를 가질 수 있으며,
- 모든 Bridged Client 가 대응하는 Virtual OCF Client 를 가질 수 있다.

 

### 9.2.2 URI 경로 예

/example/SecureModeResURI

### 9.2.3 Resource Type

resource type (rt)은 oic.r.securemode 로 정의된다.

### 9.2.4 RAML 정의

```
#%RAML 0.8
title: OCFSecureMode
version: v1.0.0-20170531
traits:
 - interface:
     queryParameters:
       if:
         enum: ["oic.if.rw", "oic.if.baseline"]

/example/SecureModeResURI:
  description: |
    This resource describes a secure mode on/off feature (on/off).
    A secureMode value of “true” means that the feature is on, and any Bridged Server that cannot
be communicated with securely shall not have a corresponding Virtual OCF Server, and any Bridged
Client that cannot be communicated with securely shall not have a corresponding Virtual OCF Client.
    A secureMode value of “false” means that the feature is off, any Bridged Server can have a
corresponding Virtual OCF Server, and any Bridged Client can have a corresponding Virtual OCF
Client.

  is: ['interface']

  get:
    description: |
      Retrieves the value of secureMode.
    responses:
      200:
        body:
          application/json:
            schema: |
              {
                "id": "https://www.openconnectivity.org/ocf-
apis/bridging/schemas/oic.r.securemode.json#",
                "$schema": "http://json-schema.org/draft-04/schema#",
                "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
reserved.",
                "title": "Secure Mode",
                "definitions": {
                  "oic.r.securemode": {
                    "type": "object",
                    "properties": {
                      "secureMode": {
                        "type": "boolean",
                        "description": "Status of the Secure Mode"
                      }
                    }
                  }
                },
                "type": "object",
                "allOf": [
                  {"$ref": "../../core/schemas/oic.core-schema.json#/definitions/oic.core"},
```

 

```
                  {"$ref": "#/definitions/oic.r.securemode"}
                ],
                "required": [ "secureMode" ]
              }
            example: |
              {
                "rt":          ["oic.r.securemode"],
                "id":          "unique_example_id",
                "secureMode":  false
              }

  post:
    description: |
      Updates the value of secureMode.
    body:
      application/json:
        schema: |
          {
            "id": "https://www.openconnectivity.org/ocf-
apis/bridging/schemas/oic.r.securemode.json#",
            "$schema": "http://json-schema.org/draft-04/schema#",
            "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
reserved.",
            "title": "Secure Mode",
            "definitions": {
              "oic.r.securemode": {
                "type": "object",
                "properties": {
                  "secureMode": {
                    "type": "boolean",
                    "description": "Status of the Secure Mode"
                  }
                }
              }
            },
            "type": "object",
            "allOf": [
              {"$ref": "../../core/schemas/oic.core-schema.json#/definitions/oic.core"},
              {"$ref": "#/definitions/oic.r.securemode"}
            ],
            "required": [ "secureMode" ]
          }
        example: |
          {
            "id":          "unique_example_id",
            "secureMode":  true
          }
    responses:
      200:
        body:
          application/json:
            schema: |
              {
                "id": "https://www.openconnectivity.org/ocf-
apis/bridging/schemas/oic.r.securemode.json#",
                "$schema": "http://json-schema.org/draft-04/schema#",
                "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
reserved.",
                "title": "Secure Mode",
                "definitions": {
                  "oic.r.securemode": {
                    "type": "object",
                    "properties": {
```

```
                    "secureMode": {
                      "type": "boolean",
                      "description": "Status of the Secure Mode"
                    }
                  }
                }
              },
              "type": "object",
              "allOf": [
                {"$ref": "../../core/schemas/oic.core-schema.json#/definitions/oic.core"},
                {"$ref": "#/definitions/oic.r.securemode"}
              ],
              "required": [ "secureMode" ]
            }
          example: |
            {
              "id":          "unique_example_id",
              "secureMode":  true
            }
```

### 9.2.5 Swagger2.0 정의

```
{
  "swagger": "2.0",
  "info": {
    "title": "OCFSecureMode",
    "version": "v1.0.0-20170531",
    "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/SecureModeResURI" : {
      "get": {
        "description": "This resource describes a secure mode on/off feature (on/off).\nA
secureMode value of 'true' means that the feature is on, and any Bridged Server that cannot be
communicated with securely shall not have a corresponding Virtual OCF Server, and any Bridged
Client that cannot be communicated with securely shall not have a corresponding Virtual OCF
Client.\nA secureMode value of 'false' means that the feature is off, any Bridged Server can have a
corresponding Virtual OCF Server, and any Bridged Client can have a corresponding Virtual OCF
Client.\nRetrieves the value of secureMode.\n",
        "parameters": [
          {"$ref": "#/parameters/interface"}
        ],
        "responses": {
            "200": {
              "description" : "",
              "x-example":
                {
                  "rt":          ["oic.r.securemode"],
```

```
                  "id":          "unique_example_id",
                  "secureMode":  false
                }
                ,
              "schema": { "$ref": "#/definitions/SecureMode" }
            }
        }
      },
      "post": {
        "description": "Updates the value of secureMode.\n",
        "parameters": [
          {"$ref": "#/parameters/interface"},
          {
            "name": "body",
            "in": "body",
            "required": true,
            "schema": { "$ref": "#/definitions/SecureMode" },
            "x-example":
              {
                "id":          "unique_example_id",
                "secureMode":  true
              }
          }
        ],
        "responses": {
            "200": {
              "description" : "",
              "x-example":
                {
                  "id":          "unique_example_id",
                  "secureMode":  true
                }
                ,
              "schema": { "$ref": "#/definitions/SecureMode" }
            }
        }
      }
    },
    "parameters": {
      "interface" : {
        "in" : "query",
        "name" : "if",
        "type" : "string",
        "enum" : ["oic.if.rw", "oic.if.baseline"]
      }
    },
    "definitions": {
      "SecureMode" :
            {
        "properties": {
          "id": {
            "description": "Instance ID of this specific resource",
            "maxLength": 64,
            "readOnly": true,
            "type": "string"
          },
          "if": {
            "description": "The interface set supported by this resource",
            "items": {
              "enum": [
                "oic.if.baseline",
                "oic.if.ll",
                "oic.if.b",
                "oic.if.lb",
                "oic.if.rw",
                "oic.if.r",
                "oic.if.a",
                "oic.if.s"
              ],
              "type": "string"
```

 

```
            },
            "minItems": 1,
            "readOnly": true,
            "type": "array"
          },
          "n": {
            "description": "Friendly name of the resource",
            "maxLength": 64,
            "readOnly": true,
            "type": "string"
          },
          "rt": {
            "description": "Resource Type",
            "items": {
              "maxLength": 64,
              "type": "string"
            },
            "minItems": 1,
            "readOnly": true,
            "type": "array"
          },
          "secureMode": {
            "description": "Status of the Secure Mode",
            "type": "boolean"
          }
        },
        "required": [
          "secureMode"
        ],
        "type": "object"
      }

  }
}
```

#### 9.2.6 Property 정의

| Property 명칭 | 값 유형 | 필수 | 액세스 모드 | 설명 |
|---|---|---|---|---|
| secureMode | boolean | 예 | Read Write | 보안 모드의 상태 |

#### 9.2.7 CRUDN 동작

| Resource URI 예 | Create | Read | Update | Delete | Notify |
|---|---|---|---|---|---|
| /example/SecureModeResURI | | get | post | | get |

### 9.3 AllJoyn Object

#### 9.3.1 개요

이 resource 는 동일한 AllJoyn object 로부터 유래된 resource 의 collection 이다.

#### 9.3.2 URI 경로 예

/example/AllJoynObject/

#### 9.3.3 Resource Type

resource type (rt)은 oic.r.alljoynobject 로 정의된다.

#### 9.3.4 RAML 정의

```
#%RAML 0.8
```

 

```
title: OCFAllJoynObject
version: v1.0.0-20170531

traits:
 - interface-baseline:
     queryParameters:
       if:
         enum: ["oic.if.baseline"]
 - interface-ll:
     queryParameters:
       if:
         enum: ["oic.if.ll"]


/example/AllJoynObject/?if=oic.if.baseline:
  description: |
    This resource is a collection of resources that were all derived from the same AllJoyn object.

  is: ['interface-baseline']

  get:
    description: |
      Retrieves the current AllJoyn object information.
    responses:
      200:
        body:
          application/json:
            schema: |
              {
                "id": "https://www.openconnectivity.org/ocf-
apis/bridging/schemas/oic.r.alljoynobject.json#",
                "$schema": "http://json-schema.org/draft-04/schema#",
                "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
reserved.",
                "title": "AllJoyn Object",
                "definitions": {
                  "oic.r.alljoynobject": {
                    "type": "object",
                    "allOf": [
                      {
                        "$ref": "../../core/schemas/oic.collection-
schema.json#/definitions/oic.collection"
                      },
                      {
                        "properties": {
                          "rt": {
                            "type": "array",
                            "minItems": 2,
                            "maxItems": 2,
                            "uniqueItems": true,
                            "items": {
                              "enum": ["oic.r.alljoynobject","oic.wk.col"]
                            }
                          }
                        }
                      }
                    ]
                  }
                },
                "type": "object",
                "allOf": [
                  {"$ref": "../../core/schemas/oic.core-schema.json#/definitions/oic.core"},
                  {"$ref": "#/definitions/oic.r.alljoynobject"}
```

 

```
                ]
              }
            example: |
              {
                "rt":    ["oic.r.alljoynobject","oic.wk.col"],
                "id":    "unique_example_id",
                "links": [
                  {"href": "/myRes1URI", "rt": ["x.example.widget.false"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                  {"href": "/myRes2URI", "rt": ["x.example.widget.true"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                  {"href": "/myRes3URI", "rt": ["x.example.widget.method1"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                  {"href": "/myRes4URI", "rt": ["x.example.widget.method2"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]}
                ]
              }
/example/AllJoynObject/?if=oic.if.ll:
  description: |
    This resource is a collection of resources that were all derived from the same AllJoyn object.

  is: ['interface-ll']

  get:
    description: |
      Retrieves the Links in the current AllJoyn object.
    responses:
      200:
        body:
          application/json:
            schema: |
              {
                "id": "https://www.openconnectivity.org/ocf-
apis/bridging/schemas/oic.r.alljoynobject-ll#",
                "$schema": "http://json-schema.org/draft-04/schema#",
                "description" : "Copyright (c) 2017 Open Connectivity Foundation, Inc. All rights
reserved.",
                "title": "AllJoyn Object Links List Schema",
                "definitions": {
                  "oic.r.alljoynobject-ll": {
                    "allOf": [
                      {
                        "$ref": "../../core/schemas/oic.collection.linkslist-
schema.json#/definitions/oic.collection.alllinks"
                      }
                    ]
                  }
                },
                "allOf": [
                  {"$ref": "#/definitions/oic.r.alljoynobject-ll"}
                ]
              }
            example: |
              [
                {"href": "/myRes1URI", "rt": ["x.example.widget.false"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                {"href": "/myRes2URI", "rt": ["x.example.widget.true"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                {"href": "/myRes3URI", "rt": ["x.example.widget.method1"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                {"href": "/myRes4URI", "rt": ["x.example.widget.method2"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]}
              ]
```

 

### 9.3.5 Swagger2.0 정의

```
{
  "swagger": "2.0",
  "info": {
    "title": "OCFAllJoynObject",
    "version": "v1.0.0-20170531",
    "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/AllJoynObject/?if=oic.if.ll" : {
      "get": {
        "description": "This resource is a collection of resources that were all derived from the
same AllJoyn object.\nRetrieves the Links in the current AllJoyn object.\n",
        "parameters": [
          {"$ref": "#/parameters/interface-ll"}
        ],
        "responses": {
            "200": {
              "description" : "",
              "x-example":
                [
                  {"href": "/myRes1URI", "rt": ["x.example.widget.false"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                  {"href": "/myRes2URI", "rt": ["x.example.widget.true"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                  {"href": "/myRes3URI", "rt": ["x.example.widget.method1"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                  {"href": "/myRes4URI", "rt": ["x.example.widget.method2"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]}
                ]
                ,
              "schema": { "$ref": "#/definitions/AllJoynObject-ll" }
            }
        }
      }
    },
    "/example/AllJoynObject/?if=oic.if.baseline" : {
      "get": {
        "description": "This resource is a collection of resources that were all derived from the
same AllJoyn object.\nRetrieves the current AllJoyn object information.\n",
        "parameters": [
          {"$ref": "#/parameters/interface-baseline"}
        ],
        "responses": {
            "200": {
              "description" : "",
              "x-example":
                {
                  "rt":    ["oic.r.alljoynobject","oic.wk.col"],
                  "id":    "unique_example_id",
```

```
                    "links": [
                      {"href": "/myRes1URI", "rt": ["x.example.widget.false"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                      {"href": "/myRes2URI", "rt": ["x.example.widget.true"], "if":
["oic.if.r","oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                      {"href": "/myRes3URI", "rt": ["x.example.widget.method1"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]},
                      {"href": "/myRes4URI", "rt": ["x.example.widget.method2"], "if":
["oic.if.rw","oic.if.baseline"], "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:11111"}]}
                    ]
                  }
                  ,
                "schema": { "$ref": "#/definitions/AllJoynObject" }
              }
          }
        }
      },
      "parameters": {
        "interface-ll" : {
          "in" : "query",
          "name" : "if",
          "type" : "string",
          "enum" : ["oic.if.ll"]
        },
        "interface-baseline" : {
          "in" : "query",
          "name" : "if",
          "type" : "string",
          "enum" : ["oic.if.baseline"]
        }
      },
      "definitions": {
        "AllJoynObject-ll" :
              {
            "allOf": [
              {
                "$ref": "../../core/schemas/oic.collection.linkslist-schema.json"
              }
            ]
          }

        ,
        "AllJoynObject" :
              {
            "allOf": [
              {
                "$ref": "../../core/schemas/oic.collection-schema.json"
              },
              {
                "properties": {
                  "id": {
                    "description": "Instance ID of this specific resource",
                    "maxLength": 64,
                    "readOnly": true,
                    "type": "string"
                  },
                  "if": {
                    "description": "The interface set supported by this resource",
                    "items": {
                      "enum": [
                        "oic.if.baseline",
                        "oic.if.ll",
                        "oic.if.b",
                        "oic.if.lb",
                        "oic.if.rw",
                        "oic.if.r",
                        "oic.if.a",
                        "oic.if.s"
                      ],
                      "type": "string"
```

 

```
                  },
                  "minItems": 1,
                  "readOnly": true,
                  "type": "array"
                },
                "n": {
                  "description": "Friendly name of the resource",
                  "maxLength": 64,
                  "readOnly": true,
                  "type": "string"
                },
                "rt": {
                  "items": {
                    "enum": [
                      "oic.r.alljoynobject",
                      "oic.wk.col"
                    ]
                  },
                  "maxItems": 2,
                  "minItems": 2,
                  "type": "array",
                  "uniqueItems": true
                }
              }
            }
          ],
          "type": "object"
        }

    }
}
```

### 9.3.6 CRUDN 동작

| Resource URI 예 | Create | Read | Update | Delete | Notify |
|---|---|---|---|---|---|
| /example/AllJoynObject/?if=oic.if.baseline | | get | post | | get |
| /example/AllJoynObject/?if=oic.if.ll | | get | | | get |

 