Bridging the Gap to the Web of Things

On the Conversion between WoT Data Models and the Semantic Definition Format

Jan Romann, University of Applied Sciences Emden/Leer, Germany

SWoCoT 2023 – First International Workshop on Semantic Web on Constrained Things, May 28, 2023
Semantic IoT device descriptions

Important for both machine-to-machine and human-to-machine interaction.

- Protocols
- Data formats
- Security mechanisms
- (human-readable) meta-data
Two kinds of interoperability problems

1. At the *instance* level
   - Incompatible data models and interface descriptions

2. At the *ecosystem* level
   - Lack of a common data and interaction model
   - → Difficult to universally describe *classes* of devices
Two Potential Solutions

- Web of Things Thing Description (WoT TD)
  - W3C Recommendation since 2020
  - Version 1.1 about to be published
  - Work on version 2.0 will begin in a few weeks

- Semantic Definition Format (SDF)
  - Specified by an IETF working group (ASDF)
  - Initiated by the OneDM group
  - Internet-Draft (currently version 13)
WoT TD and SDF

WoT TD

- Focus on device instances
- Simplifies interactions between Things and TD consumers

SDF

- Focus on device classes
- Aspires to be a universal modelling language

→ WoT TD as an interesting conversion target for SDF

However: Lack of a “canonical” mapping between the two specifications
Goals

- Comprehensive mapping between SDF and WoT
  - Support for round tripping
- Converter implementation
- Identify gaps/potential for alignment
- Discuss limitations/future improvements
Many similarities, but also important differences.
Minimal Examples for WoT TD and SDF

**WoT Thing Description**

```json
{
    "@context": "https://www.w3.org/2022/wot/td/v1.1",
    "title": "Smart Lamp",
    "security": [...],
    "securityDefinitions": {...},
    "properties": { ... },
    "actions": { ... },
    "events": { ... }
}
```

**SDF Model**

```json
{
    "sdfObject": {
        "smartLamp": {
            "label": "Smart Lamp",
            "sdfProperty": { ... },
            "sdfAction": { ... },
            "sdfEvent": { ... }
        }
    }
}
```
Two important questions

1. How can we map abstract data models to WoT (TD)?
2. How can we map instance- or ecosystem-specific information to SDF?
Thing Descriptions vs. Thing Models

WoT Thing Description

```
{
    "@context": "https://www.w3.org/2022/wot/td/v1.1",
    "title": "Smart Lamp",
    "security": [ ... ],
    "securityDefinitions": { ... },
    "properties":
    "status": {
        "type": "string",
        "forms": [ ... ]
    }
}
```

WoT Thing Model

```
{
    "@context": "https://www.w3.org/2022/wot/td/v1.1",
    "title": "Smart Lamp",
    "@type": "tm:ThingModel",
    "properties":
    "status": {
        "type": "string"
    }
}
```
SDF ↔ Thing Model Conversion

SDF Model

```json
{
    "sdfObject": {
        "Lamp": {
            "label": "Smart Lamp",
            "sdfProperty": {
                "status": {
                    "type": "string"
                }
            }
        }
    }
}
```

WoT Thing Model

```json
{
    "@context": [
        "https://www.w3.org/2022/wot/td/v1.1",
        {"sdf": "https://example.org/sdf"},
    ],
    "@type": "tm:ThingModel",
    "title": "Smart Lamp",
    "sdf:objectKey": "Lamp",
    "properties": {
        "status": {
            "type": "string"
        }
    }
}
```
SDF Mapping Files (with WoT TD terms)

SDF Model

```json
{
    "sdfObject": {
        "Lamp": {
            "label": "Smart Lamp",
            "sdfProperty": {
                "status": {
                    "type": "string"
                }
            }
        }
    }
}
```

SDF Mapping File

```json
{
    "map": {
        "#/sdfObject/Lamp": {
            "@context": "...",
            "security": [...],
            "securityDefinitions": {...},
        },
        "#/sdfObject/Lamp/sdfProperty/status": {
            "forms": [...]
        }
    }
}
```
Conversion Process

Conversion Process Diagram:
- Protocol Bindings
- Metadata
- Placeholder Map
- Thing Descriptions
  - Conversion
  - Augmentation
  - Thing Models
  - Conversion
  - SDF Model
  - Mapping Files
Conversion of Nested Models

**SDF Model**

```json
{
    "sdfThing": {
        "TopLevel": {
            "sdfProperty": {
                ...
            },
            "sdfObject": {
                "SecondLevel": {
                    ...
                }
            }
        }
    }
}
```

**WoT “TM Collection”**

```json
{
    "TopLevel": {
        ...
        "properties": {
            ...
        },
        "links": [ {
            "href": "/SecondLevel",
            "rel": "tm:submodel",
        } ]
    },
    "SecondLevel": {
        ...
    }
}
```
Converter between WoT TD and SDF (including protocol bindings).

Project description

SDF-WoT-Converter

This repository provides a Python-based converter from SDF to WoT TD including Thing Models.

The converter is both usable as a library and a command line tool. It provides conversion functions between WoT Thing Descriptions, WoT Thing Models and SDF Models (one for each combination). You can find a number of examples for the usage of the converter down below as well as overviews for the conversion between SDF and WoT TMs.

The CI pipeline is set up to automatically convert all (valid) models from the oneDM playground to WoT Thing Models.

https://github.com/JKRhb/sdf-wot-converter-py
Library API

- Six top level functions
  - SDF to WoT TM and WoT TD
  - WoT TD to WoT TM and SDF
  - WoT TM to SDF and WoT TD
CLI tool

usage: sdf-wot-converter [-h] [--indent INDENT] [--suppress-roundtripping] {sdf-to-tm,sdf-to-td,tm-to-sdf,tm-to-td,td-to-sdf,td-to-tm} ...

Convert from SDF to WoT and vice versa.

positional arguments:
{sdf-to-tm,sdf-to-td,tm-to-sdf,tm-to-td,td-to-sdf,td-to-tm}
  sdf-to-tm        Converts an SDF model and mapping files to one or more WoT Thing Models.
  sdf-to-td        Converts an SDF model and mapping files to one or more WoT Thing Descriptions.
  tm-to-sdf        Converts a WoT Thing Model to an SDF model and zero or more mapping files.
  tm-to-td         Converts a WoT Thing Model to a WoT Thing Description.
  td-to-sdf        Converts one or more WoT Thing Models to an SDF model and zero or more mapping files.
  td-to-tm         Converts one or more WoT Thing Descriptions to one or more WoT Thing Models.

optional arguments:
  -h, --help       show this help message and exit
  --indent INDENT  Indentation depth for the output JSON files.
  --suppress-roundtripping
                  Suppresses the addition of additional fields for enabling roundtripping, like "sdf:objectKey".
SDF WoT converter

```json
{
    "info": {
        "title": "Example file for OneDM Semantic Definition Format",
        "version": "2019-04-24",
        "copyright": "Copyright 2019 Example Corp. All rights reserved.",
        "license": "https://example.com/license"
    },
    "namespace": {
        "cap": "https://example.com/capability/cap"
    },
    "defaultNamespace": "cap",
    "sdfObject": {
        "Switch": {
            "sdfProperty": {
                "value": {
                    "description": "The state of the switch; false for off and true for on.",
                    "type": "boolean"
                }
            },
            "sdfAction": {
                "on": {
                    "description": "Turn the switch on; equivalent to setting value to true."
                },
                "off": {
                    "description": "Turn the switch off; equivalent to setting value to false."
                },
                "toggle": {
                    "description": "Toggle the switch; equivalent to setting value to its complement."
                }
            }
        }
    }
}
```

Settings

- [x] Output SDF Mapping files
- [x] Include additional fields for roundtripping

https://sdfwotconverter.pythonanywhere.com/
SDF Conversion Tool Collection

http://wishi.nomadiclab.com/sdf-converter/
Conclusion and Future Work

- WoT and SDF can be mapped to each other
  - Additional concepts such as SDF mapping files are needed
  - WoT TMs as intermediaries
- Flexible converter implementation in Python
  - Library, CLI tool, and web interface
Conclusion and Future Work

- However: More standardization work needed
  - “Canonical” mapping specification?
  - Nested TMs/TDs in a single document?
  - Conversion of TMs to TDs
- Semantic alignment needed
  - Additional SDF concepts (sdfRelation, sdfChoice)
  - Namespace concept of SDF vs JSON-LD
- More thorough evaluation needed
Thank you for your attention!
References


Backup
JSON-LD @context vs namespaces

SDF Model

```
{
  "namespace": {
    "cap": "https://example.com/capability/cap"
  },
  "defaultNamespace": "cap",
  "sdfObject": { ... }
}
```

WoT Thing Model

```
{
  "@context": [
    "https://www.w3.org/2022/wot/td/v1.1",
    {
      "sdf": "https://example.org/sdf",
      "cap": "https://example.com/capability/cap",
    }
  ],
  "@type": "tm:ThingModel"
}
```
Reference Mechanisms

**SDF (sdfRef)**

```json
{
    "sdfObject": {
        "foo": {
            "sdfProperty": {
                "bar": {
                    "type": "string"
                },
                "baz": {
                    "sdfRef": "/sdfObject/foo/sdfProperty/bar"
                }
            }
        }
    }
}
```

**WoT (tm:ref)**

```json
{
    ..., 
    "properties": {
        "bar": {
            "type": "string",
            "observable": true
        },
        "baz": {
            "tm:ref": "/properties/bar",
            "observable": true
        }
    }
}
```
sdfChoice

SDF (sdfChoice)

```json
{
    "sdfChoice": {
        "foo": {
            "const": 2
        },
        "bar": {
            "const": 5
        }
    }
}
```

WoT (enum) – mapping is not ideal yet

```json
{
    "enum": [
        {
            "sdf:choiceName": "foo",
            "const": 2
        },
        {
            "sdf:choiceName": "bar",
            "const": 5
        }
    ]
}
```
### Table 3
Overview of the available parameters for the sub-commands of our CLI.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Arguments</th>
<th>Sub-Commands</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>--input, -i*</td>
<td>File path(s) or URL(s)(^a)</td>
<td>all</td>
<td>—</td>
</tr>
<tr>
<td>--output, -o</td>
<td>File path</td>
<td>all</td>
<td>—</td>
</tr>
<tr>
<td>--suppress-roundtripping</td>
<td>—</td>
<td>all</td>
<td>False</td>
</tr>
<tr>
<td>--indent</td>
<td>Natural number</td>
<td>all</td>
<td>4</td>
</tr>
<tr>
<td>--origin-url</td>
<td>URL</td>
<td>sdf-to-tm, sdf-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--mapping-files</td>
<td>Zero or more file paths</td>
<td>sdf-to-tm, sdf-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--title</td>
<td>String</td>
<td>sdf-to-tm, sdf-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--version</td>
<td>String</td>
<td>sdf-to-tm, sdf-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--copyright</td>
<td>String</td>
<td>sdf-to-tm, sdf-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--license</td>
<td>String</td>
<td>sdf-to-tm, sdf-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--meta-data</td>
<td>File path or URL</td>
<td>tm-to-sdf, tm-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--bindings</td>
<td>File path or URL</td>
<td>tm-to-sdf, tm-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--placeholder-map</td>
<td>File path or URL</td>
<td>tm-to-sdf, tm-to-td</td>
<td>—</td>
</tr>
<tr>
<td>--mapping-file-output</td>
<td>File path</td>
<td>tm-to-sdf, td-to-sdf</td>
<td>—</td>
</tr>
<tr>
<td>--remove-not-required-affordances</td>
<td>—</td>
<td>tm-to-td</td>
<td>False</td>
</tr>
</tbody>
</table>

* Mandatory Parameter.

\(^a\) Can be multiple paths/URLs when converting from WoT TM/TD – the imported TMs/TDs are then treated as Collections.
### Table 1
Overview of mappings of the most important SDF keywords to WoT.

<table>
<thead>
<tr>
<th>SDF Keyword</th>
<th>WoT Class/Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdfThing</td>
<td>TM with <code>tm:submodel</code> links</td>
</tr>
<tr>
<td>sdfObject</td>
<td>TM without <code>tm:submodel</code> links</td>
</tr>
<tr>
<td>sdfProperty</td>
<td>PropertyAffordance</td>
</tr>
<tr>
<td>writable</td>
<td><code>readOnly</code> (negated)</td>
</tr>
<tr>
<td>readable</td>
<td><code>writeOnly</code> (negated)</td>
</tr>
<tr>
<td>sdfAction</td>
<td>ActionAffordance</td>
</tr>
<tr>
<td>sdfOutputData</td>
<td>output</td>
</tr>
<tr>
<td>sdfInputData</td>
<td>input</td>
</tr>
<tr>
<td>sdfEvent</td>
<td>EventAffordance</td>
</tr>
<tr>
<td>sdfOutputData</td>
<td>output</td>
</tr>
<tr>
<td>sdfData</td>
<td><code>schemaDefinitions</code> (at the TM level)</td>
</tr>
<tr>
<td>sdfRef</td>
<td><code>tm:ref</code></td>
</tr>
<tr>
<td>sdfChoice</td>
<td>Enum of JSON objects with <code>sdf:choiceName</code></td>
</tr>
<tr>
<td>sdfRequired</td>
<td><code>tm:optional</code> (by including all non-required interaction affordance keys)</td>
</tr>
<tr>
<td>namespaces</td>
<td><code>@context</code></td>
</tr>
<tr>
<td>defaultNamespace</td>
<td><code>sdf:defaultNamespace</code></td>
</tr>
<tr>
<td>info</td>
<td>Multiple targets: <code>model</code> field in <code>Version</code> class</td>
</tr>
<tr>
<td>version</td>
<td><code>sdf:title</code></td>
</tr>
<tr>
<td>title</td>
<td><code>sdf:copyright</code></td>
</tr>
<tr>
<td>copyright</td>
<td>If URL: link with relation-type <code>license</code></td>
</tr>
<tr>
<td>license</td>
<td>Else: <code>sdf:license</code></td>
</tr>
</tbody>
</table>
### Table 2
Overview of mappings of the most important WoT classes and keywords [1, section 5] to SDF.

<table>
<thead>
<tr>
<th>WoT Class/Keyword</th>
<th>SDF Keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thing</td>
<td>sdfThing (TM has tm:submodel links), sdfObject</td>
</tr>
<tr>
<td>title</td>
<td>label</td>
</tr>
<tr>
<td>description</td>
<td>description</td>
</tr>
<tr>
<td>schemaDefinitions</td>
<td>sdfData</td>
</tr>
<tr>
<td>@context</td>
<td>namespaces of the SDF model (with exceptions)</td>
</tr>
<tr>
<td>DataSchema</td>
<td>dataqualities</td>
</tr>
<tr>
<td>readOnly</td>
<td>Mapping file</td>
</tr>
<tr>
<td>writeOnly</td>
<td>Mapping file</td>
</tr>
<tr>
<td>InteractionAffordance&lt;sup&gt;a&lt;/sup&gt;</td>
<td>—</td>
</tr>
<tr>
<td>title</td>
<td>label</td>
</tr>
<tr>
<td>description</td>
<td>description</td>
</tr>
<tr>
<td>PropertyAffordance</td>
<td>sdfProperty</td>
</tr>
<tr>
<td>readOnly</td>
<td>writable (negated)</td>
</tr>
<tr>
<td>writeOnly</td>
<td>readable (negated)</td>
</tr>
<tr>
<td>observable</td>
<td>observable</td>
</tr>
<tr>
<td>ActionAffordance</td>
<td>sdfAction</td>
</tr>
<tr>
<td>input</td>
<td>sdfInputData</td>
</tr>
<tr>
<td>output</td>
<td>sdfOutputData</td>
</tr>
<tr>
<td>EventAffordance</td>
<td>sdfEvent</td>
</tr>
<tr>
<td>tm:ref</td>
<td>sdfRef</td>
</tr>
<tr>
<td>tm:optional</td>
<td>sdfRequired (by including all non-optional interaction afford- ance keys)</td>
</tr>
<tr>
<td>Link</td>
<td>Mapping file, except for special link types (e.g., license, tm:extends, tm:submodel)</td>
</tr>
</tbody>
</table>

<sup>a</sup> This is the base class of the three affordance types.