SparTDD -a SPARQL based Thing Description Directory

Christian Glomb, Élodie Thiéblin, Fabien Amarger

SparTDD - a SPARQL based Thing Description Directory

Christian Glomb[1], Élodie Thiéblin[2], and Fabien Amarger[2]

[1] Siemens Technology, Munich, Germany
firstname.lastname@siemens.com
[2] Logilab, Toulouse, France
firstname.lastname@logilab.fr

Abstract. Logilab and Siemens collaborated on the implementation of a Thing Description Directory based on a SPARQL endpoint. The Thing Description (TD), as standardized by the World Wide Web Consortium (W3C), is a JSON-LD based document comprising information about connectivity, security, and semantics of Internet of Things devices or services. TDs represent RDF triples using a defined ontology and JSON-LD context. A Thing Description Directory (TDD) is a standardized endpoint in which we can store, update and retrieve TDs. A TDD based on a SPARQL endpoint is the best way to get the most out of the potential of Semantic Web technologies combined with Thing Descriptions. This article presents SparTDD, an implementation of a SPARQL-based TDD and its future applications in the industrial digital twin context.

Keywords: Web of Things · Thing Description · Thing Description Directory · SPARQL · Asset Administration Shell

1 Context

W3C Web of Things (WoT)[3] standardization seeks to counter the fragmentation of the IoT by using and extending existing, standardized Web technologies. By providing standardized metadata and other re-usable technological building blocks, W3C WoT enables easy integration across IoT platforms and application domains. Building blocks include Thing Description (TD), Binding Templates, Scripting API, and Discovery. The TD, which is the central building block, is formatted as a JSON-LD[7] document - JSON (JavaScript Object Notation) to be attractive for web developers, LD (Linked Data) to leverage the advantages of Semantic Web technologies providing standard vocabularies to improve the inter-operability of Things in the IoT.

1.1 Thing Description (TD)

The Thing Description[3] (TD) is a formal model defined by the W3C to describe the metadata and interfaces of Things in the IoT [4]. A Thing is an abstraction of

[3] https://www.w3.org/WoT/

a physical or virtual entity that provides interactions to and participates in the Web of Things. The vocabulary is divided in two parts, the formal ontology to define the semantic model and a JSON-LD context to define the representation model. A TD details the interaction affordances of the described thing:

properties to define the main properties observed by the Thing (e.g. a temperature value)

actions to define the interactions that the Thing offers (e.g. toggle a heater on/off)

events to define events which are observable (e.g. overheating)

The interaction affordances comprise a forms field detailing how to communicate with the Thing (e.g., description of the REST API URL to toggle the heater on/off). They bridge between the WoT abstraction layer and the actual communication protocol by invoking a Binding Template. Furthermore, there is also a description of the data format to be expected or to be served (e.g., the event "overheat" is defined using an array of temperatures with the last 5-min temperatures). A TD also has to describe two other important sections:

security to define the security mechanism to be used to communicate with the Thing (e.g., basic authentication)

links to define how this Thing is related to other Things (e.g., switch can turn the heater on/off) - this section is of special interest for SPARQL queries traversing the Things hierarchy

A TD can be expressed in RDF along with the Thing Description ontology. Since the WoT TD specification defines a JSON-LD based representation format, where JSON-LD is valid JSON, TDs can be handled by tools which do not necessarily deal with semantic web technologies. The conversion between the RDF description and the JSON representation format is possible through a JSON-LD context.

1.2 Thing Description Directory (TDD)

While, the purpose of the TD ontology is to standardize the description of Things in the WoT, WoT Discovery [4] aims at making the TDs findable and retrievable.

The architecture of WoT Discovery process defines a two-phase approach:

Introduction It describes ways how either Things provide a link to their description as a TD or how a TDD hosting multiple TDs can be found by other Things. Instead of inventing new mechanisms, the usage of existing functionality, like Bluetooth beacons, Domain Name System Based Service Discovery, or hand-copied URLs is motivated.

Exploration This mechanism introduces Thing Description Directories (TDDs) as entrypoints to a set of registered TDs. The TDDs implement an RESTful HTTP API enabling the CRUDL operations (create, read, update, delete, and list) on the TDs, as well as a search API over them. SparTDD is our experimental implementation of the Exploration phase.

In a TDD, TDs must be sent and retrieved in their JSON-schema-compliant serialization and the data should round-trip through the TDD with no modification. A TDD may implement a search API in SPARQL, XPath, and/or JSONPath.

[4] `https://www.w3.org/TR/wot-discovery/`

2 SparTDD: A SPARQL-based TDD

The first TDD implementation [2] was delivered before W3C started discussing about Discovery. It only provides a raw SPARQL interface without the possibility of TD roundtripping and does not follow the current TD standard. ([9] and [6]) are newer implementations using JSON-based database engines, and [5] is a library for TD handling allowing to implement a custom TDD. However, to gain a better understanding about SPARQL endpoints in the TDD context, we decided to start from scratch.

This is why we propose **SparTDD** where the objectives are twofold: The first one is to prove that we can use a SPARQL endpoint to store data in a TDD. The second objective is to determine how the Semantic Web technologies (especially SPARQL queries, ontology formalization and reasoning) can help to improve the W3C TD and TDD standards. Figure 1 presents the global architecture of our implementation. The TDD is a server implementing the WoT Discovery API[5]. It handles the JSON-LD to RDF conversion and the JSON-LD framing on the TDs.

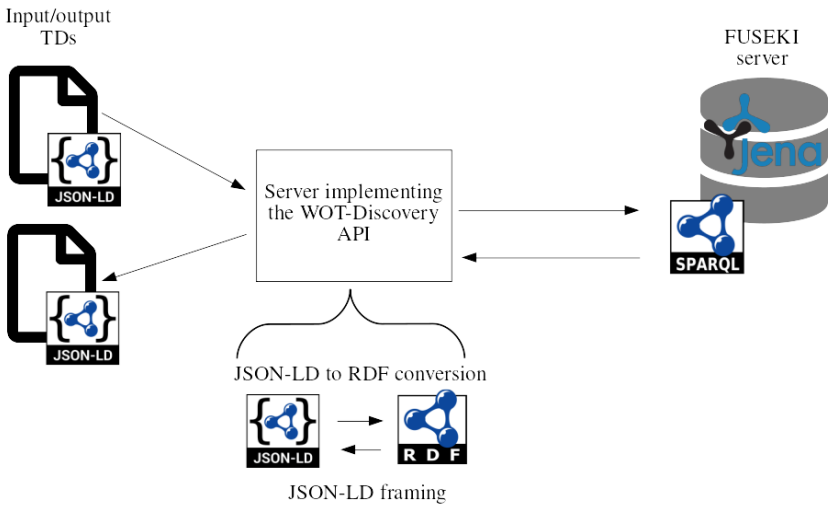

Fig. 1. Global architecture of our SPARQL-based TDD

This implementation can be used to upload Thing Descriptions (using JSON-LD format using the TD context[6] or RDF triples using TD ontology[7]. A SPARQL route is proposed, as described in the TDD standard, and it uses the SPARQL endpoint service to get the responses.

[5] `https://www.w3.org/TR/wot-discovery/#exploration-directory-api`
[6] `https://raw.githubusercontent.com/w3c/wot-thing-description/main/context/td-context-1.1.jsonld`
[7] `https://github.com/w3c/wot-thing-description/blob/main/ontology/td.ttl`

We can easily use the SPARQL endpoint server to run a reasoner and profit from the TD ontology axioms to deduce new RDF triples. This is still an ongoing issue that we want to explore in the future.

SparTDD is a proof of concept to study how a TDD can be implemented using semantic web technologies. For now, it does not handle security like authentication, etc. Only SPARQL search can be performed on the TDD. XPath and JSONPath search are not implemented, and it would not be straightforward to deal with them. All implemented features of SparTDD can be found in the implementation report[8] and some usage example are listed in the Discovery specification. Use cases about WoT in general and WoT Discovery in particular are collected by the WoT members[9].

The SparTDD implementation allows us to further investigate how semantic web technologies can help in the context of Things Discovery in the Web of Things. With this in mind, we plan to test reasoning in SparTDD and are also testing use-cases needing a SPARQL interrogation where other query languages like JSONPath are not expressive enough. Examples include querying TDs that are inter-linked over multiple levels or relating dynamic information coming from other API endpoints according to the metadata stored in the TDs.

3 Future Applications - Industrial Digital Twin context

While the application of semantic web technologies in the WoT context is already interesting and challenging enough, we think about broadening the scope and come from pure IoT scenarios, where the information carried in the TD matters, to scenarios where we are dealing with multiple layers of hierarchies and value propositions. For these scenarios, the German ZVEI association has defined the RAMI4.0 model[10] reflecting the different scopes and views in the industrial context (Fig. 2). Along with this model, the German IDTA (Industrial Digital Twin Association) aims to target the value creation perspective via a cross-manufacturer exchange of information with industry-neutral standards for communication, services, and semantics.

The most important tool for this is the digital twin, the data image of an asset. An asset is an object that is to be integrated into the information world of the industrial context. The range of possible assets is wide: it extends from machines and their components, supplier material and products to software and documents such as plans, contracts or orders, and contracts. For IDTA, The digital twin is primarily not a virtual, reality-like image of the object under consideration, but is rather considered a standardized connector that works in multiple applications across manufacturers.

[8] https://github.com/w3c/wot-testing/blob/main/events/2022.03.Online/Discovery/Results/logilabtdd.csv

[9] https://w3c.github.io/wot-usecases/#Discovery

[10] https://www.zvei.org/en/subjects/industry-4-0/the-reference-architectural-model-rami-40-and-the-industrie-40-component

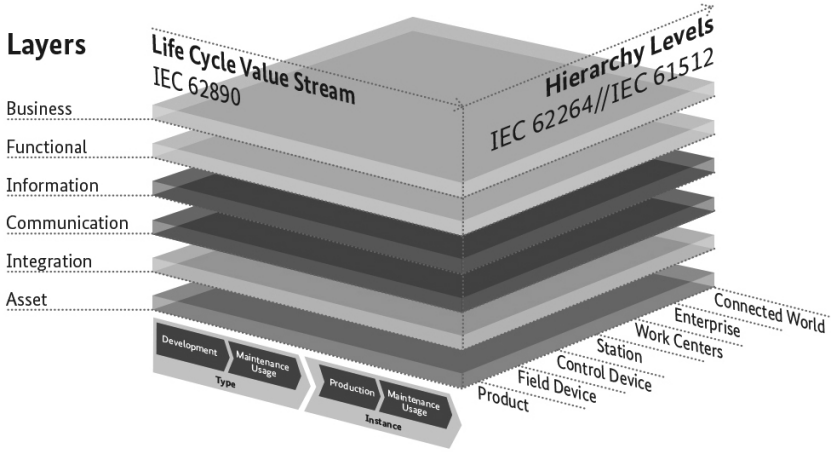

Fig. 2. The Reference Architectural Model Industry 4.0

The digital twin contains all the information that characterize the features and behaviors of an asset. The **Asset Administration Shell (AAS)** designed by IDTA implements the digital twin for the industrial context. Through complete interoperability, the AAS should pave the way to higher digital value creation. It simplifies data management equally for non-smart and smart devices. The AAS maps the entire life-cycle of products, devices, machines, and plants.

The AAS consists of a set of sub-models that describe all the data and functionalities of a given asset - such as characteristics, properties, states, parameters, measurement data, and capabilities. The AAS enables the use of different communication channels as well as applications and connects objects to the networked, digital and decentralized world.

Three types of the AAS are defined: Passive AAS (type 1) is a compressed file containing the sub-models in a given folder structure; Passive AAS with API (type 2) can be implemented as a micro-service giving access to the sub-models via a defined API; Active AAS (type 3) extends type 2 by adding computational as well as communication components such that the AAS can act autonomously in appropriate environments. According to [1] the certain AAS types fulfill certain layers of the RAMI4.0 model (see Table 1).

RAMI4.0 Layers / AAS Types			
	Type 1	Type 2	Type 3
Business Layer			X
Functional Layer		X	X
Information Layer	X	X	X
Communication Layer		X	X
Integration Layer	X	X	X
Asset Layer	X	X	X

Table 1. RAMI4.0 layers implemented by AAS types

AAS type 3 can comprise communication components, where one communication channel might relate to the interaction with an asset or service as described by the AAS. That sounds similar to what a TD is designed for, such that we could imagine having a TD like sub-model responsible for such task.

Asset Interface Description (AID)[8] is an approach to standardize such sub-model. Although the discussion just started, we expect this sub-model to be standardized in the near future, opening the AAS door towards the mechanisms and lessons learned in WoT. This includes everything we learned from WoT discovery, such that we are looking forward to exciting new application fields for SparTDD.

Testing SparTDD on real data, helped us contribute to the W3C WoT Thing Description and Discovery specification, especially with respect to the JSON-LD context, and to learn how to handle concrete discovery tasks. With AAS and the upcoming sub-model AID, new use-cases will be investigated for SparTDD and may raise new issues that semantic web technologies could solve.

References

1. Belyaev, A., Diedrich, C.: Specification "demonstrator i4.0-language" v3.0 (07 2019)
2. Charpenay, V.: Semantics for the Web of Things: Modeling the Physical World as a Collection of Things and Reasoning with their Descriptions. Ph.D. thesis, Universität Passau (2019)
3. Charpenay, V., Käbisch, S.: On modeling the physical world as a collection of things: The w3c thing description ontology. In: European Semantic Web Conference. pp. 599–615. Springer (2020)
4. Charpenay, V., Käbisch, S., Kosch, H.: Introducing thing descriptions and interactions: An ontology for the web of things. In: SR+ SWIT@ ISWC. pp. 55–66 (2016)
5. Cimmino, A., García Castro, R.: Java api for thing descriptions of wot (2021)
6. Costanzi, E., Aguzzi, C., Gigli, L., Di Felice, M.: Unibo-prismlab/thingdescriptiondirectory (2021)
7. Kellogg, G., Champin, P.A., Longley, D.: JSON-LD 1.1–A JSON-based Serialization for Linked Data. Ph.D. thesis, W3C (2019)
8. Pakala, H.K., Oladipupo, K.O., Käbisch, S., Diedrich, C.: Integration of asset administration shell and web of things (2021)
9. Tavakolizadeh, F., Devasya, S.: Thing directory: Simple and lightweight registry of iot device metadata. Journal of Open Source Software **6**(60), 3075 (2021)