End-to-End Semantic Interoperability Mechanisms for IoT

Eftychia Lakka, Nikolaos E. Petroulakis, George Hatzivasilis, Othonas Soultatos, Manolis Michalodimitrakis
Foundation for Research and Technology – Hellas (FORTH)
Heraklion, Crete, Greece
elakka@ics.forth.gr, npetro@ics.forth.gr, hatzivas@ics.forth.gr, soultatos@ics.forth.gr, manmix@ics.forth.gr

Urszula Rak, Karolina Waledzik
BlueSoft Sp. z.o.o.
Warsaw, Poland
urszula.rak@bluesoft.com, karolina.waledzik@bluesoft.com

Darko Anicic, Vivek Kulkarni
Siemens AG Corporate Technology Siemens
Munich, Germany
darko.anicic@siemens.com, vivekkulkarni@siemens.com

***Abstract*—Semantic interoperability is the designed property where various systems can interact with each other and exchange data with unambiguous, shared meaning. This enables knowledge discovery, machine computable reasoning and federation of different information systems. Traditionally, technologies from the Semantic Web are adapted in order to capture the inherited properties of the Internet of Things (IoT) domain. Such technologies provide common description and representation of data and services; they characterize things and their capabilities, deal with the semantic annotation, resource discovery, access management, knowledge extraction in a machine-readable or interoperable manner. Thus, the common interpretation of semantic information in a globally shared ontology could be quite useful. However, several local systems may utilize popular or standardized ontologies, eventually they extend them and establish their own semantics and interfaces. As a result, the direct interaction between these systems is not feasible. With this in mind, our goal is to propose the use of semantic interoperability mechanisms, which correlates the required information and enables the interoperability of systems with different semantics or cross-domain interaction. A motivating example of smart sensing is analyzed along with the implementation of the proposed approach.**

***Index Terms*—IoT, semantic, interoperability, thing description, ontology alignment, semantic validator**

I. Introduction

Traditionally, the Internet of Things (IoT) is a network, consisting of physical or virtual object/devices (Things), equipped with sensing, accumulating and transferring data over the internet automatically. With billions of sensors and actuators already deployed, and combined into a number of domain-specific platforms, are among the biggest conceptual and technological challenges of our time [1]. As a result, a widespread interest of both industry and academy was increased for the researching and developing IoT technology [2]; many approaches at different levels (from the IoT device-level to cloud/backend IoT platforms) have been proposed. Due to that, a huge amount of heterogeneous data forms from multiple sources has been increasing tremendously; the management and the maintenance of these becomes priority and a complicated issue in order to facilitate society, economy and daily living [3]. For this reason, several researchers have started to study the application of semantic interoperability principles applicable to the IoT ecosystems. Particularly, interoperability is the ability of a system to work with or use the components of another system. It is relatively easy to achieve integration of different systems within the same domain or between different implementations within the stack of a specific software vendor [4], [5], [6], [7], [8]. For example, a smart watch developed in Android cannot interact with a smart bulb without the relevant proprietary gated application provided by the same vendor [9]. Thus, islands of IoT functionality are established leading towards a vertically-oriented 'Intranet-of-Things' rather than the 'Internet-of-Things'.

In general, interoperability among heterogeneous systems can be understood and involves the technical interoperability, the syntactical interoperability, the semantic interoperability, and the organizational interoperability. This paper focus on how to solve the end-to-end semantic interoperability in an IoT landscape. For full semantic interoperability, it is necessary that the devices interpret the shared information correctly and act accordingly, which is covered by the semantic aspect of interoperability usually referred to as Information Model. Hence, improvements have to be made regarding distributed ontologies, semantic web, or semantic device discovery [10].

In other words, to take advantage of the full potential of the IoT vision, we need standards to enable the horizontal and vertical communication, operation, and programming across devices and platforms, regardless of their model or manufacturer. As it concerns the meaning of data, which is the main focus of this paper, semantics can settle commonly agreed information models and ontologies for the used terms that are processed by the interfaces or are included in the exchanged data. However, as there are several ontologies for describing each distinct Thing, we need semantic interoperability mechanisms in order to perform common data mapping across the various utilized formats (e.g. XML or JSON) and ontology alignment.

In order to address those challenges, we propose a frame-

978-1-7281-1016-5/19/$31.00 ©2019 IEEE

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:44:57 UTC from IEEE Xplore. Restrictions apply.

work to enable end-to-end compatibility and cooperation at all layers. Thus, semantic interoperability mechanisms across all levels must be deployed in order to resolve semantics between the field and backend layers. The below-proposed approach is based on the development and the architecture that is carried out in SEMIoTICS[1], a horizon 2020 project. The Fig. 3 illustrates the main components of SEMIoTICS architecture, which consists of three layers: Field Layer, SDN/NFV Orchestration Layer and Application Orchestration Layer [11].

The rest paper is organized as follows: the second section begins with the background and related work regarding the various semantic theory/technologies. The section 4 details the implementation of the proposed approach. Our conclusions are drawn in the final section.

II. Background

This section details the background and related work regarding the various semantic technologies. This includes basic notation, the description of Things and the functionality of each mechanism, how they are all linked together, and for what purpose. The semantic interoperability of our proposal is inspired by the modern Linked-Data (LD) approach. Finally, semantic models for IoT and smart objects are presented in the last subsection.

A. Related work

1) Theory of Semantics: In the literature, Berners-Lee et al. [12] proposed the most appropriate informal definition that says "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation". Particularly, the purpose of the Semantic Web is to provide the Web content in a form which can be understandable and processable by any machine. Hence, Semantic Web technologies that include both intelligent methods (i.e. reasoning) and automated tasks are proposed to increase the proliferation of connected devices that are part of the IoT, using Semantic Web standards [13]. A Uniform Resource Identifier (URI) provides a simple and extensible means for identifying a resource (RFC 3986[2]). Fig. 1 disassembles the URI syntax.

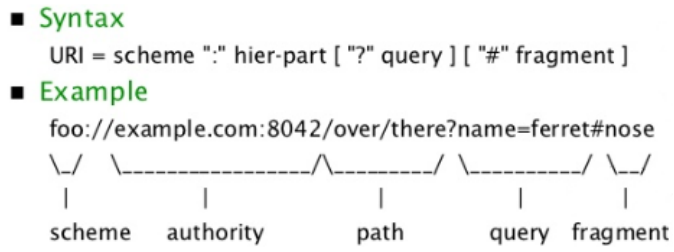


Fig. 1: URI syntax

Several schemes are used nowadays for URIs, like http://, ftp://, tel:, urn:, and mailto: (the URI scheme is not the same as the underlying protocol). The HTTP URIs are the most common data access and things identification mechanism. They provide globally unique names, distributed ownership, and allow people to look up those names. The Resource Description Framework (RDF) is a data format for representing things and their interrelations. The RDF data model is formed as triples of (subject - predicate - object).

In contrast to the Semantic Web technologies that focus on the ontological level or knowledge inference, LD is mainly designed for publishing structured data in RDF using URIs. The provided simplification lowers the entry barrier for data provider and enables the wide-spread adoption. The LD approach proposes 4 principles:

- Use URIs to name things on the Web
- Use HTTP URIs allowing to look-up those names on the Web
- When someone looks-up a URI provide useful information
- Include links to other URIs to allow discovery of more things

The abovementioned links are usually RDF properties that are interpreted as hyperlinks. The LD setting accomplishes ease of discovery and information consumption, reduced redundancy, and added value. JSON-LD is a popular implementation of the LD concept. It is developed by leveraging the Schema.org vocabulary. It is a joint effort by Google, Bing, Yahoo, and Yandex to establish a unified structured data vocabulary for the Web. JSON-LD annotates elements on a web page and structures the data. These features are utilized by search engines in order to disambiguate elements and derive facts surrounding entities. Once associated, they can create a more organized and better Web overall.

2) Semantic Models for IoT and Smart Objects: In general, an ontology is defined as "a formal, explicit specification of a shared conceptualization" and is used to represent knowledge within a domain as a set of concepts related to each other. In the area of IoT domain, an ontology provides all the crucial semantics for the IoT devices as well as the specifications of the IoT solution (input, output, control logic) that is deployed in such devices. The abovementioned semantics shall include the terminology related to sensors and observations and extend them to capture also the semantics of devices beyond sensors (e.g. actuators, tags, embedded devices, features of interest).

There are some ontologies that are among the most widely used in the IoT domain. In fact, the Semantic Sensor Network (SSN) and Sensor Observation Sampling Actuator (SOSA) ontologies by the W3C community are attracting considerable interest [14]. The SOSA/SSN ontologies model sensors, actuator, samplers as well as their observation, actuation, and sampling activities. The ontologies capture the sensor and actuator capabilities, usage environment, performance, and enable contextual data discovery. This also constitutes the standardized ontologies for semantic sensor networks. The cooperation of SSN and SOSA offers different scopes and degrees of axiomatization that enable a wide range of application scenarios towards the Web of Things [15]. In detail,

[1]https://www.semiotics-project.eu/
[2]https://www.ietf.org/rfc/rfc3986.txt

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:44:57 UTC from IEEE Xplore. Restrictions apply.

the SSN ontology is a suite of general-purpose ontologies. It embodies the following 10 conceptual modules: 1) Device, 2) Process, 3) Data, 4) System, 5) Deployment, 6) PlatformSite, 7) SSOPlatform, 8) OperatingRestriction, 9) ContraintBlock, and 10) MeasuringCapability. The modules consist of 41 concepts and 39 object properties.

Additionally, others other semantic models for smart objects are the SWAMO[3] which is created to enable dynamic, composable interoperability of sensors, web products and services. It focuses on the sensor domain and particularly on processes to control them. It is interoperable with the Sensor Web Enablement (SWE) descriptions. CSIRO[4] designed to describe and reason about sensors, observations and scientific models. It provides a semantic description of sensors for use in workflows. It was used to develop the Semantic Sensor Network (SSN) ontology. Also, the Open Mobile Alliance (OMA) Lightweight for Machine to Machine (LWM2M) or IoT device management is based on a client component, which resides in the LWM2M Device, and a server component, which resides within the M2M Service Provider or the Network Service Provider. A client may have any number of resources and these resources are organized into objects. The OMA LWM2M enabler focuses on device management and service enablement for LWM2M Devices. Each resource supports one or more operations. The OMA Lightweight M2M[5] ontology describes the resources, objects and operations supported by the OMA LWM2M architecture.

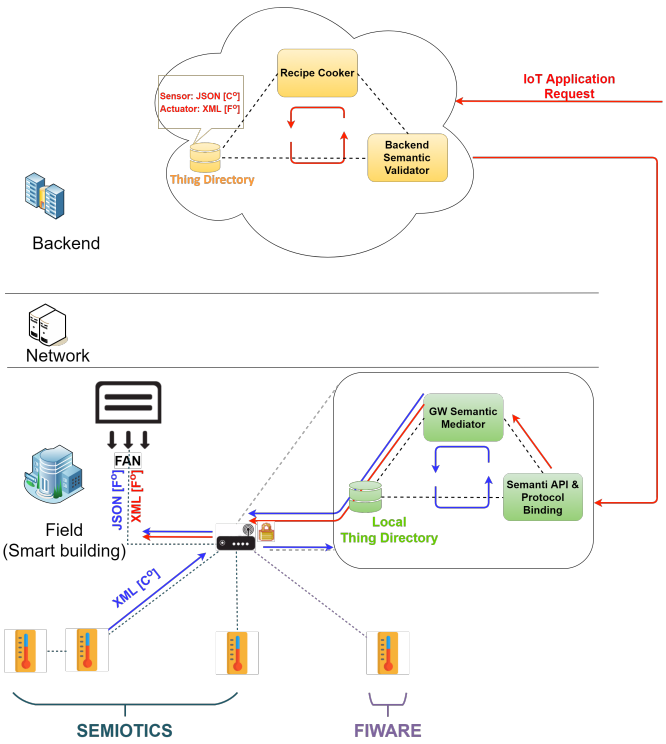


Fig. 2: The smart sensing interoperability scenario

III. Methodology

A. Motivated Scenario

As a motivating example, we consider the smart sensing scenario. A smart building deploys several sensing equipment in order to support pervasive and ubiquitous functionality. Horizontal operation in the field layer is mandatory as well as vertical cooperation with the backend.

The main functionality of the system is the optimization of energy consumption and can be deployed either in the home gateway and/or in the backend layer. The interoperability of the underlying IoT devices and the system services must be guaranteed regardless their brand or manufacturer.

As an indicative scenario, we consider the case where the user installs temperature sensors in the rooms. Three types of sensory devices are modelled (Fig. 2). The first one is bought from a European vendor, it measures the temperature in the Celsius scale (°C) and transmits data in an XML format; the second one is bought from USA, it measures the temperature in the Fahrenheit scale (°F) and transmits JSON messages; the third sensor is compatible with the semantics of the FIWARE[6] project, it measures the temperature in °C and transmits JSON messages.

In fact, we model the process of semantic interoperability where the system uses the collected data in order to take real-time decisions. The system functionality must retain a specified temperature value in the building. The components of the SEMIoTICS architecture (See Fig. 3) that are involved in this process are both at backend and field layer. Namely, the Backend Semantic Validator (BSV) which is responsible for semantic validation mechanisms; the Thing Directory component that are the repository of knowledge containing the necessary Thing models; the Recipe Cooker component, which is responsible for cooking (creating) recipes reflecting user requirements on different layers (cloud, edge, network) as well as transforming recipes into understandable rules for each layer. It uses the Thing Directory with all the models required to create these rules. At the field layer, the GW Semantic Mediator (GWSM) component for the semantic mapping between different data models; the Semantic API Protocol Binding (SAPB) component for binding different protocol and exposing a common semantic API located at the Generic IoT Gateway layer; Local Thing Directory, which stores locally the semantic description of Things in the Generic IoT Gateway.

With this in mind, we consider two main data flow types that implement both the horizontal and vertical communication, operation, and programming across devices and platforms, regardless their model or manufacturer. They are depicted in the Fig. 2, with blue and red colours respectively, and are detailed in the text below.

[3]https://www.w3.org/2005/Incubator/ssn/XGR-ssn-20110628/SWAMO
[4]https://www.w3.org/2005/Incubator/ssn/XGR-ssn-20110628/
[5]https://sites.google.com/site/smartappliancesproject/ontologies/oma-lightweight$_m 2m - ontology$
[6]https://www.fiware.org/

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:44:57 UTC from IEEE Xplore. Restrictions apply.

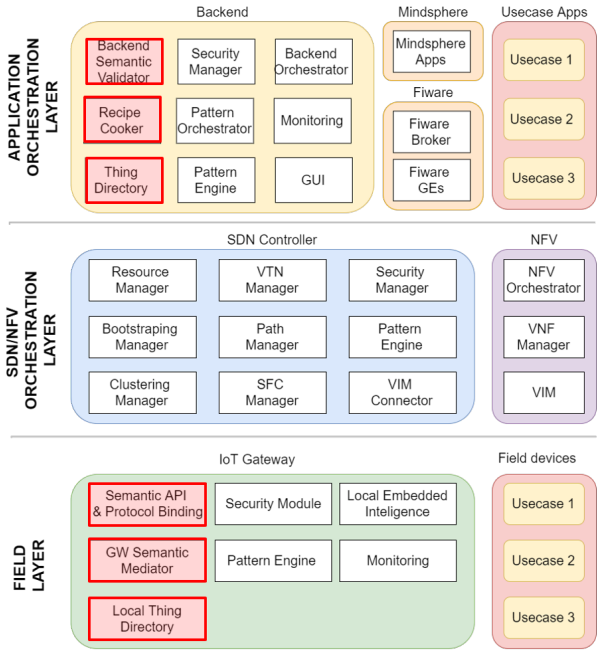


Fig. 3: SEMIoTICS Architecture - Components

In the horizontal scenario, the semantic interoperability mechanisms run in the local gateway with the aim to retain a specified temperature value in the building. If the temperature in a room goes beyond the specified threshold, the relevant fan equipment is adjusted accordingly. The blue data flow (See Fig. 2) depicts the sensor device which sends data (temperature) in (°C) and uses the XML format to transmit data. However, the actuator (fan) requires data in (°F) and a JSON format file for transmission. Hence, the semantic interoperability mechanisms are responsible for resolving this semantic difference. Particularly, the procedure starts with searching for the necessary Thing models in the Local Thing Directory Component, in order to detect the above potential semantic conflicts between the interacting Things (sensor, actuator). Afterwards, the Semantic Edge Platform in the SAPB component is responsible to solve the semantic conflicts of temperature units, using the Adaptor Nodes that configure an Interaction Pattern in accordance to the application's requirements. Finally, the GWSM component is triggered to send the request in an appropriate format to the target Thing (actuator). Following the above scenario, in the horizontal, the main purpose is the same as the previous description, but it requires vertical operation and communication across devices from different layers. Specifically, the BSV component can receive a request from an IoT application for the interaction between a sensor and an actuator, which are described with two different TDs (based on W3C Thing Descriptions that are serialized to the JSON-LD standard format), respectively. The functionality of this component consists of: 1) The search of the necessary Thing models in the Thing Directory component, in order to detect any potential semantic conflicts between the interacting domains. In this case (See Fig. 2) , the request refers to a sensor device, which sends the data (temperature) in (°C) and uses JSON format to transmit data. However, the actuator (fan) requires data in (°F) and XML format file. 2) The connection with the Recipe Cooker component to resolve the semantic conflicts of temperature units, using the Adaptor Nodes that configure an Interaction Pattern in accordance to the application's requirements. 3) The transfer of the translated request to the SAPB component which is responsible to trigger the GWSM component in the field layer, in order to send the request in an appropriate format to the target Thing (actuator).

B. Implementation

This chapter outlines the methodology that is implemented for the proposed approach, data transformation techniques and validation mechanisms, to ensure end-to-end semantic interoperability. In Fig. 4, a sequence diagram highlights the procedure of the semantic interoperability mechanisms between the backend and the field layer. As stated in previous subsection, the aim is the connection between two Things (Sensor and Actuator), by an IoT application which sends request in the backend.

From a technical point of view, the implementation of the semantic interoperability procedure contains the following parts:

- **IoT application request**: The BSV component receives the request from the IoT application, using grpc[7] framework (in JSON-LD/JSON format). In fact, gRPC provides protocol buffer compiler plugins that generate client- and server-side code. On the server side, that means in BSV component, a method is implemented declared by the service and runs a gRPC server to wait and handle the client calls. On the client side, IoT application has this role, implements the same methods as the above service. The client call this method and wrap the parameters for the call in the appropriate protocol buffer message type.
- **JSON-LD/JSON Parser**: The JSON-LD/JSON Parser (in Java) is implemented as part of the BSV component, using the org.json.* library, in order to analyze the received input and extract the meaningful information from these set of data (See Code1).
- **Setup of Thing Directory**: The Thingweb Directory[8] is used for the implementation both of Thing Directory component (in backend layer) and Local Thing Directory component (in field layer). The interaction is achieved by the HTTP endpoint that provides an HTML client to register and discover TDs. This client accesses a REST API to manage TDs that complies to the IETF Resource Directory specification. Registration is done by POSTing and discovery can be performed by using a SPARQL graph pattern as a query parameter (GET).
- **Interaction with Thing Directory**: The BSV component should interact with the Thing Directory component; this

[7]https://grpc.io/
[8]https://github.com/thingweb/thingweb-directory

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:44:57 UTC from IEEE Xplore. Restrictions apply.

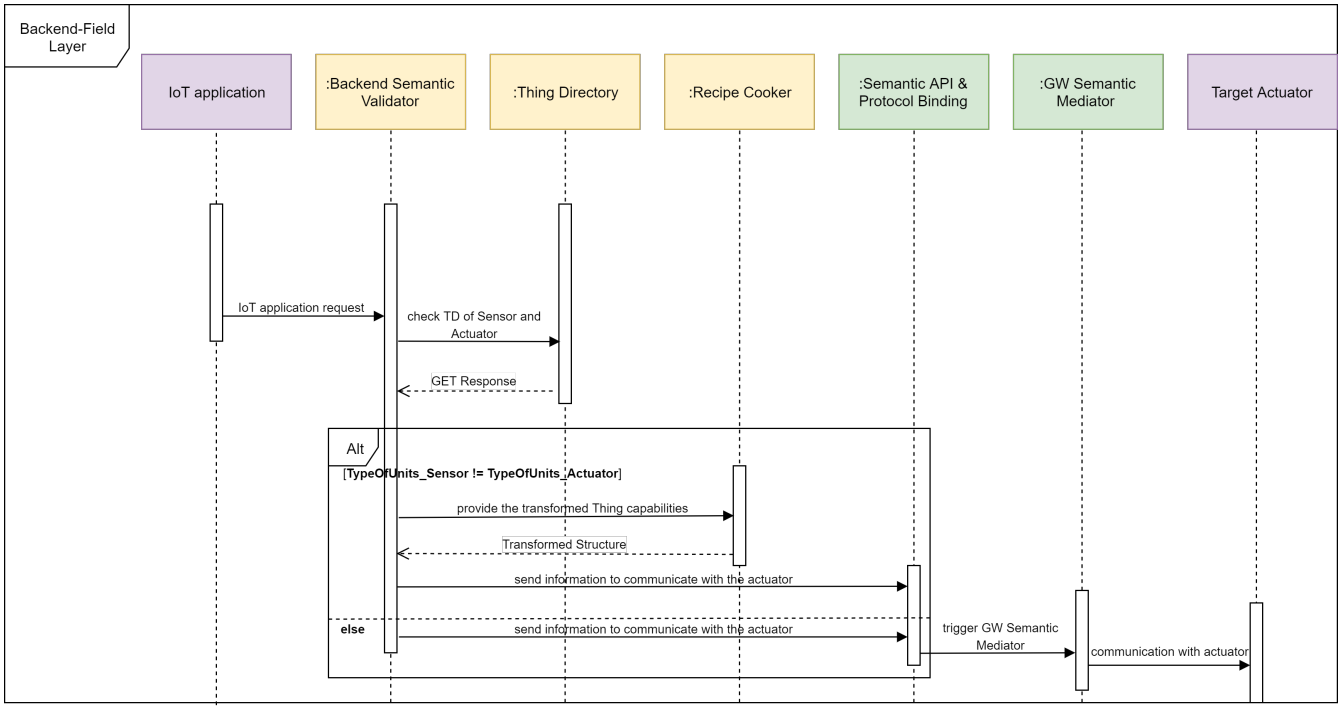


Fig. 4: Sequence diagram for semantic interoperability mechanisms

stage consists of two procedures, the TD discovery of the specific Thing and the TD registration for the case that this Thing is not included in the Thing Directory. In first case,the sendGET function is developed (see Code2) that uses HttpURLConnection for send an HTTP GET request to Thing Directory in order to get the search result. For this discovering, SPARQL query can be used to retrieve TDs based on their IDs and should be percent-encoded (urlEncoder function).

Hence, depending the above result, if the TD of the Thing is not in the Thing Directory, a POST request in Thing Directory is implemented for the registration of new TD. The Code3 depicts the send of a POST request with a JSON body using the HttpClient, setting both the "Request Content-Type Header Parameter" and "Accept Request Header", as application/td+json.

- **Interaction with Recipe Cooker component**: The Recipe Cooker (based on the Node-RED framework[9]) is responsible to harmonize the semantic model capabilities with the registration of extra Adaptor nodes, which provide the functionality of temperature value transformation from the Celsius to the Fahrenheit scale and vice versa in the recipe, based on the Temperature Control Example[10].
- **Connection with the Field Layer**:For the final stage, as the sequence diagram depicts, the SAPB component and the GWSM component from the backend layer are responsible to transmit the information, in an appropriate format, to the target actuator.

IV. Conclusion

This work presented the development of data transformation techniques and validation mechanisms to ensure end-to-end semantic interoperability. The proposed approach includes mappings between datatypes to ensure that data flow is possible between smart objects (Things). In addition, semantic transformation methods are defined with the purpose of resolving, if possible, conflicts among the semantic annotations.The representative motivated scenario is briefly described involving smart temperature sensing; the functional components of the SEMIoTICS architecture, which are involved and are responsible for logical and structural data transformation, are illustrated in detail. As for the future work, the framework will be enhanced with further semantic data by new devices, in order to be applicable to industrial and healthcare domain.

```
import org.json.*;
public static JSONObject ParseJSON(String
    input, String key)throws
    FileNotFoundException, IOException
{
  JSONObject obj = new JSONObject(input);
  String id  = obj.getString(key);
  return obj;
}
```

Code1: JSON-LD/JSON Parser

[9]https://nodered.org/

[10]https://github.com/iot-schema-collab/iotschema-node-red/blob/master/example-doc.md

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:44:57 UTC from IEEE Xplore. Restrictions apply.

```
//HTTP GET request in the Thing Directory
public static void sendGet(String url,
    JSONObject inputJSON) throws Exception
{
  final String USER_AGENT = "Mozilla/5.0";
  URL obj = new URL(url);
  HttpURLConnection con = (
      HttpURLConnection) obj.openConnection
      ();
  // optional default is GET
  con.setRequestMethod("GET");
  //add request header
  con.setRequestProperty("User-Agent",
      USER_AGENT);
  int responseCode = con.getResponseCode();
  BufferedReader in = new BufferedReader(
      new InputStreamReader(con.
      getInputStream()));
  String inputLine;
  StringBuffer response = new StringBuffer
      ();
  while ((inputLine = in.readLine()) !=
      null)
  {
    response.append(inputLine);
  }
  in.close();
}
//Encoder for Query in Thing Directory
public static String urlEncoder(String
    inputURL) throws
    UnsupportedEncodingException
{
    String url = IPAddress + "td-lookup/sem
        ?query=" + URLEncoder.encode(
        inputURL, "UTF-8");
    return url;
}
```

Code2: Discovery of a TD based on Thing ID in Thing Directory

```
//HTTP POST request in the Thing Directory
public static void sendPost(JSONObject
    varInput) throws Exception
{
  CloseableHttpClient client = HttpClients.
      createDefault();
  HttpPost httpPost = new HttpPost(url);
  StringEntity entity = new StringEntity(
      varInput.toString();
  httpPost.setEntity(entity);
  httpPost.setHeader("Accept", "application
      /td+json");
  httpPost.setHeader("Content-type", "
      application/td+json");
  CloseableHttpResponse response = client.
      execute(httpPost);
  client.close();
}
```

Code3: Registration of a new TD in Thing Directory

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreements No. 780315 (SEMIoTICS) and No. 830927 (CONCORDIA).

REFERENCES

[1] M. Ganzha, M. Paprzycki, W. Pawlowski, P. Szmeja, and K. Wasielewska, "Semantic Technologies for the IoT - An Inter-IoT Perspective," in *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, apr 2016, pp. 271–276.

[2] G. Fortino, C. Savaglio, C. E. Palau, J. S. de Puga, M. Ganzha, M. Paprzycki, M. Montesinos, A. Liotta, and M. Llop, "Towards Multi-layer Interoperability of Heterogeneous IoT Platforms: The INTER-IoT Approach." Springer, Cham, 2018, pp. 199–232.

[3] M. G. Kibria, S. Ali, M. A. Jarwar, and I. Chong, "A framework to support data interoperability in web objects based IoT environments," in *2017 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, oct 2017, pp. 29–31.

[4] J. Serra, D. Pubill, A. Antonopoulos, and C. Verikoukis, "Smart HVAC control in IoT: energy consumption minimization with user comfort constraints." *TheScientificWorldJournal*, p. 161874, jun.

[5] IEEE Communications Society, IEEE Photonics Society, Optical Society of America, and Institute of Electrical and Electronics Engineers, *2016 Optical Fiber Communications Conference and Exhibition (OFC) : 20-24 March 2016*.

[6] N. E. Petroulakis, K. Fysarakis, I. Askoxylakis, and G. Spanoudakis, "Reactive security for sdn/nfv-enabled industrial networks leveraging service function chaining," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 7, p. e3269, 2018.

[7] K. Fysarakis, G. Hatzivasilis, C. Manifavas, and I. Papaefstathiou, "RtVMF: A Secure Real-Time Vehicle Management Framework," *IEEE Pervasive Computing*, vol. 15, no. 1, pp. 22–30, jan 2016.

[8] G. Hatzivasilis, I. Papaefstathiou, and C. Manifavas, "SCOTRES: Secure Routing for IoT and CPS," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2129–2141, dec 2017.

[9] G. Hatzivasilis, I. Askoxylakis, G. Alexandris, D. Anicic, A. Broring, V. Kulkarni, K. Fysarakis, and G. Spanoudakis, "The Interoperability of Things: Interoperable solutions as an enabler for IoT and Web 3.0," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, sep 2018, pp. 1–7.

[10] V. Gazis, M. Goertz, M. Huber, A. Leonardi, K. Mathioudakis, A. Wiesmaier, and F. Zeiger, "Short Paper: IoT: Challenges, projects, architectures," in *2015 18th International Conference on Intelligence in Next Generation Networks*. IEEE, 2015, pp. 145–147.

[11] N. E. Petroulakis *et al.*, "Semiotics architectural framework: End-to-end security, connectivity and interoperability for industrial iot," in *2019 Global IoT Summit (GIoTS)*. IEEE, 2019, pp. 1–6.

[12] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, pp. 34–43, may 2001.

[13] I. Tachmazidis, J. Davies, S. Batsakis, G. Antoniou, A. Duke, and S. Stincic Clarke, "Hypercat RDF: Semantic Enrichment for IoT." Springer, Cham, 2016, pp. 273–286.

[14] A. Haller, K. Janowicz, S. J. Cox, M. Lefrançois, K. Taylor, D. Le Phuoc, J. Lieberman, R. García-Castro, R. Atkinson, and C. Stadler, "The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation," *Semantic Web*, vol. 10, no. 1, pp. 9–32, dec 2018.

[15] D. Zeng, S. Guo, and Z. Cheng, "The web of things: A survey," *JCM*, vol. 6, no. 6, pp. 424–438, 2011.

Authorized licensed use limited to: Foundation for Research and Technology (FORTH). Downloaded on September 07,2020 at 11:44:57 UTC from IEEE Xplore. Restrictions apply.