Semantic Discovery in the Web of Things

Fernando Serena, María Poveda-Villalón, and Raúl García-Castro*

Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
{fserena,mpoveda,rgarcia}@fi.upm.es

Abstract. While the number of things present in the Web grows, the ability of discovering such things in order to successfully interact with them becomes a challenge, mainly due to heterogeneity. The contribution of this paper is two-fold. First, an ontology-based approach to leverage web things discovery that is transparent to the syntax, protocols and formats used in things interfaces is described. Second, a semantic model for describing web things and how to extract and understand the relevant information for discovery is proposed.

Keywords: discovery, interoperability, web of things, ontologies

1 Introduction

The Internet of Things is characterized by its inherent heterogeneity [1], which is evident when thinking about the diversity of things that can be accessed through the Internet. Such diversity does not apply only to the type of things, e.g., thermostat, traffic light; but also to many other aspects like their communication protocols, data formats and even the IoT standards [2] they implement. Furthermore, the range of possibilities for the aforementioned aspects are not expected to stop growing, so one can say that heterogeneity can only evolve.

The number of various things that are being made available through the Internet is growing steadily[1]. Therefore, IoT consumers cannot be asked to be aware of every possible aspect, platforms and individual things out there, so that it is necessary to rely on mechanisms and services that enable them to search for and discover what they want to consume. In other words, discovery is meant to cope and take advantage of the heterogeneity and large population of things in the IoT. For example, discovery is one of the Common Service Functions of the architecture proposed by the oneM2M standarization organization[2].

A detailed description of the open issues in discovery for the so-called Web of Things (WoT) is provided in Section 2, including a characterization of web

* This research is partially supported by the European project VICINITY: Open virtual neighbourhood network to connect intelligent buildings and smart objects (H2020-688467) http://vicinity2020.eu/vicinity/

[1] http://www.businessinsider.com/there-will-be-34-billion-iot-devices-installed-on-earth-by-2020-2016-5

[2] http://www.onem2m.org/technical/published-documents

things based on the current discussion of the corresponding W3C Working Group (WG)[3], and outlining a desirable common data model for thing descriptions.

A particular approach for semantic discovery of web things, their interaction patterns and attributes regardless of specific communication protocols, syntax and data formats used in their web interfaces is provided as first contribution of this paper and it is described in Section 3. A use case that implements the proposed approach is also provided.

Since such approach is ontology-based, an overview of the developed ontologies for supporting the proposed solution is presented in Section 4. This description framework for the Web of Things in the form of an ontology which answers to what a thing is, where in the Web are its interfaces and how to extract and understand the discovery-relevant information from it represents the second contribution of the present work.

Existing approaches for WoT discoverability and related semantic models are outlined in Section 5 before concluding and discussing future work in Section 6.

2 Discovery in the Web of Things

Discovery in the IoT can be thought of as search on web pages: consumers issue some search criteria that can result in the discovery of a set of resources relevant to the consumer and possibly yet unknown. Although, inevitably, in order to yield useful results, it is necessary to have the means to characterize or describe resources so that potential matches with the provided search criteria can be detected. As it is well known, search engines work with documents and count on standards like HTML. Nevertheless, discovery services in the IoT are still orphaned in common formats and syntax for that purpose in a global scope.

Therefore, a common data model for describing things, their features and capabilities is required. In this sense, the W3C Web of Things WG is already working on defining and standardizing a Thing Description (TD) data model.

2.1 Web things

The Web of Things (WoT) aims to make everything that belongs to an IoT ecosystem (e.g., devices, systems, or data) part of the Web, leveraging it as a platform. Thus, all individual things accessible through a web interface may belong to the WoT. Still, along with the description of the features and capabilities of these web things, their web interfaces have to be described as well.

At discovery time, web things intertwine what has traditionally been separated: resource and service discovery [3]. Discovery clients are not only interested in what web things are but also in where and how to reach them out on the Web. Regarding the "where", descriptions must inform about the corresponding dereferenceable links for accessing the thing; regarding the "how", descriptions should include relevant metadata that report on aspects to be taken into account when invoking each link, e.g., communication protocol, data formats and

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

security constraints. In this way, descriptions of discovered things may provide complete "views" that can lead consumers to implement informed interactions, even automatically, and bringing a minimum support to IoT interoperability.

2.2 Description scope

Not all thing attributes that are relevant for discovery can be expressed in a static and shareable description; mainly because they are dynamic, protected or both. For instance, the geo-location of certain physical things can be considered as sensitive and only be obtained under specific security and privacy constraints, through its endpoints. Besides, its value may dynamically vary as the physical thing changes its position. Therefore, if this casuistry is not taken into account, the location-based discovery of such kind of things will not be possible.

A solution to this would involve describing as well how data provided by secured endpoints map to specific thing attributes. By following this approach, descriptions might inform on how to automatically and securely retrieve and map their own missing attribute values, by means of what we call *access mappings*.

2.3 Access mappings

The adoption of *access mappings* in the data model for web things leads to a wider scope solution: to gather values for any kind of thing attributes from its own web interfaces, significantly extending the support to interoperability in the IoT ecosystem. In order to achieve this, data models for web things should also support describing the exchanged data with mentioned links or endpoints, i.e., they should not just describe its format but also its content. Thus, rather than expecting to receive data from endpoints in a specific syntax, descriptions would inform consumers on how to process responses and extract useful information.

For instance, the description of a temperature sensor may tell that the data received after invoking an endpoint contains the latest measured value in Celsius and where to find such value in the response. Thanks to this, discovery clients might be able to issue search criteria for things measuring temperature in Celsius and get, extract and interpret values from the discovered things' endpoints.

3 Ontology-based Approach for Discovery

Having a common data model turns out to be the cornerstone of interoperability in the Web of Things. Besides, the richer the model, the more interoperable things will be. Still, increasing the richness does not necessarily mean that the model has to be more complex, but rather better represent the ecosystem. Even having the richest model imaginable shared by all actors of the ecosystem is not sufficient to facilitate discovery. It is also desirable that approaches for discovery in WoT meet the following requirements:

- Depending on the context and use case, at least one of the following interaction patterns[4] for discovery should be supported: a) finding things around

[4] `https://www.w3.org/WoT/IG/wiki/Discovery_Categories_and_Tech_Landscape`

spatial coordinates, b) finding things on a network, c) searching in directories, and d) accessing thing metadata.
- Communication technologies used by things should not condition the process of discovery. They just have to be properly described in case there are endpoints that implement them.
- Requests should be expressed as queries based on the common data model.
- The model for describing things must be agnostic of whatever discovery interaction patterns are required in the use case; web things are what they are regardless of the mechanisms implemented to discover them.

In any case, all approaches for discovery in the WoT would mainly build on the ability of involved actors to generate, publish, understand and query thing descriptions. Not all actors of the WoT need to have all these abilities; it shall depend on the role each one plays in the ecosystem, e.g, consumers are not required to have the same abilities as publishers. In what follows, descriptions of the aforementioned abilities as well as their impact on the different approaches for discovery are provided.

Generate. Descriptions of things provided by web interfaces need to be generated so as to become part of the WoT and, in turn, to be interoperable in the ecosystem. Such generation may be performed manually, e.g., by the owner of the web thing; or automatically, by a hypothetical system capable of characterizing web things. Such a task of automatically describing the type of a certain thing plus some of its attributes and features might seem to be not too complicated if sufficient metadata about it can be obtained automatically as well, e.g., HATEOAS, CoRE Link Format. However, it is not the case of automatic description of their web interfaces, even if they were built using standards like OData[5], OpenAPI[6] or RAML[7].

Publish. Once a thing description is generated it has to be made available in a machine-readable format so that others can eventually consume it. Intuitively, there are at least two ways of publishing these descriptions: a) the actual thing directly exposes its own description through an endpoint, b) a third-party entity is given thing descriptions so as to be the directory of the ecosystem. Choosing one or the other may significantly affect the architectural approach to address the discovery problem, e.g., the former may be crucial in peer-to-peer solutions.

Understand. In order for actors to interact with a discovered thing, they must be able to read, parse and understand its description. Thus, it is required to have a well-defined set of predefined serialization formats and common syntax shared by all actors. The usage of the common data model in descriptions is what shall enable actors to understand such descriptions and to parse them correctly.

[5] http://www.hydra-cg.com/spec/latest/core/
[6] https://openapis.org/
[7] http://raml.org/

Query. Search criteria can be considered as the queries that trigger the discovery process which is expected to end up providing a ranked set of matching thing descriptions to the issuer. In case a query language, e.g., SPARQL[8] is used for expressing semantic search criteria, all involved actors should at least know about its protocol, i.e., communications established between an issuer and a directory must implement the corresponding query language protocol. A high level of expressiveness of the query language used will likely broaden the possibilities for discovery requests, e.g., features for filtering and aggregating. However, its counterpart is that involved actors might need to be much more intelligent.

3.1 Semantic discovery

Although all the aforementioned requirements and considerations are met and covered in a hypothetical Web of Things, there is still room for improvement in discovery and, in turn, in interoperability in general. In addition to the described abilities, actors can infer implicit information from thing descriptions by leveraging the semantics of the ecosystem by means of reasoning.

Those ecosystems that promote reasoning will allow actors of the discovery process to query about things whose specific type may be not known to the issuer, but its abstract type is. Further, some discovery queries may express interest in things that measure humidity in general, no matter if it is relative or absolute. In both cases, it is the explicit definition of semantics into the common data model what supports the ability to reason within the whole ecosystem. In this paper, we propose an approach for discovery that takes advantage of domain semantics and implements its common data model in the form of an ontology.

In Computer Science, the term ontology is used to refer to a “formal, explicit specification of a shared conceptualisation” [4]. First of all, “share” reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group. In our case, the shared conceptualization to be represented in this ontology is the domain of WoT being discussed in the W3C WoT WG. For doing so, it is needed to describe the concepts, properties and constraints that apply to the given domain. All those entities need to be described explicitly, so that we cover as much as possible of the world phenomenon that we are trying to represent. Next, being formal refers to the need of implementing the ontology following a machine-readable ontology language that can be easily processed. For doing so, the proposed ontologies are formalized following Description Logics and being implemented in the W3C Web Ontology Language standard OWL.[9]

3.2 Use case: VICINITY

VICINITY is an H2020 European project that aims to be an open virtual neighbourhood to connect IoT infrastructures. In VICINITY, a peer-to-peer

[8] https://www.w3.org/TR/sparql11-query/
[9] https://www.w3.org/TR/owl-ref/

network is created, which is composed of IoT infrastructures and value-added services that are integrated so as to become semantically interoperable, while IoT providers can keep control over their data. All nodes in the network share a common data model, namely the VICINITY ontology network[10].

The VICINITY architecture implements the present ontology-based approach for discovery in the WoT as follows:

- Any node of the network can issue discovery requests in the form of SPARQL queries expressed with the VICINITY ontology.
- Queries are issued to a cloud-based central directory that holds descriptions of all things in the VICINITY ecosystem.
- Nodes wanting to integrate their assets into VICINITY are responsible for sending their corresponding descriptions to the central directory.
- For each SPARQL query that the central repository receives, it returns a set of query-relevant thing descriptions to the issuer.
- As nodes receive thing descriptions as query results, they still have to figure out whether such descriptions contain all query-relevant data that allows them to make an informed evaluation of its own discovery query. As we mentioned in section 2.2, there might be cases for which some additional information has to be collected at discovery time, e.g., geo-location.
- In case some extra information is needed for discovery, issuer nodes (securely) invoke the query-relevant remote endpoints, taking advantage of provided access mappings in descriptions so as to correctly understand responses.
- Nodes evaluate the SPARQL query and yield the found things.

As just described, the discovery process in VICINITY is distributed. Nodes rely on the central directory to provide them with as much query-relevant information as it knows, but both extension of description scope and SPARQL query evaluation take place at client side.

4 Semantics for understanding Thing Descriptions

As presented above, the proposed approach for discovery in the WoT is based on the semantic description of web things and of the way of accessing them. In order to fill in the gap for this goal some ontologies are being developed. The following sections will briefly present such ontologies, namely the WoT ontology (describing "what", "where" and "how" they are accessed) and a Mapping ontology (describing the how the information should be understood).

For the sake of readability, prefixes will be used for representing ontology namespaces along the section, including text and figures. The list of prefixes used and their corresponding ontologies and namespaces are listed in Table 1.

4.1 WoT ontology

This section provides an overview of the WoT ontology as well as a brief description of the development process and resources available.

[10] http://vicinity.iot.linkeddata.es/vicinity/

Table 1. Prefix listing of References Ontologies

Prefix	Name	URI
wot	WoT ontology	http://iot.linkeddata.es/def/wot#
map	Mapping ontology	http://iot.linkeddata.es/def/wot-mappings#
core	VICINITY core ontology	http://iot.linkeddata.es/def/core#
om	Ontology of units of Measure (OM) 1.8	http://www.wurvoc.org/vocabularies/om-1.8/
rdf	RDF model	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	rdf(s)	http://www.w3.org/2000/01/rdf-schema#
owl	OWL ontology	http://www.w3.org/2002/07/owl#

The WoT ontology has been developed to define "what", "where" and "how" (see Section 2) things can be discovered or accessed in the Web of Things. In this sense, the shared conceptualization to be represented in this ontology is the domain of the Web of Things, that is, it will describe the virtual counterpart of physical objects according to the Web Thing Model discussed in the W3C.

As most engineering projects, the ontology development life cycle usually starts with the Ontology Requirements Specification activity. In order to extract requirements for the WoT ontology, the documentation provided by the WoT Interest Group (IG) of the W3C,[11] was analysed.While the complete list of requirements initially defined for the WoT ontology are available online[12], in the following an excerpt of the main requirements is provided in the form of competency questions [5] or natural language sentences:

1. What is a thing in the web thing context? The abstract concept of a physical entity that can either be a real-world artefact, such as a device, or a virtual entity that represents physicality, such as a room or group of devices.
2. What is a property? A property provides readable and/or writeable data that can be static (e.g., supported mode, rated output voltage, etc.) or dynamic (e.g., current fill level of water, minimum recorded temperature, etc.).
3. What is an action? The Action interaction pattern targets changes or processes on a Thing that take a certain time to complete.
4. What is an event? The Event interaction pattern enables a mechanism to be notified by a Thing on a certain condition.
5. A thing interaction can be available over different or multiple protocols.
6. Each thing has at least an interaction pattern.
7. An interaction pattern can have different endpoints.
8. Each interaction pattern has an endpoint.
9. Each endpoint has minimum two attributes: URI and media-type.
10. An endpoint can be associated with a thing without determine the interaction patterns.

After defining the first set of requirements, though modification and addition of requirements is allowed during the development, the ontology implementation phase has been carried out through a number of sprints in which some requirements are selected in order to be incorporated in the current model. The current

[11] http://w3c.github.io/wot/current-practices/wot-practices

[12] http://vicinity.iot.linkeddata.es/vicinity/requirements/wot/report.html

conceptual model defined by the WoT ontology is depicted in Figure 1. This ontology introduces some new concepts closely related to the WoT domain, namely:

- *Thing*: this concept represents anything (both physical and non-physical) which has a distinct and independent existence and can have one or more web representations.
- *Interaction pattern*: this concept represents, in the context of WoT, an exchange of data between a web client and a Thing. This data can be either given as input by the client, returned as output by the Thing or both.
- *Data format*: this concept represents the input data or output data of a given interaction pattern which includes information such as the data type used and which unit of measurement is the data represented in, if needed.
- *Endpoint*: this concept indicates the web location where a service can be accessed by a client application.

The main concepts defined in the ontology, as shown in Figure 1, are `wot:Thing`, `wot:InteractionPattern`, `wot:DataFormat` and `wot:Endpoint` according to the above definitions. It is worth noting that the class `wot:Thing` defines things in the context of the Web of Things and does not intend to be the top class of all possible concepts as `owl:Thing` does. According to the model, a particular thing is linked to the interaction patterns it provides by means of the object property `wot:providesInteractionPattern`. An interaction pattern can be either a *property*, an *action* or an *event*, represented by the concepts `wot:Property`, `wot:Action` and `wot:Event`, respectively.

As shown in Figure 1, a thing or an interaction pattern can be associated to one or more endpoints either directly or through its interaction patterns by means of the object property `wot:isAccessibleThrough`. The main information provided by the endpoint class is about the web location in which the service is provided which is indicated by the attribute `wot:href`. Every endpoint should have a value and only one value for such attribute. Attached to such endpoint the information about the expected media type can be specified by means of the property `wot:isProvidedOverProtocol` which links instances of endpoints to the individual that represents the possible web protocols.

Finally, some interaction might have input or output data associated, or both, for example for writable properties. In order to model that, the relationships `wot:hasInputData` and `wot:hasOutputData` were created. These properties allow the connection from a given interaction pattern to an instance that will be linked to a certain data type and a certain unit of measure by means of the properties `wot:isMeasuredIn` and `wot:hasValuetype`, respectively. This modelling decision responds to the use of the ontology design pattern for representing n-ary relationships as it is needed to relate the given interaction patterns with both the unit of measure and the expected data type.

It should be mentioned that the presented ontology is under development and new concepts might be included or extended. Some ongoing lines of work on the ontology include the modelling of more complex datatypes, to detail security aspects, and to further describe the actions and events as they are defined in the W3C working group.

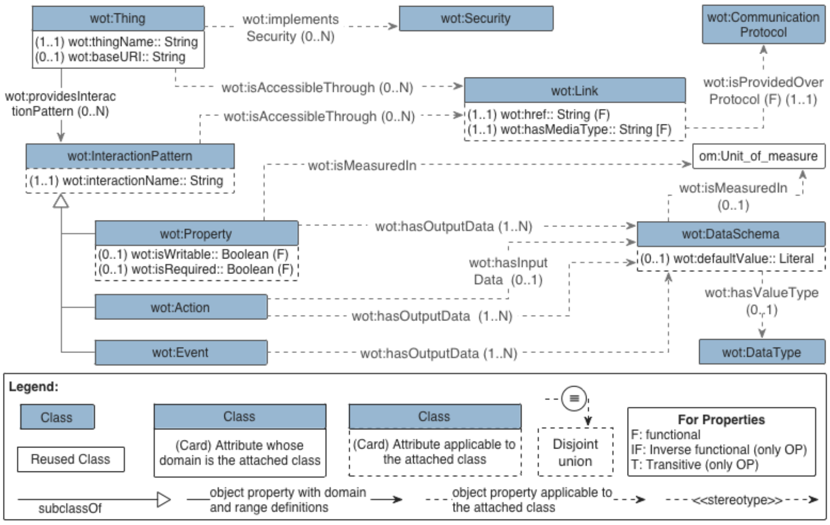

Fig. 1. General overview of the WoT ontology

4.2 Mapping ontology

Additionally to the WoT ontology, another model for describing how thing values should be understood (see Section 2) has been developed. The conceptualization to be represented in this ontology is the mechanism for accessing the values provided by web things. In this sense what is needed is to represent the mappings between the values provided under a given endpoint for example in JSON format to common semantic vocabularies.

The current conceptual model defined by the Mapping ontology is depicted in Figure 2. In order to model this information, it should be first established what does a mapping mean in this context:

– *Mapping*: A mapping indicates the relation between a given key (provided as structure data in an on-line resource) and the RDF property to which the values should be mapped and the target type of object.

Taking this definition as starting point and together with sample data, the ontology shown in Figure 2 was designed. The main concepts defined in such ontology are `map:Mapping` and `map:AccessMapping`. The former correspond to the mapping concept above-defined allowing the connection between a key provided within structure data in an on-line resource, represented by the datatype property `map:key`, to the RDF property to which it should be mapped, represented by the object property `map:predicate`.

The instances of the class `map:Mapping` can be further classified into one of its two subclasses, `map:ObjectPropertyMapping` and `map:DatatypePropertyMapping`, depending on whether the predicate attached to them is an `owl:ObjectProperty` or an `owl:DatatypeProperty`, respectively. As it can be observed, `map:Mapping` is defined as the disjoint union of both subclasses, as an instance of `map:Mapping` can belong to any of the subclasses but only can belong to one of them.

As it can be observed in the figure, another difference between the subclasses of `map:Mapping` is the target element expected for the values transformed, for the case of the `map:ObjectPropertyMapping` the expected target should be an instance of `owl:Class` while for the case of `map:DatatypePropertyMapping` it should map values to instances of the class `rdfs:Datatype`. The mappings are linked to these target elements by means of the properties `map:targetClass` and `map:targetDatatype`, respectively.

The class `map:AccessMapping` is included in the model in order to link one or more mappings that are executed with a given endpoint. This allows the definition of the mappings independently of the endpoint in which they can be executed since the link to the endpoint is established from the access mapping. A thing description, represented by the class `core:ThingDescription`, may have zero or more access mappings attached by means of the object property `map:hasAccessMapping`. The object property `map:isExecutedAfter` indicates dependency on the order of execution between access mappings.

Finally, the object property `map:valuesTransformedBy`, which can only be applied to `map:ObjectPropertyMapping` instances, is used to estate that the obtained values from a resource when applying a mapping should be transformed according to the referenced `core:ThingDescription`. This predicate is oriented to support the WoT discovery feature proposed at the W3C IG, taken literally *"The relationship between things provides a further basis for discovery. The relationships are defined through the models for things, where a thing has properties whose values are other things."*[13]

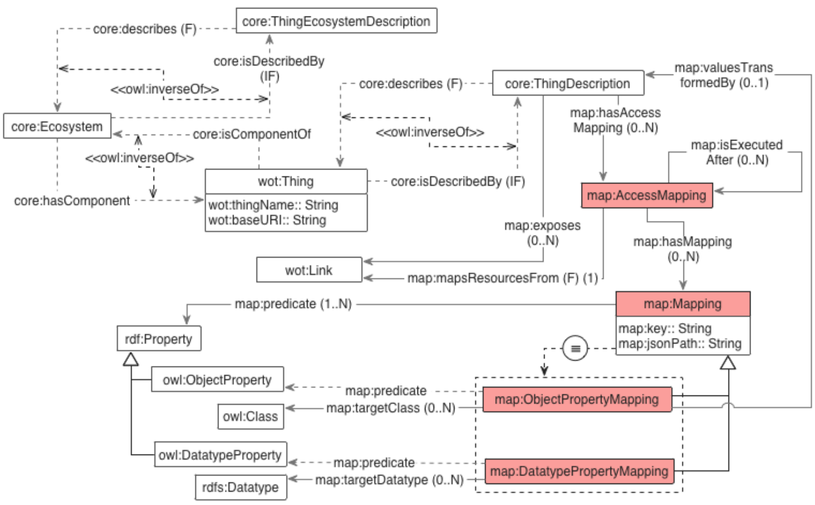

Fig. 2. General overview of the WoT mapping ontology

[13] See for further information https://www.w3.org/WoT/IG/wiki/Discovery_TF

5 Related work

The IoT ecosystem has been very prolific in producing different mechanisms for resource and service discovery. Datta et al. [2] analyse the most representative technologies and architectures used for discovery and propose a set of categories, namely distributed and P2P discovery, centralized architecture, semantic based, among others. Regarding the semantic based category, it is DiscoWoT [6] the only one that is based on the Web of Things. Even if the proposed solution is very flexible to different "Discovery Strategies", it neither considers extending descriptions by means of accessing the discovered devices, nor supports describing endpoint security constraints.

The W3C WoT IG made a great effort on evaluating[14] the technology landscape relevant to the standardization initiative. It is clear from such report that there has been a great deal of development and evolution with regards to discovery in the IoT. However, semantic web technologies seem to have a minor representation in the landscape: just a SPARQL endpoint that centralizes all thing descriptions. In addition, we can claim that while the IoT domain has gathered a lot of attention and numerous ontologies[15] have been defined to cover it in many ways [7], the WoT field has not been object of much attention.

One of the ontologies for modelling WoT is SWOT-O which was developed in the context of the SWoT4CPS framework [8]. This ontology represents the main WoT elements as entities, properties, actions and events. However, the actions and events represented in this ontology seem to have a narrower scope than the actions and events defined in the W3C WoT working group. For the documentation given, the actions and events in SWoT-O are related to actuators while our intention is to attach them to a more general concept including all flavours of web things. In addition, this model does not include where and how to access the values provided by the web things interaction patterns.

Alam and Noll [9] have developed an ontology for representing Web of Things concepts. The main issue with this work is that no pointer to the OWL ontology implementation is provided which restrains its reuse. Additionally, as for the documentation provided in the paper, this ontology does not consider interaction patterns modelling and actions.

Finally, the WoT ontology described by Charpenay et al. [10] proposes and extension of the Identifier-Resource-Entity pattern to include WoT resources. This ontology does not provide mechanisms to indicate where and how the resource values can be accessed and interpreted. Event though it is a good practice to extend upper level ontologies, the fact of importing DUL by means of the `owl:imports` predicate, makes this ontology too heavy for the given use case. More precisely, the WoT WG intention is to provide a neat lightweight core vocabulary, which of course could serve as the basis for further extensions.

[14] https://www.w3.org/WoT/IG/wiki/Tech_Landscape_Evaluation
[15] http://sensormeasurement.appspot.com/?p=ontologies

6 Conclusions

Along this paper the problem of discovery in the Web of Things has been characterised, highlighting that even if a common data model for describing things is a requirement for bringing interoperability to the IoT ecosystem, it is also very convenient to make use of it when implementing any discovery solution. Besides, an ontology-based approach for discovery has been proposed, which builds on the benefits of explicitly describing the semantics of the WoT ecosystem. In addition, the core ontologies involved in such solution have been described.

Regarding the ontology development, we plan to reuse existing ontologies that could fit in the models presented in this paper, for example reusing terms from or alignments with the ontology described by Charpenay and colleagues [10] as it is also being developed in the context of the W3C Web of Things WG. In addition, we also plan to provide examples of how to use the presented models including also connections with existing ontologies or datasets.

Finally, it is expected to evolve the proposed approach from the experience gained with the VICINITY project. Further, some experiments in other use cases will be performed with the intention of refining and generalizing the approach.

References

1. Zaslavsky, A., Jayaraman, P.P.: Discovery in the internet of things: The internet of things (ubiquity symposium). Ubiquity **2015**(October) (October 2015) 2:1–2:10
2. Datta, S.K., Ferreira da Costa, R.P., Bonnet, C.: Resource discovery in Internet of Things: Current trends and future standardization aspects. In: WF-IOT 2015, IEEE 2nd World Forum on Internet of Things, Milan, Italy. (12 2015)
3. Cirani, S., Davoli, L., Ferrari, G., Lone, R., Medagliani, P., Picone, M., Veltri, L.: A scalable and self-configuring architecture for service discovery in the internet of things. IEEE Internet of Things Journal **1**(5) (Oct 2014) 508–521
4. Studer, R., Benjamins, V.R., Fensel, D.: Knowledge engineering: Principles and methods. Data & Knowledge Engineering **25**(1-2) (1998) 161–197
5. Grüninger, M., Fox, M.S.: Methodology for the design and evaluation of ontologies. In: IJCAI'95, Workshop on Basic Ontological Issues in Knowledge Sharing. (1995)
6. Mayer, S., Guinard, D.: An extensible discovery service for smart things. In: Proceedings of the Second International Workshop on Web of Things. WoT '11, New York, NY, USA, ACM (2011) 7:1–7:6
7. Gyrard, A., Bonnet, C., Boudaoud, K., Serrano, M.: Lov4iot: A second life for ontology-based domain knowledge to build semantic web of things applications. In: Future Internet of Things and Cloud (FiCloud), IEEE (2016) 254–261
8. Wu, Z., Xu, Y., Yang, Y., Zhang, C., Zhu, X., Ji, Y.: Towards a semantic web of things: A hybrid semantic annotation, extraction, and reasoning framework for cyber-physical system. Sensors **17**(2) (2017) 403
9. Alam, S., Noll, J.: A semantic enhanced service proxy framework for internet of things. In: Proceedings of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, IEEE Computer Society (2010) 488–495
10. Charpenay, V., Käbisch, S., Kosch, H.: Introducing thing descriptions and interactions: An ontology for the web of things. In: Proceedings of the 1st Workshop on SemanticWeb technologies for the Internet of Things (SWIT) at ISWC. (2016)