Advances in Web of Things for IoT Interoperability

Standardized Thing Description for Interoperability

Soumya Kanti Datta and Christian Bonnet
EURECOM, Sophia Antipolis, France
Emails: dattas@eurecom.fr, bonnet@eurecom.fr

***Abstract*—It is widely acknowledged that the current Internet of Things (IoT) ecosystem is highly fragmented. The Web of Things (WoT) aims to counter the fragmentation using the principles of Web, standardized metadata and APIs. This paper describes the recent advances in W3C WoT including its architecture and building blocks. We also highlight an experiment using WoT Thing Description that help achieve interoperability between two IoT systems.**

Keywords*—*Internet of Things; Interoperability; Thing Description; Web of Things.

I. Introduction

The IoT is revolutionizing the current industries and creating new ones. The enterprises and consumers get numerous benefits in terms of remote monitoring and control of buildings, assets, health conditions as well as emergency response. But the current IoT ecosystem trends have given rise to silo-based development and deployments [1]. They fragment the consumer market where devices, applications and services are not interoperable. Our motivation is to use the Web of Things (WoT) to improve interoperability in the IoT systems. The WoT seeks to counter the fragmentation through wider adoption of the Web principles, standardized Thing Description, metadata and semantic web technologies[1]. They can be easily integrated into the IoT platforms to enable horizontal business cases and applications. Apart from that, utilization of the Web reduces development time and deployment of RESTful web services. Our goal in this paper is twofold - (i) to describe the recent advancements and accomplishments in WoT that aids in IoT interoperability and (ii) to report our interoperability experimentation with a WoT system for connected cars.

II. Web of Things Architecture and Building Blocks

WoT is an evolution of the IoT. The World Wide Web Consortium (W3C) proposes to extend the Web from a Web of pages to a Web of Things. The WoT architecture has been abstracted [2] (shown in Fig. 1) and is composed of three main components - (i) the connected device (Thing) level, (ii) the gateway or Edge level and (iii) the Cloud level.

The starting point in the WoT and its architecture is with the Thing itself, it can be real or virtual, connected or not. The Things are exposed to higher layer applications and services as software objects with APIs corresponding to events, properties and actions. Together they are called Thing Description (TD). The TD can represent a local Thing (sensor or actuator) or a remote Thing. The WoT is composed of three main building blocks - (i) Thing Description, (ii) Binding Templates and (iii) Scripting API while security mechanisms[2] are applied to all three of them (shown in Fig. 2).

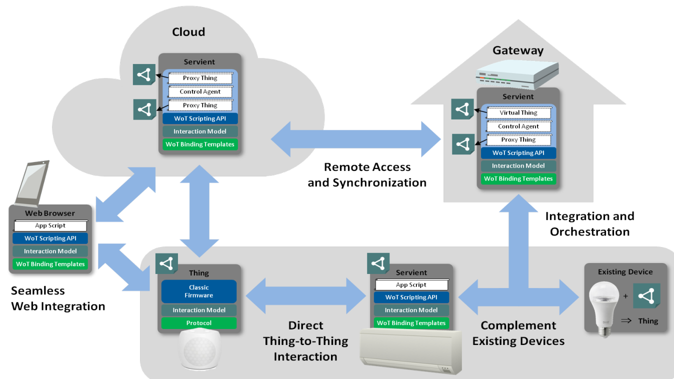

Fig. 1. WoT abstract architecture. [2]

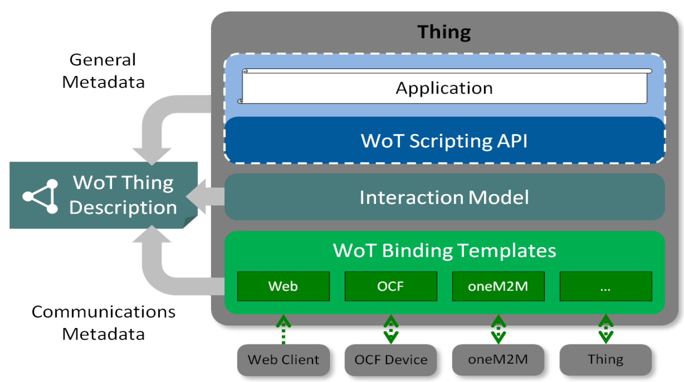

Fig. 2. Functional building blocks of WoT. [2]

A. WoT Thing Description

The TD is built upon a formal interaction model and support different messaging paradigms including request/response and publish/subscribe. The TD interaction pattern is composed of events, properties and actions [3]. Events are basically generic interactions where remote endpoints, other Things or IoT applications exchange data. Properties are data points of the Things that can be read (for sensors) and/or written to (for actuators). Actions are invokable processes mainly related to control purposes in actuators. Such TDs are mainly stored in a Thing Directory which is similar to CoRE Resource Directory. The Thing Directory provides several web services for TD registration, update, (automatic) removal and discovery [4]. SPARQL endpoints may be exposed to allow semantic

[1]https://www.w3.org/2016/08/wot-white-paper

[2]Discussion on WoT Security and threat models are out of scope for this paper.

queries and semantic based discovery mechanisms. The TD are serialized using JSON-LD which provides good mechanism for machine-understandable semantics. Below is a part of the TD (Fig. 3) we created for a connected car that uses WoT mechanisms to send data to a Cloud.

```
{
        "@context": [
                "http://w3c.github.io/wot/w3c-wot-td-context.jsonld",
                {
                        "actuator": "http://example.org/actuator#",
                        "sensor": "http://example.org/sensors#"
                }
        ],
        "@type": ["Thing"],
        "name": "BMW_X5",
        "base": "http://192.168.1.122:3000/",
        "interaction": [
                {
                        "@type": ["Property","sensor:gearPosition"],
                        "name": "gear",
                        "outputData": { "type": "string" },
                        "writable": false,
                        "link": [{
                                "href" : "north/sensor/gearPosition",
                                "mediaType": "application/json"
                        }]
                },
                {
                        "@type": ["Action","actuator:turnOn"],
                        "name": "turnOn",
                        "link": [{
                                "href" : "north/rear-left-window/turnon",
                                "mediaType": "application/json"
                        }]
                },
```

Fig. 3. Example of WoT Thing Description. [2]

B. WoT Binding Templates

The WoT TD enables interoperability in describing Things' capabilities and how to access them. Still the Things belong to different IoT platforms which follow (i) standards like oneM2M, OCF or (ii) proprietary technology. Moreover, these protocols may use a variety of communication technologies and protocols for Thing data exchange. The WoT Binding Template makes it easier for WoT applications to interact with IoT platforms through an interoperable communication metadata blueprints. Such template is created once for each IoT platform and is instantiated & recorded in all relevant TDs. A WoT Client or application that discovers and consumes the TDs must implement the corresponding protocol binding (e.g. between LwM2M and HTTP). The communication metadata (which is a part of the TD) in this case includes IoT platform, transfer protocol, media type and security parameters. It is shown in Fig. 4.

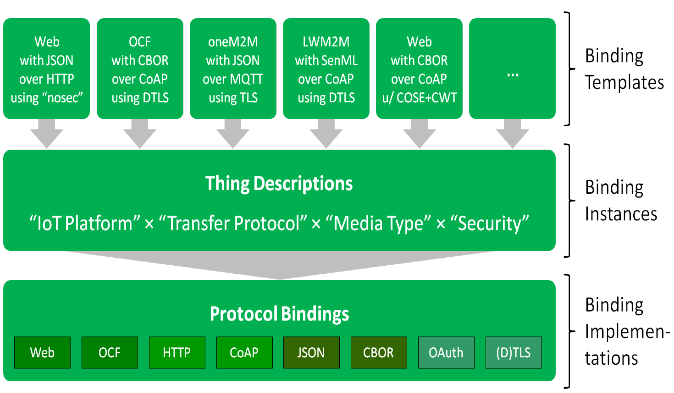

Fig. 4. WoT Binding Templates and Protocol Bindings. [2]

C. Scripting API

For ease of the IoT application development, WoT provides Scripting API. It provides a runtime system for IoT applications (similar to a Web Browser) to improve productivity and reduce software integration cost. Such a standardized API allows portability of application logic from constrained Things to more powerful gateways or Cloud systems.

III. Interoperability Experiment

We created a Thing Description for a connected car following the W3C WoT ongoing standardization effort. Parts of the TD is shown in Fig. 3 and it was a part of a device servient representing the connected car. Another team developed an application servient that discovers and retrieves the car TD from a Thing Directory. The TD JSON-LD[3] file listed the capabilities of six sensors and two actuators. The goal of the interoperability experiment was twofold - (i) the application servient will parse and consume the TD and (ii) the application servient will be able to extract the web services for Things and exchange sensing & actuation data with them.

We report that the application servient used a JSON-LD parser and a standardized mechanism to parse the car TD. Both the objectives of the interoperability experiment were met and the event was successful. It should be noted that the application and device servients belong to two different IoT systems using RESTful HTTP based web services. Therefore, the first step of interoperability through Thing Description was achieved between two IoT systems. This is one major step towards harmonizing the IoT Ecosystems using the WoT building blocks.

IV. Conclusion

This paper describes the recent advances made in the W3C WoT in terms of its abstract architecture and building blocks. Our main contribution lies in reporting the successful interoperability experiment using TD (of a connected car) between two IoT systems. As of future work, we are now considering an advanced interoperability scenario including both TD and WoT Binding Templates.

V. Acknowledgement

This work has been funded through the French Project SelfPower IPCOM Pole SCS. EURECOM acknowledges the support of its industrial members - BMW Group, IABG, Monaco Telecom, Orange, SAP, and Symantec. The authors also acknowledge the work done by W3C WoT Interest Group and Working Group members.

References

[1] S. K. Datta, C. Bonnet, R. P. F. D. Costa, and J. Hrri, "Datatweet: An architecture enabling data-centric iot services," in *2016 IEEE Region 10 Symposium (TENSYMP)*, pp. 343–348, May 2016.

[2] "Web of things (wot) architecture (editorial draft) - https://w3c.github.io/wot-architecture/," 2018.

[3] S. K. Datta and C. Bonnet, "Describing things in the internet of things: From core link format to semantic based descriptions," in *2016 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pp. 1–2, May 2016.

[4] A. Bröring, S. K. Datta, and C. Bonnet, "A categorization of discovery technologies for the internet of things," in *Proceedings of the 6th International Conference on the Internet of Things*, IoT'16, (New York, NY, USA), pp. 131–139, ACM, 2016.

[3] https://json-ld.org/