Linked Data in the Web of Things

Gerald Haesendonck, Anastasia Dimou, Ben De Meester, Ruben Verborgh
IDLab, Dep. of Electronics and Information Systems,
Ghent University – imec, Zwijnaarde, Belgium

Introduction

The Internet of Things (IoT) consists of many devices and services producing or consuming data over a network, and, by extension, the internet. There are various protocols and data models used by different vendors of things, or middleware, to expose data and APIs to communicate with and consume data of things. The Web of Things (WoT) addresses IoT's fragmentation by forming a Web-based abstraction layer capable of interconnecting existing IoT platforms, devices, and cloud services and complementing available standards. Specifications of the Web of Things describe data and interaction models exposed to applications, and communication and security requirements for platforms to communicate effectively. At the core of the WoT specifications is the *Thing Description*, a semantic description of the data and interaction model(s) for a Thing. This helps other Things to perform actions on a Thing, e.g. read or write its *properties* (the data or *state* of a Thing). We believe that the WoT can benefit from semantically enhancing a Thing so it contains data values, or state, in the form of self-describing data. This allows powerful semantic processing and reasoning upon its state, and possibly the history of its states. To enable this, we propose a rule-based approach to generate self-describing data from a Thing's state(s).

The Thing Description: semantic metadata

A central building block in the Web of Things (WoT) is the Thing Description [1]. It consists of four elements. A first element is some metadata about the Thing, for instance its unique identifier, a name, a context to indicate used namespaces in the description, … . A second element describes a collection of affordances to interact with the Thing such as reading or writing properties, or triggering an operation manipulating state. A third element consists of schemas providing the necessary semantics for machine understandability. A final element is web links to express any relation to other Things, for example a lamp Thing contains a link to a switch Thing controlling the lamp. An application or other Thing that wants interaction with the Thing first consults the Thing Description. It then knows which protocol to use, which endpoint to contact and how to interpret and optionally manipulate the state of the Thing. The Thing Description uses information models and formats that are aligned with Linked Data for machine understandability, and offers extension points to include domain-specific vocabularies. This enables powerful semantic processing of the *metadata*. However, the *raw data* or state itself is not presented as Linked Data.

An example

To illustrate our approach, we introduce the following example. A certain lamp has a property *status* that consists of two values: a *level* that indicates the intensity of the light expressed as a number ranging from 0 to 255, and a *time* that represents the timestamp of the most recent measurement. The data of a single state of the lamp looks like the following snippet:

```
{
  "level": 50,
  "time" : "2017-06-06T12:36:13+00:00"
}
```

The corresponding Thing Description might look like this (some descriptions are left out for the sake of clarity):

```
{
 "@context": {
   "http": "http://iotschema.org/protocol/http",
   "iot": "http://iotschema.org/",
   "sosa": "www.w3.org/ns/sosa/"
 },
 "name": "MyLampThing",
 "id": "http://example.org/mylamp",
 …
 "properties": {
   "status" : {
     "type": "object",
     "properties": {
       "level": {
         "@type": ["iot:LevelData"],
         "type": "integer",
         "minimum": 0,
         "maximum": 255
       },
       "time": {
         "@type": ["sosa:ResultTime"],
         "type": "dateTime",
       }
     },
     "forms": […]
   }
 }
}
```

Towards Linked Data

We believe that in some cases more intelligent applications may be built if not only the metadata, but also the the state of a Thing is semantically enhanced. In our example, we could use the SOSA ontology, part of SSN [2], to represent the state of the lamp as Linked Data expressed in RDF [3] . Line numbers are added in front:

```
1  @prefix sosa: <http://www.w3.org/ns/sosa/>.
2  @prefix iot: <http://iotschema.org/>.
3  @prefix xsd: <http://www.w3.org/2001/XMLSchema#>.

4  <http://example.org/mylamp/status/2017-06-06T12%3A36%3A13%2B00%3A00>
5    a sosa:Observation ;
6    sosa:observedProperty <http://example.org/mylamp/status> ;
7    sosa:madeBySensor <http://example.org/mylamp> ;
8    sosa:hasResult [
9      a iot:LevelData;
```

```
10     iot:level "50"^^xsd:integer;
11   ];
12   sosa:resultTime "2017-06-06T12:36:13+00:00"^^xsd:dateTimeStamp .
```

The state is semantically enhanced, which affords the following:

i. Everything gets a globally unique identifier. Not only a state (line 4), but in this case also a sensor (line 7) or a property (line 6). This makes it possible to refer to individual states, sensors and properties and interlinking them so applications can get more context processing a state by following the links (Linked Data).
ii. The state/data becomes self-describing. In this example, by reading the state, we can see that it is an *observation* (line 5), that it concerns the property *status* (line 6) of a sensor with ID *http://example.org/mylamp* (line 7). The data itself is the *integer 50* (line 10) of which the exact interpretation is given by the *iot:LevelData* object (line 9). Things or applications reading this data do not need to know about the WoT in order to process it.
iii. A plethora of tools exist in the Semantic (Web) Technologies domain to process the states. For example, if we want to know when the lamp in our example had brightness level 50, we can simply issue a SPARQL query [4] without transforming the data to fit into some database. A more advanced application is for instance root cause analysis, where performing semantic reasoning over states of Things (e.g. sensors) can detect the cause of the failure of a certain Thing [5].

Rule Based Linked Data Generation

We propose a rule-based approach to generate Linked Data, represented in RDF, from Web of Things data. This way, we distinguish the rules from the implementation. Namely, the rules can be processed without affecting the implementation and a proper implementation may be chosen depending on the use case. Templates for the proposed models may be published and reused and automatically applied to different Things. Our proposed approach relies on the YARRRML language [6] to declare the rules and the RML.io tool stack, and in particular the RMLStreamer, to support defining and executing the rules [7][8].
The RMLStreamer executes rules to generate Linked Data from other data sources, like the state of Things. It might be deployed for instance on a *servient* (server role), or on middleware, so that it can be integrated in a Thing. The component can generate Linked Data from static data pulled from a source, or from streaming data pushed by a source. This fits in the action- or event-based interaction model of the Web oThings.
The rules are expressed in YARRRML, a human readable text-based representation, in YAML, for declarative generation rules. The rules are used to generate Linked Data (RDF) from (semi) structured data using a given ontology, such as the one(s) used in the Thing Description. If the data model of a Thing changes, then the rules need to be updated accordingly, but this requires no software modifications.
In our example, the rules to generate the RDF representation from a state in simple JSON, expressed in YARRRML look as follows:

```
prefixes:
  iot : "http://iotschema.org/"
  sosa: "http://www.w3.org/ns/sosa/"
  xsd : "http://www.w3.org/2001/XMLSchema#"
```

```
mappings:
  lamp:
    sources:
      - ['state.json~jsonpath', '$']
    s: http://example.org/mylamp/status/$(time)
    po:
      - [a, sosa:Observation]
      - [sosa:observedProperty, "status"]
      - p: sosa:hasResult
        o:
        - mapping: results
  results:
    sources:
      - ['state.json~jsonpath', '$']
    po:
      - [a, iot:LevelData]
      - [iot:level, $(level), xsd:dateTimeStamp]
```

To expose a Thing's state as Linked Data to other Things, only a small description has to be added to a Thing Description. A new Form can be added to an existing property in the Thing Description. In the case of action-based interaction with the Thing, the form has a readproperty operation, and in the case of event-based interaction with the Thing, the form has an observeproperty. Other operations are not relevant because it's only about reading the state of a Thing. The contentType is a Linked Data media type, such as *application/ld+json*, *application/rdf+xml*, or *text/turtle*.
In our example, the Thing Description gets an extra form (marked in bold), and looks like:

```
{
  "@context": {
    "http": "http://iotschema.org/protocol/http",
    "iot": "http://iotschema.org/",
    "sosa": "www.w3.org/ns/sosa/"
  },
  "name": "MyLampThing",
  "id": "http://example.org/mylamp",
  …
  "properties": {
    "status" : {
      "type": "object",
      "properties": {
        "level": {
          "@type": ["iot:LevelData"],
          "type": "integer",
          "minimum": 0,
          "maximum": 255
        },
        "time": {
          "@type": ["sosa:ResultTime"],
          "type": "dateTime",
        }
      },
      "forms": [
        {
          "href": "http://example.org/mylamp/status/rdf",
```

```
          "htv:methodName": "GET",
          "op": "readproperty",
          "response": {
            "contentType": "text/turtle"
          }
        },
        …
      ]
    }
  }
}
```

Conclusions

In this document, we show that representing the state of a Thing as Linked Data, besides the original raw data, can be useful to enhance semantic processing of a Thing's data. We propose a rule-based approach that can be plugged into the WoT architecture, without altering the current WoT specifications. The solution fits in the interaction model of a Thing, which can be action-based or event-based. Distinguishing rules from the underlying implementation ensures that the rules can be processed without affecting the implementation and a proper implementation may be chosen depending on the use case. The solution fits in the interaction model of a Thing, which can be action-based or event-based.

Bibliography

1: Kaebisch, S., Kamiya, T., McCool, M., Charpenay, V., Web of Things Thing Description, W3C Editor's Draft, 2019, https://w3c.github.io/wot-thing-description/

2: Haller, A., Janowicz, K., Cox, S., Le Phuoc, D., Taylor, K., Lefrançois, M., Semantic Sensor Network Ontology, W3C Editor's Draft, 2019, https://w3c.github.io/sdw/ssn/

3: Cyganiak, R., Wood, D., Lanthaler, M., RDF 1.1 Concepts and AbstractSyntax, 2014, http://www.w3.org/TR/rdf11-concepts/

4: Harris, S., Seaborne, A., SPARQL 1.1 Query Language. Recommendation, 2013, https://www.w3.org/TR/sparql11-query/

5: Steenwinckel, B., Heyvaert, P., De Paepe, D., Janssens, O., Vanden Hautte, S., Dimou, A., De Turck, F., Van Hoecke, S., Ongenae, F., Towards adaptive anomaly detection and root cause analysis by automated extraction of knowledge from risk analyses, 2018

6: Heyvaert, P., De Meester, B., Dimou, A., Verborgh, R., Declarative Rules for Linked Data Generation at your Fingertips!, 2018

7: Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van deWalle, R., RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data, 2014

8: Haesendonck, G., Heyvaert, P., Vanden Hautte, S., Dahdah, K., Dimou, A., Van Hoecke, S., Verborgh, R., Workshop on Linked Data on the Web, 2019