Thing Definition Language

One Data Model Liaison Group

Michael Koster

March 26, 2019

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]
  }
]
```

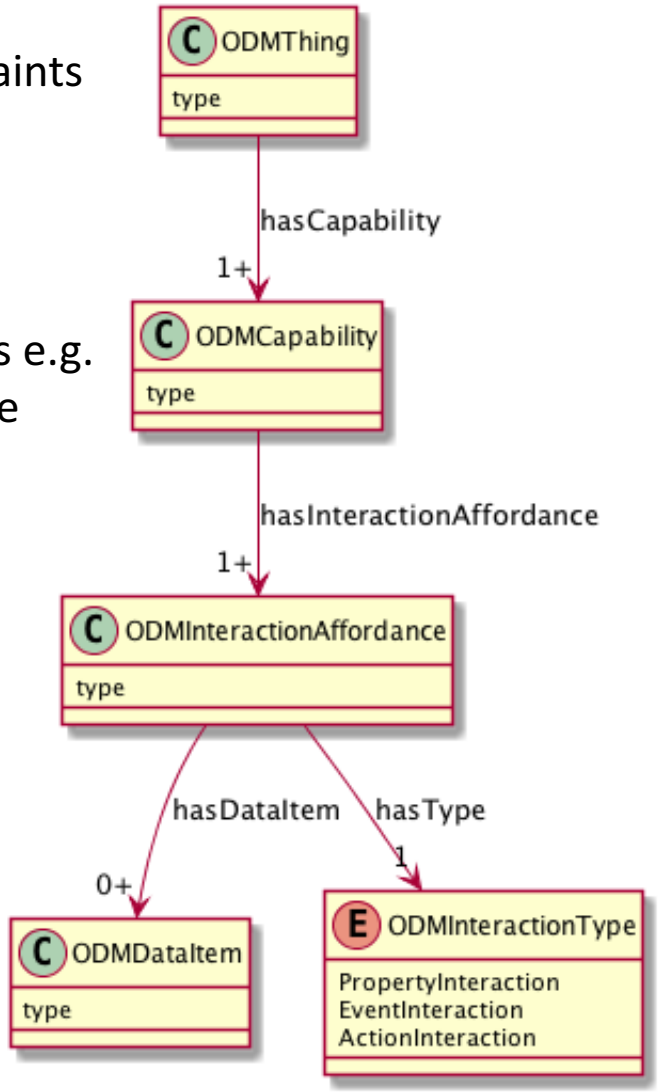
UML Model

Device Type and Constraints
e.g. Thermostat, Light

Composable Capabilities e.g.
onoff, level, temperature

Events, Actions, Properties

Data Types, Enums



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

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
  ]
}
```