Definitions about properties/actions/events

July 14, 2021

Ryuichi Matsukura

Fujitsu

Introduction

- How to integrate other platforms/protocols like LWM2M or IPSO.
- Two primary models for API design: REST and RPC, for most software developers.

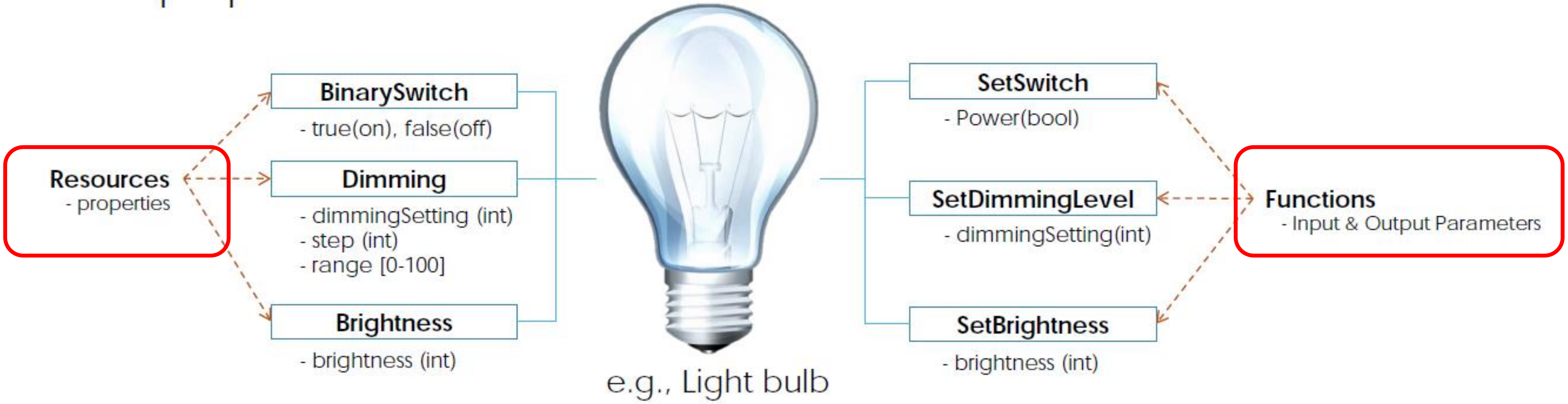
  - gRPC vs REST: Understanding gRPC, OpenAPI and REST and when to use them in API design
  - https://cloud.google.com/blog/products/api-management/understanding-grpc-openapi-and-rest-and-when-to-use-them
  - REST vs RPC: What problems are you trying to solve with your APIs?
  - https://cloud.google.com/blog/products/application-development/rest-vs-rpc-what-problems-are-you-trying-to-solve-with-your-apis

From OCF 2.0.3 Specification overview slides.

Approaches to definition of various Things

- By defining resources of things and its properties

- By defining functions/operations of things



- (no Verbs) + Objects

  *Fixed set of verbs (CRUDN) from transport layer will be used

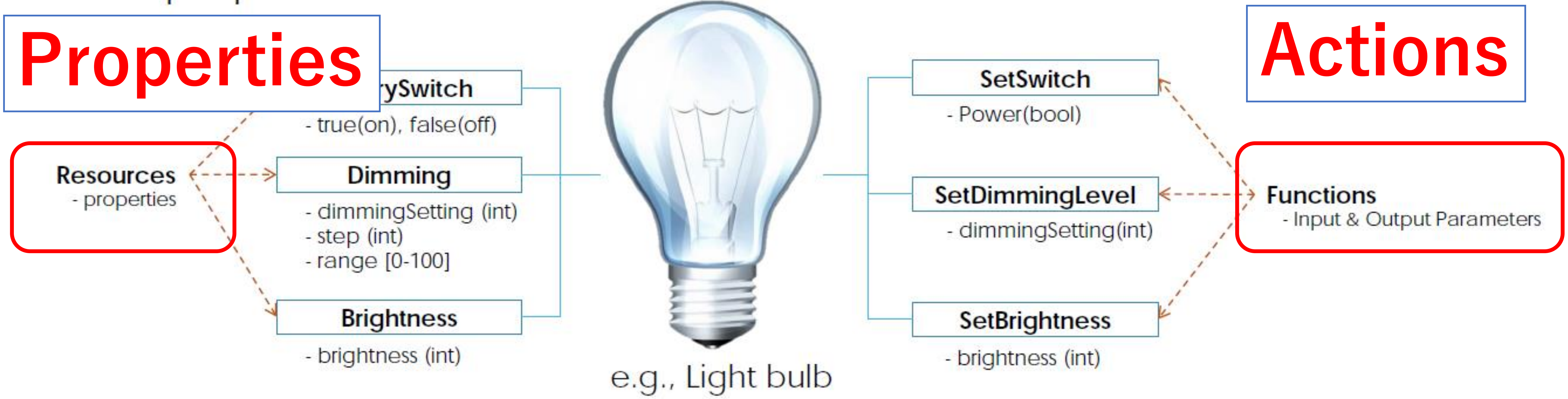
- Resource model in RESTful Architecture (e.g., W3C, CSEP, etc.)

- (Verbs + Objects)
- RPC model

From OCF 2.0.3 Specification overview slides.

Approaches to definition of various Things

- By defining resources of things and its properties
- By defining functions/operations of things



- (no Verbs) + Objects

  *Fixed set of verbs (CRUDN) from transport layer will be used
- Resource model in RESTful Architecture (e.g., W3C, CSEP, etc.)

- (Verbs + Objects)
- RPC model

```
“properties”: {
        “Switch”: {
                “type”: “boolean”,
                “forms”: […]
        },
        “Brightness”: {
                “type”: “number”,
                “forms”: […]
        },
},
```

Definition with just properties

```
“properties”: {
        “Switch”: {
                “type”: “boolean”,
                “readOnly”: true,
                “forms”: […]
        },
        “Brightness”: {
                “type”: “number”,
                “readOnly”: true,
                “forms”: […]
        },
},
“actions”: {
        “SetSwitch”: {
                “input”: {
                        “Switch”: “boolean”
                },
                {
                “output”: {
                        “Switch”: “boolean”
                },
                “forms”: […]
        },
        “SetBrightness”: {
                “input”: {
                        “Brightness”: “number”
                },
                {
                “output”: {
                        “Brightness”: “number”
                }
```

Definition with properties and actions

Property vs. Action

- Device defined by Properties can be easy to map to both Properties and Actions of WoT.

  - LWM2M, NETCONF, and TR-069(BBF) are defined with Properties.
    - These protocols are used in the device managements.
  - OCF, KNX, BACnet, and ECHONET mainly defined with Properties.

- Device defined by Actions CANNOT be easy to map to Properties of WoT unless if Input = Output.

Events

```
“properties”: {
        “Temperature”: {
                “type”: “number”,
                “observable”; true,
                “forms”; […]
        }
}
```

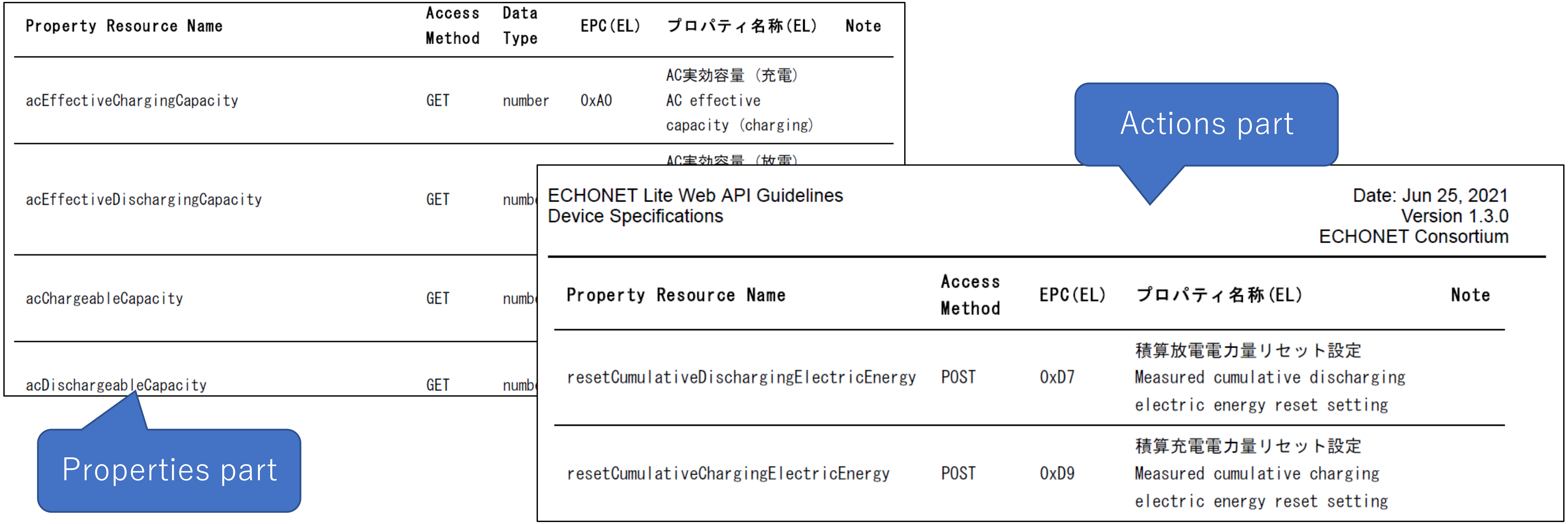
Definition with just properties

```
“properties”: {
        “Temperature”: {
                “type”: “number”,
                forms”; […]
        }
}
“events”: {
        “TemperatureChanged”: {
                “Temperature”: {
                        “type”: “number”,
                }
                “forms”: […]
        }
}
```

Definition with properties and events

Example: StorageBattery – ECHONET

- Storage battery has 55 properties and 2 actions



Property Resource Name	Access Method	Data Type	EPC(EL)	プロパティ名称(EL)	Note
acEffectiveChargingCapacity	GET	number	0xA0	AC実効容量（充電） AC effective capacity (charging)	
acEffectiveDischargingCapacity	GET	numbe		AC実効容量（放電）	
acChargeableCapacity	GET	numbe			
acDischargeableCapacity	GET	numbe			

Properties part

Actions part

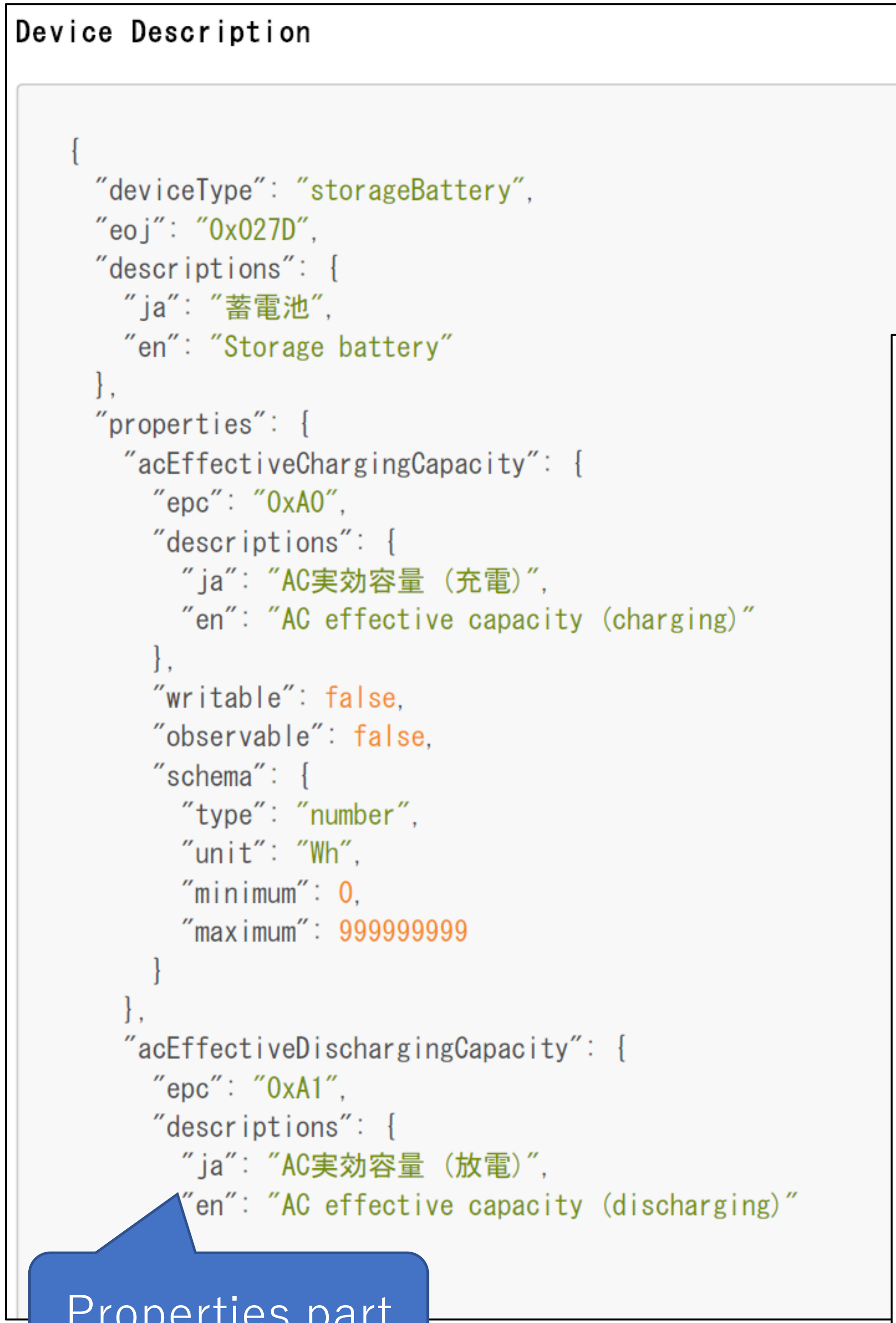
ECHONET Lite Web API Guidelines
Device Specifications

Date: Jun 25, 2021
Version 1.3.0
ECHONET Consortium

Property Resource Name	Access Method	EPC(EL)	プロパティ名称(EL)	Note
resetCumulativeDischargingElectricEnergy	POST	0xD7	積算放電電力量リセット設定 Measured cumulative discharging electric energy reset setting	
resetCumulativeChargingElectricEnergy	POST	0xD9	積算充電電力量リセット設定 Measured cumulative charging electric energy reset setting	

https://echonet.jp/wp/wp-content/uploads/pdf/General/Download/web_API/ECHONET_Lite_Web_API_Dev_Specs_v1.3.0.pdf

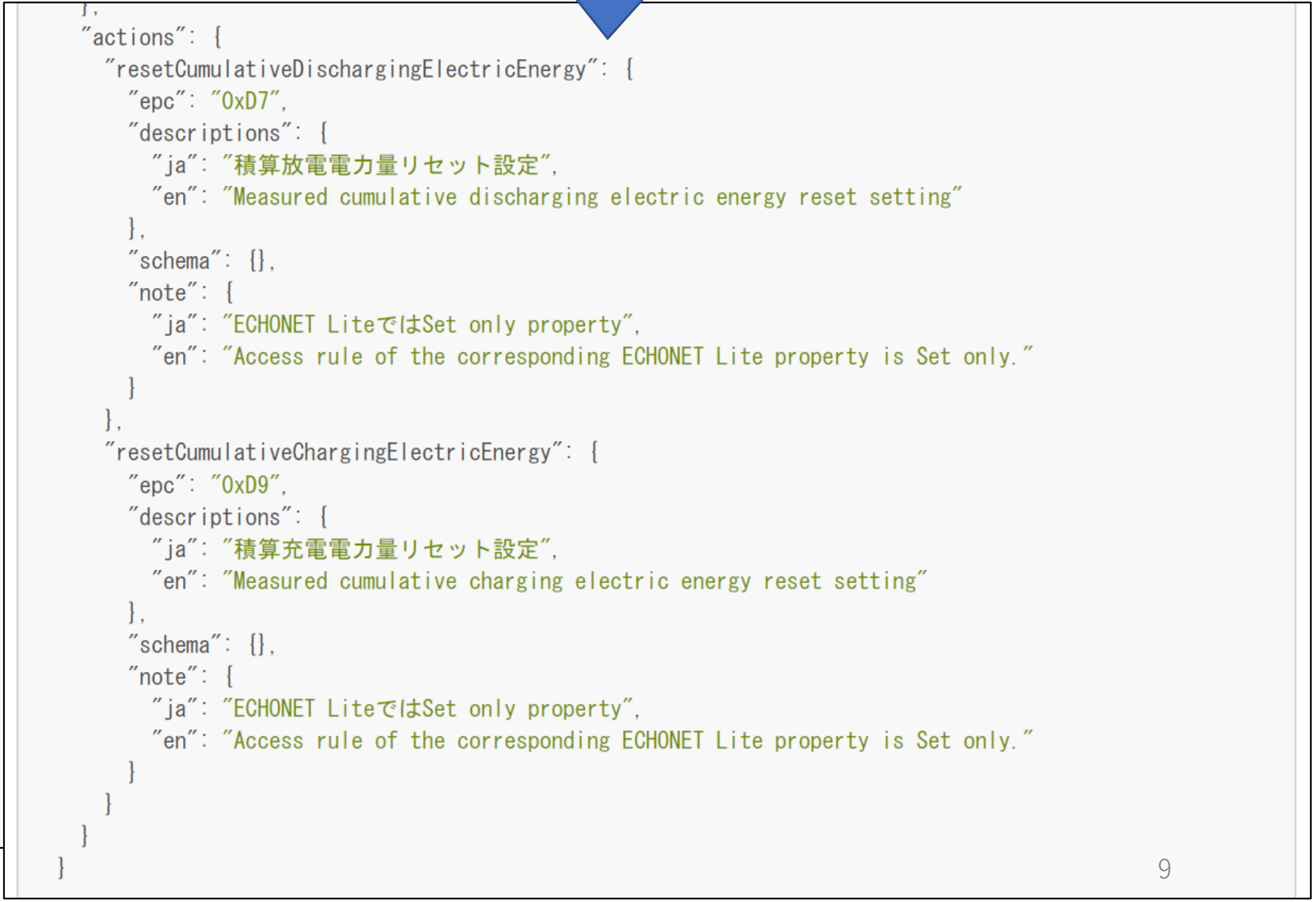
Device Description



```
{
  ”deviceType”: ”storageBattery”,
  ”eoj”: ”0x027D”,
  ”descriptions”: {
    ”ja”: ”蓄電池”,
    ”en”: ”Storage battery”
  },
  ”properties”: {
    ”acEffectiveChargingCapacity”: {
      ”epc”: ”0xA0”,
      ”descriptions”: {
        ”ja”: ”AC実効容量 (充電)”,
        ”en”: ”AC effective capacity (charging)”
      },
      ”writable”: false,
      ”observable”: false,
      ”schema”: {
        ”type”: ”number”,
        ”unit”: ”Wh”,
        ”minimum”: 0,
        ”maximum”: 999999999
      }
    },
    ”acEffectiveDischargingCapacity”: {
      ”epc”: ”0xA1”,
      ”descriptions”: {
        ”ja”: ”AC実効容量 (放電)”,
        ”en”: ”AC effective capacity (discharging)”
```

Properties part

Actions part



```
  },
  ”actions”: {
    ”resetCumulativeDischargingElectricEnergy”: {
      ”epc”: ”0xD7”,
      ”descriptions”: {
        ”ja”: ”積算放電電力量リセット設定”,
        ”en”: ”Measured cumulative discharging electric energy reset setting”
      },
      ”schema”: {},
      ”note”: {
        ”ja”: ”ECHONET LiteではSet only property”,
        ”en”: ”Access rule of the corresponding ECHONET Lite property is Set only.”
      }
    },
    ”resetCumulativeChargingElectricEnergy”: {
      ”epc”: ”0xD9”,
      ”descriptions”: {
        ”ja”: ”積算充電電力量リセット設定”,
        ”en”: ”Measured cumulative charging electric energy reset setting”
      },
      ”schema”: {},
      ”note”: {
        ”ja”: ”ECHONET LiteではSet only property”,
        ”en”: ”Access rule of the corresponding ECHONET Lite property is Set only.”
      }
    }
  }
}
```