Common Definition Format

One Data Model Liaison Group

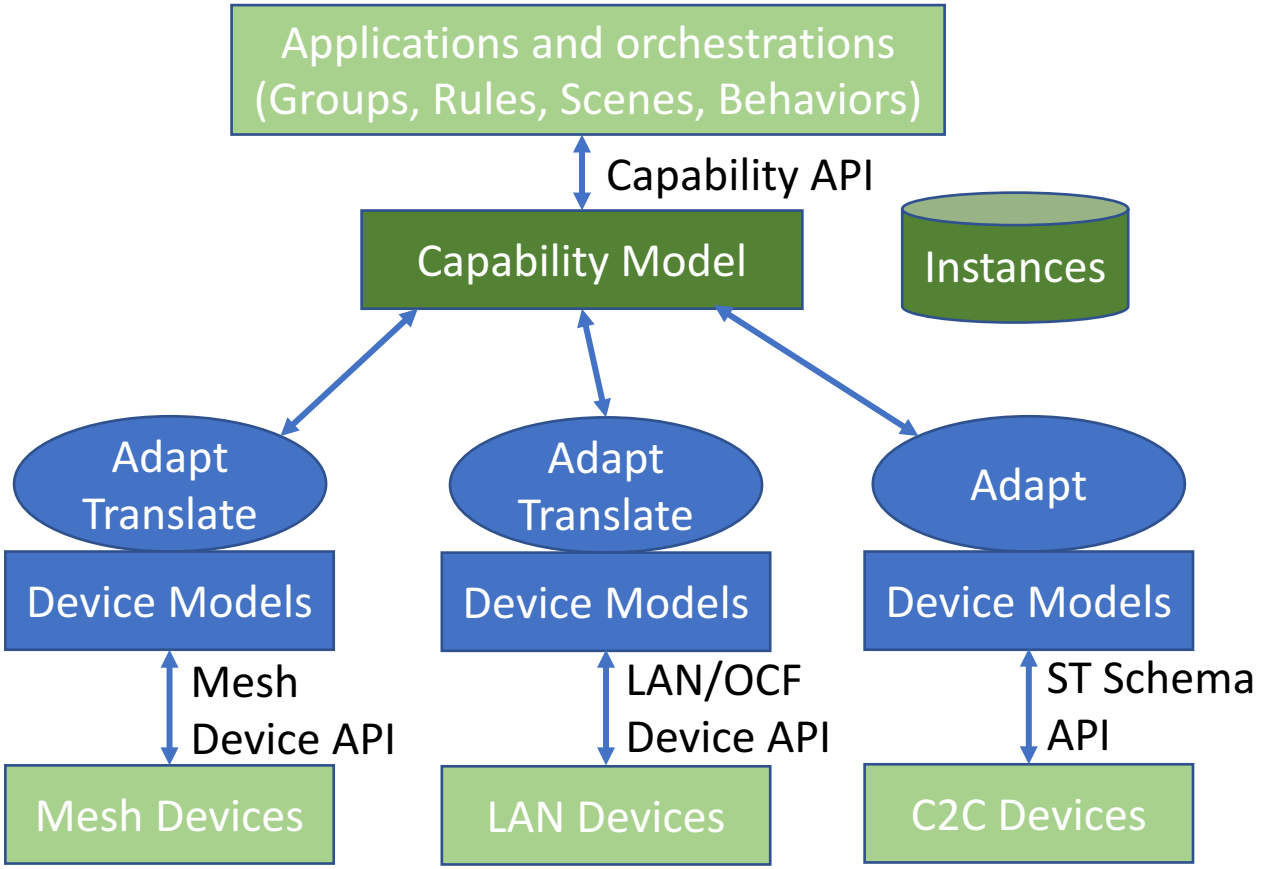
Michael Koster

March 24, 2019

Common Definition Format

- SmartThings Capability Model and Examples
- Common Definition Format
- UML model
- RDF Examples
  - ZCL lighting clusters - mapped capabilities
  - OCF lighting RTs - mapped capabilities
  - ST lighting Capabilities - mapped capabilities
- Protocol Binding using OCF Swagger
- A high level Thing Definition Language

SmartThings Capability Model



SmartThings Capability Definitions

```
name: Switch
status: live
attributes:
  switch:
    schema:
      type: object
      additionalProperties: false
      properties:
        value:
          $ref: SwitchState
      required: ["value"]
    type: ENUM
    values:
      - 'off'
      - 'on'
    enumCommands:
      - command: 'on'
        value: 'on'
      - command: 'off'
        value: 'off'
commands:
  'off': arguments: []
  'on': arguments: []
public: true
id: switch
ocfResourceType: x.com.st.powerswitch
version: 1
```

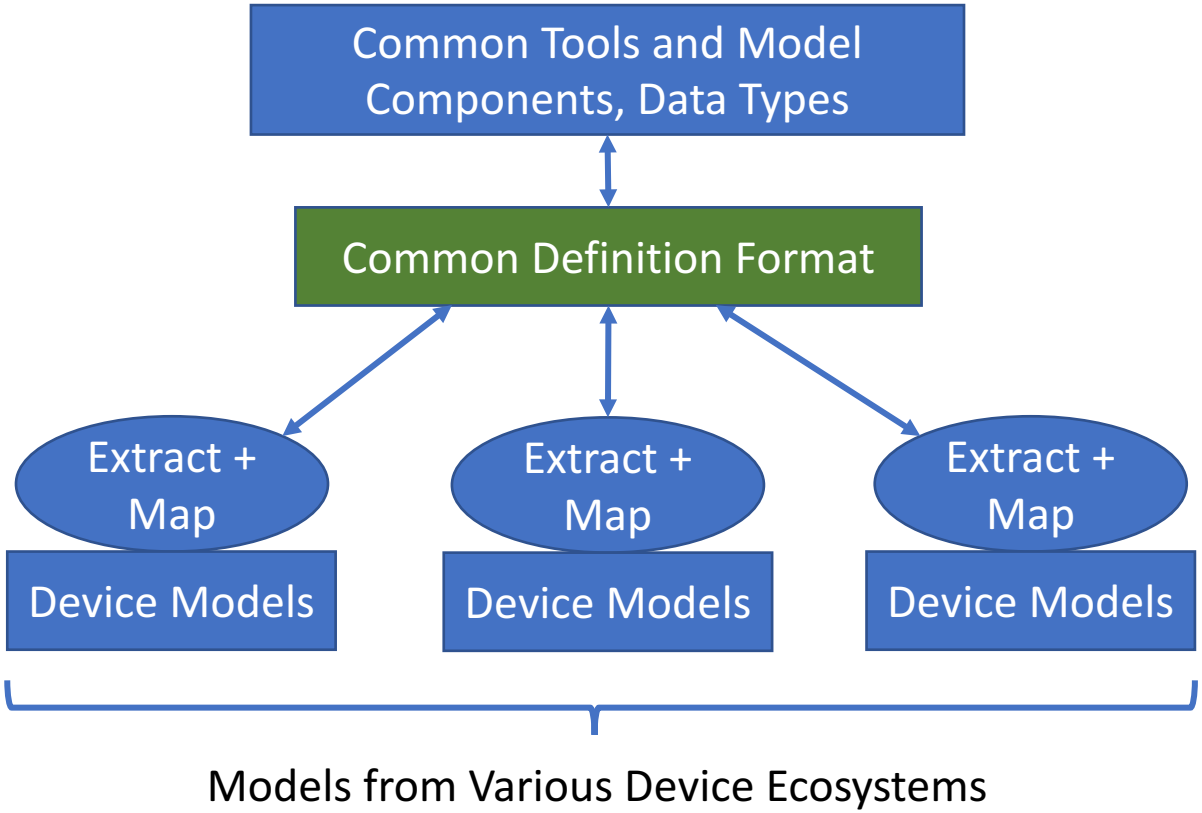
```
name: Switch Level
status: live
attributes:
  level:
    schema:
      $ref: IntegerPercent
    type: NUMBER
    setter: setLevel
commands:
  setLevel:
    arguments:
    - name: level
      schema:
        type: integer
        minimum: 0
        maximum: 100
      type: NUMBER
      required: true
    - name: rate
      schema:
        $ref: PositiveInteger
      type: NUMBER
      required: false
public: true
id: switchLevel
ocfResourceType: oic.r.light.dimming
version: 1
```

SmartThings DataType Definitions

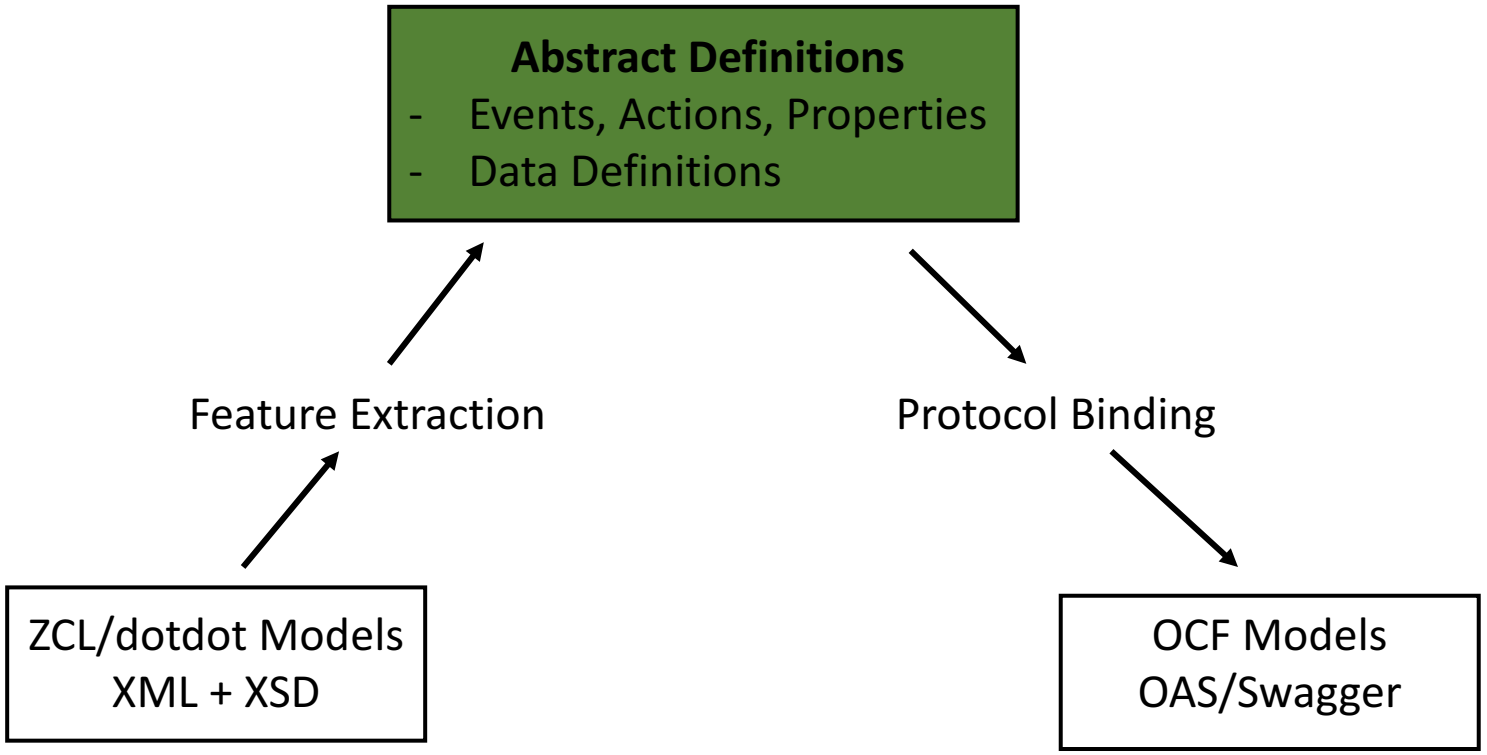
```
title: SwitchState
type: string
enum:
  - 'on'
  - 'off'
```

```
title: IntegerPercent
type: object
additionalProperties: false
properties:
  value:
    type: integer
    minimum: 0
    maximum: 100
  unit:
    type: string
    enum: ['%']
    default: '%'
required: ["value"]
```

Proposal for a Common Definition Format



Supports This Pattern



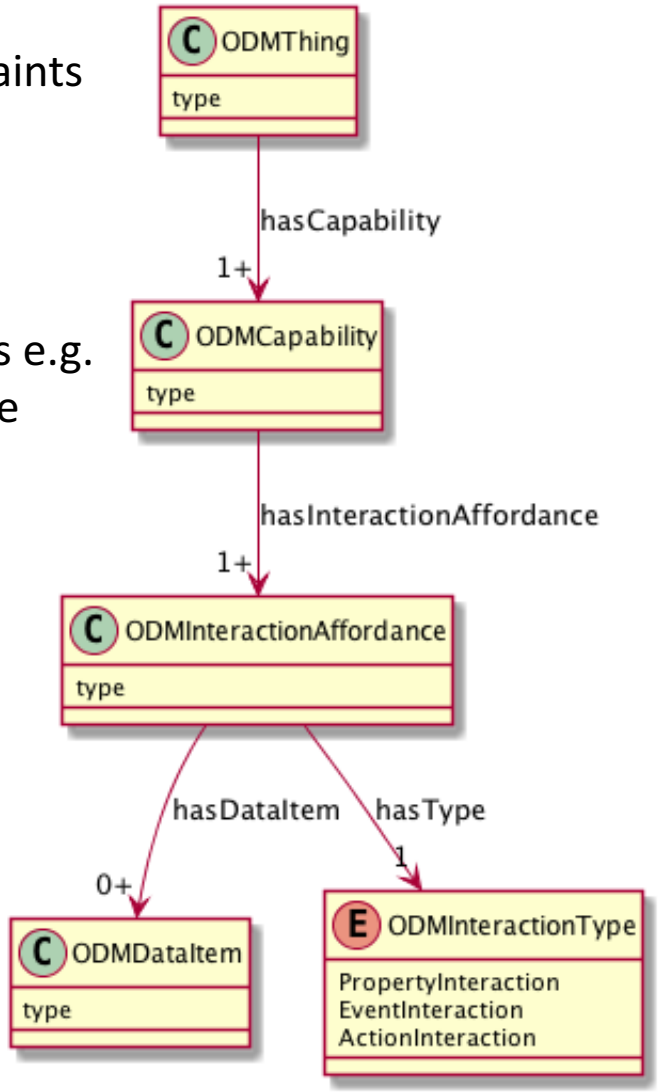
UML Model

Device Type and Constraints
e.g. Thermostat, Light

Composable Capabilities e.g.
onoff, level, temperature

Events, Actions, Properties

Data Types, Enums



Examples

- JSON-LD (JSON format with RDF extensions)
- Files for semantic definitions of specific types:
  - Thing (Device level definitions)
  - Capability (onoff, level…)
  - InteractionAffordance (Event, Action, Property)
  - Data Types (value types, enums)
- Definition hierarchy follows the UML model
- Core schema for the UML model in JSON-LD
- Full examples at: https://github.com/mjkoster/ODM-Examples
- (TBD) Thing definitions to apply optionality to Capability sets, Interactions, and Data Types

ST Sourced definitions

- ODM InteractionAffordance definitions, manually extracted from some SmartThings Capability definitions
- Attributes map to ODM Property type
- Commands map to ODM Action type

ST Based Capabilities

```
{
  "@id": "st:Switch",
  "rdfs:subClassOf": "odm:Capability",
  "rdfs:comment": "Basic On/Off Switch Capability",
  "rdfs:label": "SmartThings Switch Capability",
  "odm:hasInteractionAffordance": [
    "st:Switch.value",
    "st:Switch.on",
    "st:Switch.off"
  ]
},
{
  "@id": "st:SwitchLevel",
  "rdfs:subClassOf": "odm:Capability",
  "rdfs:comment": "Capability to control the level",
  "rdfs:label": "SmartThings SwitchLevel Capability",
  "odm:hasInteractionAffordance": [
    "st:SwitchLevel.level",
    "st:SwitchLevel.setLevel"
  ]
}
```

Properties, Actions, Events

```
{
  "@id": "st:SwitchLevel.level",
  "rdfs:comment": "The current level setting",
  "rdfs:label": "SwitchLevel levelProperty",
  "@type": "odm:PropertyInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": "st:SwitchLevel.levelData"
},
{
  "@id": "st:SwitchLevel.setLevel",
  "rdfs:comment": "Action to set the level",
  "rdfs:label": "SwitchLevel setLevelAction",
  "@type": "odm:ActionInteraction",
  "rdfs:subClassOf":"odm:InteractionAffordance",
  "odm:hasDataItem": [
    "st:SwitchLevel.levelData"
    "st:SwitchLevel.rateData"
  ]
}
```

Data Items

```
{
  "@id": "st:Switch.valueData",
  "rdfs:comment": "value data for Switch (on/off string
encoding)",
  "rdfs:label": "SmartThings SwitchLevel.levelData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:string",
  "js:enum": ["on", "off"]
},
{
  "@id": "st:SwitchLevel.levelData",
  "rdfs:comment": "Level data for SwitchLevel",
  "rdfs:label": "SmartThings SwitchLevel.levelData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:integer",
  "js:minimum": 0,
  "js:maximum": 100
},
{
  "@id": "st:SwitchLevel.rateData",
  "rdfs:comment": "Rate time data for setLevelAction",
  "rdfs:label": "SmartThings SwitchLevel.rateData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:integer",
  "js:minimum": 0,
  "js:maximum": 65535
}
```

ZCL Sourced definitions

- ODM InteractionAffordances, manually extracted from ZCL definitions
- Attributes map to ODM Property type
- Commands map to ODM Action type

ZCL Example

```
{
  "@id": "zcl:Level",
  "rdfs:subClassOf": "odm:Capability",
  "rdfs:comment": "Level Control Capability",
  "rdfs:label": "ZCL Level Capability",
  "odm:hasInteractionAffordance": [
    "zcl:Level.CurrentLevel",
    "zcl:Level.RemainingTime",
    "zcl:Level.OnOffTransitionTime",
    "zcl:Level.OnLevel",
    "zcl:Level.OnTransitionTime",
    "zcl:Level.OffTransitionTime",
    "zcl:Level.DefaultMoveRate",
    "zcl:Level.MoveToLevel",
    "zcl:Level.Move",
    "zcl:Level.Step",
    "zcl:Level.Stop",
    "zcl:Level.MoveToLevelWithOnOff",
    "zcl:Level.MoveWithOnOff",
    "zcl:Level.StepWithOnOff"
  ]
}
```

ZCL Example

```
{
  "@id": "zcl:Level.MoveToLevel",
  "rdfs:comment": "Action move to a given level",
  "rdfs:label": "ZCL Level.MoveToLevelAction",
  "@type": "odm:ActionInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": [
    "zcl:Level.LevelData",
    "zcl:Level.TransitionTimeData"
  ]
},
{
  "@id": "zcl:Level.Move",
  "rdfs:comment": "Action move at a given rate",
  "rdfs:label": "ZCL Level.MoveAction",
  "@type": "odm:ActionInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": [
    "zcl:Level.MoveModeData",
    "zcl:Level.RateData"
  ]
},
```

ZCL Example

```
{
  "@id": "zcl:Level.OffTransitionTimeData",
  "rdfs:comment": "Off Transition Time Data",
  "rdfs:label": "ZCL Level.OffTransitionTimeData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:integer",
  "js:minimum": 0,
  "js:maximum": 65534
},
{
  "@id": "zcl:Level.DefaultMoveRateData",
  "rdfs:comment": "Default Move Rate Data",
  "rdfs:label": "ZCL Level.DefaultMoveRateData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:integer",
  "js:minimum": 0,
  "js:maximum": 254
}
```

OCF Sourced definitions

- ODM InteractionAffordances, manually extracted from OCF Resource Type definitions
- Properties map to ODM Property type
- Actions added for simple cases like brightness change with ramp time

OCF Example

```
{
  "@id": "ocf:BinarySwitch",
  "rdfs:subClassOf": "odm:Capability",
  "rdfs:comment": "On/Off Switch Capability",
  "rdfs:label": "OCF BinarySwitch Capability",
  "odm:hasInteractionAffordance": [
    "ocf:BinarySwitch.value",
    "ocf.BinarySwitch.On",
    "ocf.BInarySwitch.Off"
  ]
},
{
  "@id": "ocf:Brightness",
  "rdfs:subClassOf": "odm:Capability",
  "rdfs:comment": "Capability to control the
brightness",
  "rdfs:label": "OCF Brightness Capability",
  "odm:hasInteractionAffordance": [
    "ocf:Brightness.Brightness",
    "ocf:Brightness.SetBrightness"
  ]
},
```

OCF Example

```
{
  "@id": "ocf:Brightness.brightness",
  "rdfs:comment": "Brightness Property",
  "rdfs:label": "OCF Brightness.brightnessProperty",
  "@type": "odm:PropertyInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": "ocf:Brightness.brightnessData"
},
{
  "@id": "ocf:Brightness.SetBrightness",
  "rdfs:comment": "Set Brightness Action",
  "rdfs:label": "OCF Brightness.SetBrightnessAction",
  "@type": "odm:ActionInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": [
    "ocf:Brightness.BrightnessData",
    "ocf:RampTime.RampTimeData"
    ]
},
```

OCF Example

```
{
  "@id": "ocf:Brightness.brightnessData",
  "rdfs:comment": "Brightness Data",
  "rdfs:label": "OCF Brightness.brightnessData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:integer",
  "js:minimum": 0,
  "js:maximum": 255
},
{
  "@id": "ocf:RampTime.ramptimeData",
  "rdfs:comment": "Ramp Time Data",
  "rdfs:label": "OCF RampTime.ramptimeData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:integer",
  "js:minimum": 0,
  "js:maximum": 65535
},
```

OCF Example

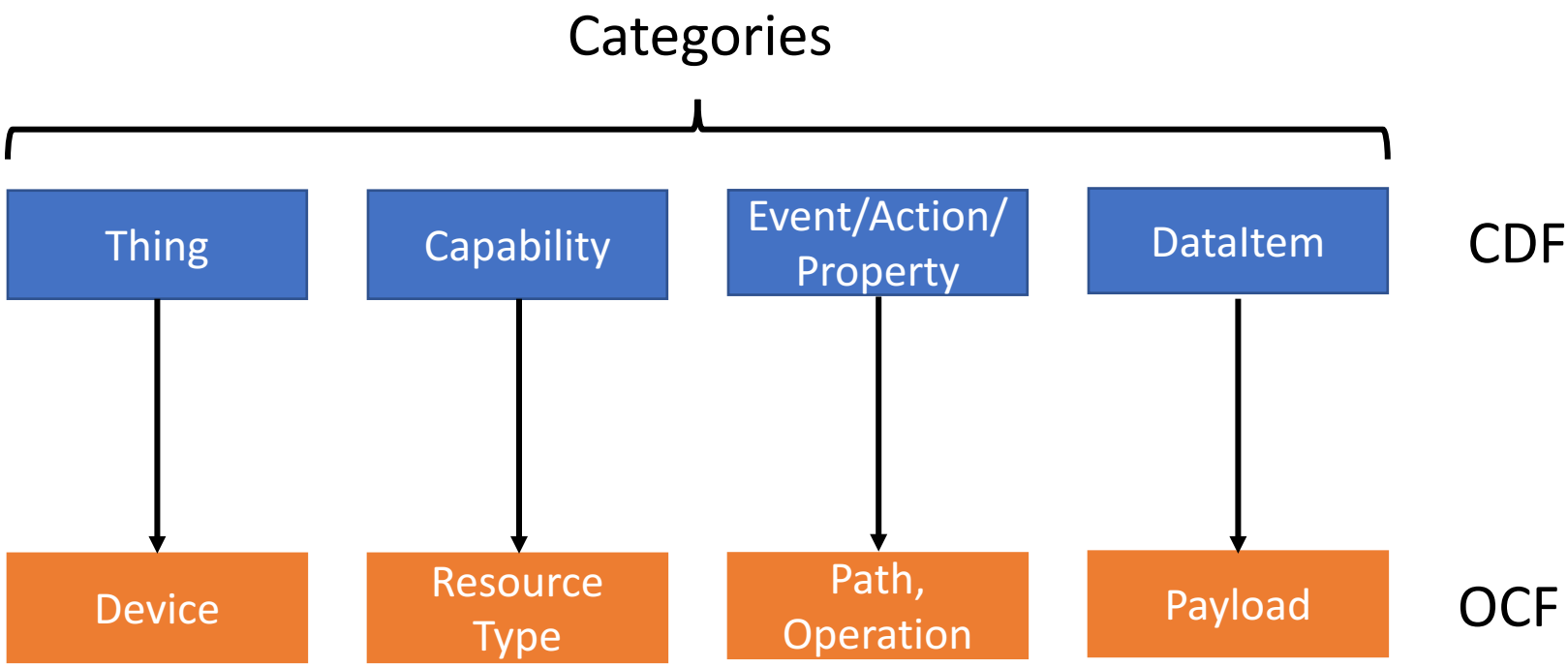
```
{
  "@id": "ocf:BinarySwitch.On",
  "rdfs:comment": "Binary Switch On Action",
  "rdfs:label": "OCF BinarySwitch.valueProperty",
  "@type": "odm.ActionInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": "ocf:BinarySwitch.OnData"
},

{
  "@id": "ocf:BinarySwitch.OnValueData",
  "rdfs:comment": "Boolean value for On state",
  "rdfs:label": "OCF BinarySwitch.OnData",
  "rdfs:subClassOf": "odm:DataItem",
  "odm:DataItemType": "js:boolean",
  "js:const": true
},
```

OCF Protocol Binding/Mapping

- Example of an OCF Resource Type definition (OAS/Swagger) file with annotations/extensions for modeling ODM Actions
- Mapping can be done to existing OCF types in some cases using CDF annotation
- OAS target patterns can be generated using templates and annotated with CDF semantics
- Enables an ODM-Capable Bridge or adaptation client to use ODM to generate OCF API calls

OCF Protocol Binding Example



OCF Definition with annotations

```
    "title": "Binary Switch",
    "version": "v1.1.0-20160519",
    "license": {
      "name": "copyright 2016-2017 Open Connectivity
Foundation, Inc. All rights reserved.",
    }
  },
  "@type": "ocf:BinarySwitch",
  "schemes": ["http"],
  "consumes": ["application/json"],
  "produces": ["application/json"],
  "paths": {
    "/BinarySwitchResURI" : {
      "@type": [
        "ocf:BinarySwitch.value",
        "ocf:BinarySwitch.On",
        "ocf:BinarySwitch.Off"
      ],
```

OCF Definition with annotated paths and operations

```
"paths": {
  "/BinarySwitchResURI" : {
    "@type": [
      "ocf:BinarySwitch.value",
      "ocf:BinarySwitch.On",
      "ocf:BinarySwitch.Off"
    ],
    "get": {
      "@type": "ocf:BinarySwitch.value",
     }

    "post": {
      "@type": [
        "ocf:BinarySwitch.value",
        "ocf:BinarySwitch.On",
        "ocf:BinarySwitch.Off"
      ],
    }
```

OCF Definition with data annotations

```
    (...)
    "minItems": 1,
    "readOnly": true,
    "type": "array"
},
"value" :
          {
    "@type": [
      "ocf:BinarySwitch.valueData",
      "ocf:BinarySwitch.OnData",
      "ocf:BinarySwitch.OffData"
    ],
    "description": "Status of the switch",
    "type": "boolean"
},
```

Example Device Level Definitoin

```
{
  "@id": "st:DimmableLight",
  "rdfs:subClassOf": "odm:Thing",
  "rdfs:comment": "Simple Dimmable Light Bulb",
  "rdfs:label": "SmartThings DimmableLight",
  "odm:hasCapability": [
    "st:Switch",
    "st:SwitchLevel"
  ]
}
```

Thing Definition Language

- A high level language using a few simple patterns
- Developers can create and augment definitions in this high level language
- Automatic feature extraction can output this language
- The JSON Definition Format can be produced automatically, and protocol bindings generated

Thing Definition Language

```
context http://onedm.org#tdl
uses [odm js]
scope st
define DimmableLight {
  extends Thing
  hasCapability [
    st:Switch
    st:SwitchLevel
  ]
}
```

```
context http://onedm.org#tdl
uses [odm js]
scope st
define [
  Switch {
    extends Capability
    hasProperty Switch.value
    hasAction [Switch.on Switch.off]
  }
  Switch.value {
    extends Property
    hasDataItem Switch.valueData
  }
  Switch.on {extends Action}
  Switch.off {extends Action}
  Switch.valueData {
    extends DataItem
    type string
    enum [on off]
  }
]
```

Keywords

- `context` - works like JSON-LD context to define namespaces and terms
- `uses` - specifies one or more default source namespaces, evaluated in order
- `scope` - specifies default namespace that definitions are added to
- `define` - creates a definition in some namespace, args are a new term and a definition block
- `extends` - specifies a class template to use in the definition block

Structure

- keywords
- [list] items determined by keyword, use where multiple items are allowed
- { block } contains key-value pairs, whitespace delimited, according to the class template defined by extends
- Basic Definition Format
  - define <new term> {extends <class> key1 value1 key2 value2 … }
  - key-value pairs add constraints to the definition

Namespace Resolution

0. keywords
1. local block
2. enclosing block
3. namespace declared with scope keyword
4. namespaces declared with uses keyword, in declared order

Example Definition

```
context http://onedm.org#tdl
uses [odm js]
scope st
define [
  Switch {
    extends Capability
    hasProperty Switch.value
    hasAction [Switch.on Switch.off]
  }
  Switch.value {
    extends Property
    hasDataItem Switch.valueData
  }
  Switch.on {extends Action}
  Switch.off {extends Action}
  Switch.valueData {
    extends DataItem
    type string
    enum [on off]
  }
]
```

Example Generated JSON

```
{
  "@id": "st:Switch",
  "rdfs:subClassOf": "odm:Capability",
  "odm:hasInteractionAffordance": [
    "st:Switch.value",
    "st:Switch.on",
    "st:Switch.off"
  ]
},
{
  "@id": "st:Switch.value",
  "@type": "odm.PropertyInteraction",
  "rdfs:subClassOf": "odm:InteractionAffordance",
  "odm:hasDataItem": "st:Switch.valueData"
},
{
  "@id": "st:Switch.valueData",
  "rdfs:subClassOf": "odm:DataItem",
  "js:type": "js:string",
  "js:enum": ["on", "off"]
},
```

Defining a Device Type

```
context http://onedm.org#tdl
uses [odm js]
scope st
define DimmableLight {
  extends Thing
  hasCapability [
    st:Switch
    st:SwitchLevel
  ]
}
```