BIG IoT – Bridging the Interoperability Gap of the Internet of Things

Deliverable 3.2.a

Semantic Interoperability Design for Smart Object Platforms and Services

- first release

Date: 31.12.2016

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688038.

Responsible Person and Affiliation	Dr. Achille Zappa (NUIG)
Due Date / Delivery Date	31 December 2016
Author(s):	Sebastian Kaebisch (Siemens), Darko Anicic (Siemens), Aparna Saisree Thuluva (Siemens), Victor Charpenay (Siemens), Arne Bröring (Siemens), Yong Wang (TU Clausthal), Hoan Quoc (NUIG), Achille Zappa (NUIG).
Reviewer(s):	Martin Lorenz (ATOS) Luca Gioppo (CSI)
State	Final
Version	1.0
Confidentiality	Public

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbrevia-tion	Meaning
AC	Alternating Current (Here: Slow Charging Stations)
API	Application Programming Interface
APM	Advanced Parking Management
BCN	Barcelona (pilot)
BIG IoT	Bridging the Interoperability Gap of the Internet of Things
CoAP	Constrained Application Protocol
CSI	Project Partner Piedmont, Italy
CSP	Project Partner Piedmont, Italy
DC	Direct Current (Here: Fast Charging Stations)
DOA	Description of Action
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
ICE	Inter City Express (German High Speed Train)
IERC	IoT European Research Cluster
IoT	Internet of Things
IoT-A	Internet of Things - Architecture
ISO	International Organization for Standardization
MQTT	Message Queuing Telemetry Transport
NG	Northern Germany (pilot)
OD	Offering Description
OGC	Open Geospatial Consortium
PIE	Piedmont (pilot)
RDF	Resource Description Framework
REST	Representational State Transfer
SDK	Software Development Kit
SSN	Semantic Sensor Network
TD	W3C Things Description
TIC	Traffic Information Center
TUC	TU Clausthal Partner Northern Germany
UI	User Interface
UPC	Project Partner Barcelona, Spain
URI	Uniform Resource Identifier
VMZ	Project Partner Berlin, Northern Germany
W3C	World Wide Web Consortium
Wi-Fi	Wireless Local Area Network

WLAN	Wireless Local Area Network
WMS	Web Map Service
WP	Work Package

BIG IoT Terminologies and Definitions

Terms	Definitions
Accounting	Accounting collects data about each access to an Offering and relates it to the respective Subscription.
BIG IoT API	A set of specifications for Providers and Consumers to interact with the BIG IoT Marketplace to authenticate, register, discover and subscribe to Offerings; and perform accounting
Consumers to directly access the Resources offered by a Provider	The BIG IoT API defines the supported communication protocols, data formats, semantic descriptions, etc. In order to facilitate BIG IoT Applications, Services and Platforms to implement and use the BIG IoT API, dedicated Provider and Consumer Libs (SDKs) are provided for various platforms and programming languages, offering also programming interfaces to developers.
BIG IoT Application (or short Application)	An application software that uses the BIG IoT API to discover Offerings on the BIG IoT Marketplace, subscribe to Offerings and access the offered Resources. A BIG IoT Application acts merely as an Offering Consumer.
BIG IoT Application / Service / Platform Developer (or short BIG IoT Developer)	A software developer that implements or integrates a BIG IoT Service, Application or Platform.
BIG IoT Application / Service / Platform / Marketplace Provider or Operator	The organization that operates a BIG IoT Application, Service, Platform, Marketplace instance. It is hereby not relevant if a particular instance is hosted on the provider organization's own infrastructure or a 3rd party infrastructure.
BIG IoT Core Developer	A software developer that implements or extends BIG IoT Marketplace and/or BIG IoT Lib components.
BIG IoT enabled Platform (or short BIG IoT Platform or just Platform)	An IoT Platform (or Smart Object Platform) that implements and uses the BIG IoT API to register Offerings on the BIG IoT Marketplace and provide access to the offered Resources. A BIG IoT Platform acts merely as an Offering Provider.
BIG IoT Marketplace	The BIG IoT Marketplace allows Providers to register their Offer-

	ings (based on semantic descriptions) and Consumers to discover relevant Offerings (based on semantic queries) at runtime. It also provides accounting support for Consumers and Providers to track the amount of resources accessed, as well as a web portal for developers and administrators to support the implementation and management of their Applications, Services, and Platforms.
BIG IoT Service (or short Service)	A BIG IoT Service implements and uses the BIG IoT API to consume and/or provide Offerings via the BIG IoT Marketplace. A BIG IoT Service can act both as an Offering Consumer and Provider. It typically consumes basic Information or Function in order to offer "higher-value" Information or Functions on the BIG IoT Marketplace.
BIG IoT User	A User of a BIG IoT Application. A BIG IoT User is typically an employee of an Enterprise, SME or Organization (e.g. City Authority), but not limited to that.
Billing	Billing collects Charging data and creates invoices.
Charging	Charging is based on the collected Accounting data. The Charging Service multiplies the accounting data with the respective Price data of an Offering, and also takes into account special Consumer (group) accounting models, to compute the final amount to be charged.
Device-level BIG IoT enabled IoT Platform (= Device-level BIG IoT Platform or just Device-level Platform)	A BIG IoT enabled Platform that is implemented directly on a Smart Object, as opposed to on a backend or cloud infrastructure.
Endpoint	An Endpoint in the context of BIG IoT is a web based interface for consumers to access Offerings via a Provider. An Endpoint description consists of properties like Endpoint type and URI.
Function	Functionality that can be invoked by Consumers and is provided by a task on an actuator (as part of an IoT Platform) a Service that provides some computational functions or higher level functionality delegating to one or more lower level Functions
Information	Data provided to Consumers by a sensor (as part of an IoT Platform) a Service that takes one or more Information sources and combines them to provide some added value
IoT Service (or short Service)	Software component enabling interaction with resources through a well-defined interface in order to access or manipulate information or to control entities. An IoT Service can be orchestrated

	together with non-IoT services (e.g., enterprise services). Interaction with the service is done via the network. (based on [IoT-A])
IoT Platform (= Smart Object Platform)	A computing and communication system that hosts software components enabling interaction with Smart Objects in order to access or manipulate information or to control them. An IoT Platform may be implemented on a backend or cloud infrastructure, or directly on a Smart Object. Interaction with the platform is done via the network.
License	The Provider of an Offering can choose the License terms for the provided Information.
Offering	BIG IoT enables Providers to offer or trade access to Information and Functions with Consumers via the Marketplace. An Offering is defined by an Offering description, which describes a set of Resources offered on the Marketplace. It typically encompasses a set of related Information or Functions. An Offering description provides a semantic description of the Resource(s) provided to a Consumer once the Offering is accessed. The description also entails context and meta information about the Offering, including information like the Region (e.g. a city or spatial extent) where the Resource(s) relate to, the Price for accessing the Resource(s), the License of the Information provided, input & output data fields, etc.
Offering Consumer (or short Consumer)	A BIG IoT Application or Service that is interested to discover and access IoT resources in order to provide a new service or function. A Consumer discovers and subscribes to relevant Offerings via the BIG IoT Marketplace, and accesses the offered resources via the BIG IoT API.
Offering Provider (or short Provider)	A BIG IoT Platform or Service that wants to offer or trade IoT resources via the BIG IoT Marketplace. A Provider registers its Offering(s) on the BIG IoT Marketplace, and provides access to the offered resources via the BIG IoT API.
Offering Query (or short Query)	Consumers are able to discover offerings of interest on the marketplace by providing an (Offering) Query. A Query describes the properties of Offerings a client is interested in (Offering type, input & output data fields, Price, License, ...)
Physical Entity	Any physical object that is relevant from a user or application perspective. [IoT-A]
Price	The Provider of an Offering can choose the accounting model (e.g. Free or Per Month or Per Access) and amount of money (if applicable) a Consumer has to pay when accessing a Resource.
Resource	Abstraction for either Information or Function.

Smart Object (= Thing)	A Device able to compute and communicate information about itself or related artifacts (Physical Entities) to other devices or computer applications; a Smart Object is typically attached to or embedded inside a Physical Entity. Smart Objects either monitor a Physical Entity (sensing) or interact with the physical world through actuators (actuation). Those functions can be either controlled autonomously by local computations or triggered from remote.
Subscription	Agreement to access the Resource(s) of a single Offering. This comprises: a Consumer's willingness to access the Offering (he checked License, service level, rating, description, ...) the Consumer's consent to pay for the access to the Resources (according to the specified Price), if applicable
Semantic Offerings Composition Recipe (or BIG IoT Recipe)	Semantic Offerings Composition Recipe or shorter Recipe provides description of the composition of offerings. It is a specification of requirements of an added value service, and hence it represents a template that can be fulfilled by multiple offerings.
BIG IoT Semantic Core Model	BIG IoT Semantic Core Model specifies all important domain concepts in BIG IoT project including all basic conceptual entities and their relationships. This semantic model is used as basis for (1) the Offering Description to define the capabilities of offerings provided by IoT platforms or services, and (2) the underlying data model of the BIG IoT Marketplace.
Mobility Domain Model	Mobility Domain Model defines a common terms used in the mobility domain. The model aims to improve information exchange and data interoperability between mobility systems, in particular used in Internet of Things applications. This model represents an extended schema.org vocabulary for the mobility domain (mobili-ty.schema.org).
BIG IoT Semantic Application Domain Model	BIG IoT Semantic Application Domain Model defines an application-specific vocabulary that is built on both, BIG IoT Semantic Core Model and Mobility Domain Model, and can be used for annotating Offering Descriptions.
Semantic Recipe Model	Semantic Recipe Model defines a model for BIG IoT Recipes. A Recipe is a specification of requirements of an added value service, and hence it represents a template that can be fulfilled by multiple offerings. All terms and their relations, required for specifying Recipes, are defined in Semantic Recipe Model.

European Commission
Horizon 2020
European Union funding
for Research & Innovation

1. Introduction

1.1. Scope of this document

This deliverable introduces the mechanisms for enabling the interoperability for smart object platforms and services at the semantic level. A semantic model is introduced to facilitate the formal semantic descriptions of concepts and properties used in the context of the BIG IoT API. A central role plays the concept of “offering”, which represents a set of resources offered by a platform or service. The defined concepts and properties are utilized to uniquely identify and semantically annotate meaningful relationships and contexts of smart object platforms and services as well as to specify semantically defined values for relevant attributes. The interoperability of data represented in this fashion is instrumented by standardized machine-readable formats whereby consumers of these resources can autonomously interpret the meaning of the annotated data. The needed vocabulary management includes features such as the registration, retrieval, update, and removal of semantic descriptions, as well as the alignment between predefined semantic descriptions.

Another important aspect of this deliverable is the *selection* of existing higher semantic models for smart object platforms and services and their properties. Once those models are selected, a next step is the design of application specific semantic models to capture the knowledge of the use cases to be realized, which is done in Task 4.2.

1.2. Audience

This deliverable addresses the following audiences:

- **Researchers, developers and integrators within the BIG IoT consortium**, which will use this deliverable and the therein defined ontology as shared conceptualisation, i.e. shared vocabulary and taxonomy of the BIG IoT domain.

- **Platform owners** who wish to join BIG IoT will be able to use the offering description to annotate their offerings. These descriptions should comply with the semantic model proposed within BIG IoT.
- **Members of other Internet of Things** (IoT) communities and projects (such as projects of the IERC cluster) can take this document as an initial reference or inspiration to design and implement their own marketplace that also stores resources that are semantically annotated.
- **Open call participants** will be able to understand better the technical details needed for them to join the BIG IoT consortium.
- **Standardization bodies**. As a public document, this deliverable will be accessible by any groups listed above and including standardization bodies. The content of this deliverable is also part of some extensions towards standardization activities around the semantic representation, which standard organizations and its individuals can also benefit from the content introduced.

1.3. Relation to other Deliverables

The tasks of WP3 and WP4 are highly interrelated, however, their deliverables have each their own, clear responsibilities. The figure below follows the general BIG IoT architecture description in D2.4 and illustrates at hand of this architecture the scope of each of the different deliverables of the tasks 3.1, 3.2, 4.1, 4.2, and 4.3.

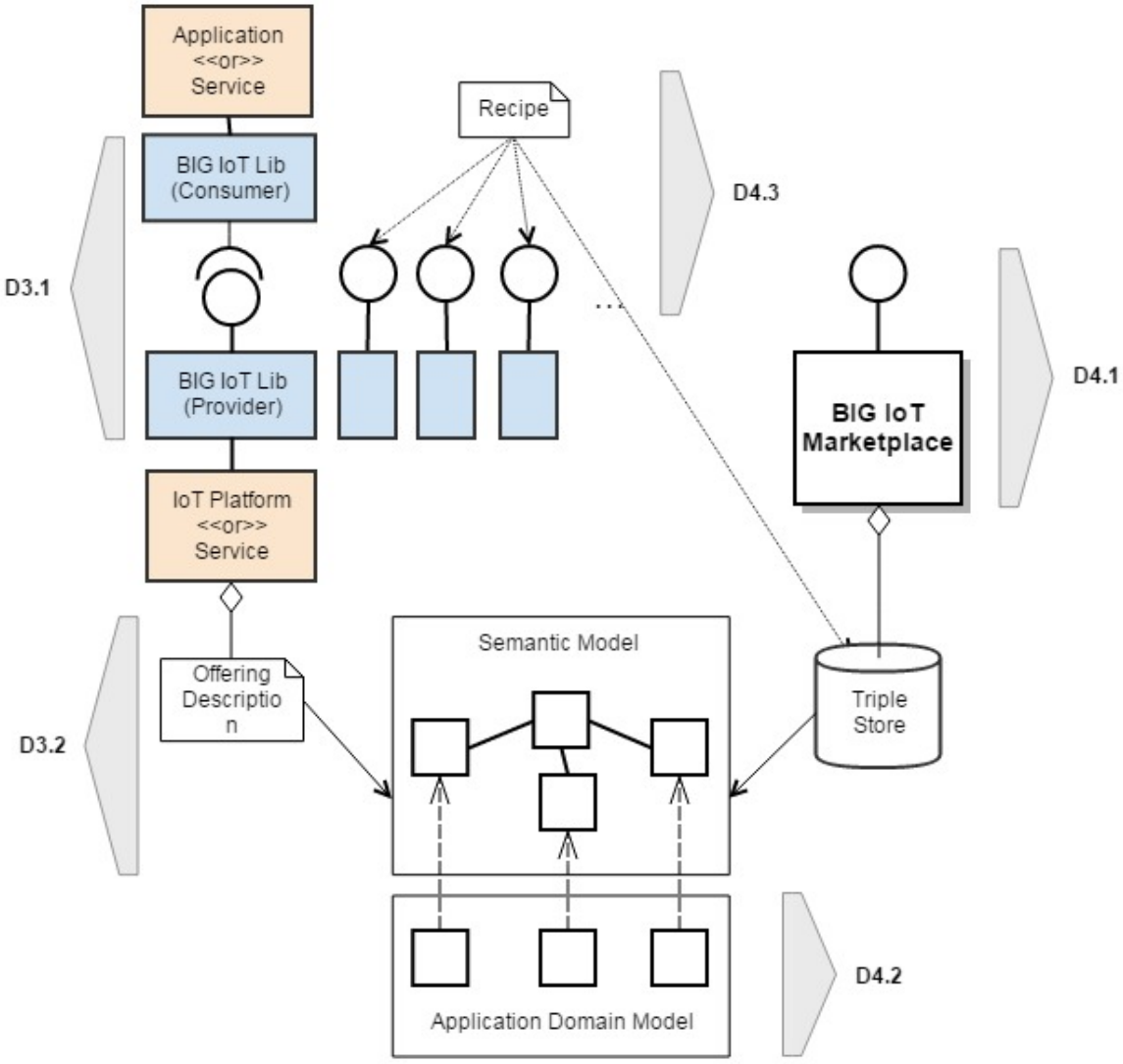

Figure 1: Relation of deliverables.

D3.1 covers the description of the BIG IoT API. This entails a specification of the Web API (e.g., interaction model, and encodings) as well as description of the programmatic API that is implemented as part of consumer and provider lib of the SDK.

D4.1 describes the architecture of the BIG IoT Marketplace. This includes the general design, the workflows of interactions, the GraphQL-based API of the market-

place, as well as the user interface of the portal to utilize the marketplace. Also contained in this deliverable is a description on how to map between the GraphQL API of the marketplace frontend and its SPARQL based triple store backend.

D3.2 describes the model and schema of the core semantic model of BIG IoT. This semantic model is used as a basis for (1) the Offering Description to define the capabilities of offerings provided by IoT platforms or services, and (2) the underlying data model of the BIG IoT Marketplace.

D4.2 builds up on the core semantic model of D3.2 and defines the application domain model, which specifies a vocabulary of terms that can be used in Offering Descriptions, and in the triple store of the marketplace.

D4.3 addresses the orchestration and composition of the offerings of BIG IoT providers (i.e., platforms or services). A composition of offerings can be specified by defining a Recipe. The semantic model and examples of such recipes is the scope of this deliverable.

1.4. Structure of this Document

Section 2 outlines the background knowledge and reviews related work. The first part (Section 2.1) contains the introductory section with references aiming for the common understanding about interoperability challenges in IoT. The second part (Section 2.2) introduces the state of the art of existing semantic models for IoT platforms and services. This mainly includes existing works towards adding semantics to the Internet-of-Things, as well as an overview of existing IoT platforms together with a brief outline how interoperability and heterogeneity are addressed in these platforms. Section 3 presents the generic semantic model of BIG IoT resources. The rationale is that the model follows the recommended best-effort practice to reuse existing, popular ontologies/vocabularies as much as possible. Section 4 represent a crucial part of the deliverable, introducing the BIG IoT Offering Description. The conclusion and future works will be given in Section 5.

2. Background and Related Work

IoT (Internet of Things) refers to objects (“things”) and the virtual representations of these objects on the Internet. It defines how the things are connected to the Internet and how those things “talk” amongst other things and communicate with other systems in order to expose their capabilities and functionalities. The IoT world is not only linking connected devices by using the Internet; it is also Web-enabled data exchange in order to enable systems with more capacities to become “smart”. IoT aims to integrate the physical world with the virtual world by using the Internet as the medium to communicate and exchange information.

IoT is supported by continuous however, heterogeneity of underlying devices and communication technologies and interoperability in different layers, from communication and seamless integration of devices to interoperability of data generated by the IoT resources, is a challenge for expanding generic IoT solutions to a global scale.

In a white paper on interoperability [1] it is discussed that many layers of interoperability exist:

- Technical Interoperability
- Syntactical Interoperability
- Semantic Interoperability
- Organizational Interoperability
- Dynamic interoperability
- Static interoperability

Discovery, understanding, and collaboration at this level require more than just an ability to interface and to exchange data. Whereas interoperability is “the ability of two or more systems or components to exchange data and use Information" [2] , semantic interoperability “means enabling different agents, services, and applications to

exchange information, data and knowledge in a meaningful way, on and off the Web" [2] [3].

Semantic interoperability is achieved when interacting systems attribute the same meaning to an exchanged piece of data, ensuring consistency of the data across systems regardless of individual data format. This consistency of meaning can be derived from pre-existing standards or agreements on the description and meaning of data or it can be derived in a dynamic way using shared vocabularies either in a schema form or in an ontology-driven approach.

In this section, we present various of these semantic interoperability challenges to overcome to deliver the potential of innovative services that IoT is looking for.

2.1. Challenges for Semantic Interoperability in IoT

2.1.1. Semantics Interoperability

The overall challenge in interoperability is first to ensure technical interoperability from technologies to deliver a mass of information and then complementary challenges are for the information to be understood and processed. The tables below present a summary of the challenges for technical and semantic interoperability, as reported by the European Research Cluster on the Internet of Things in [4].

Table 1 IoT Semantic Interoperability Challenges/Requirements

Requirement(s)	Rationale
Best practices Avoid spreading effort in addressing interoperability for internationally adopted protocols.	Use clear models development and testing methodologies leading to improve quality while reducing time and costs in a full chain optimized development cycle. Define if needed specifications to improve interoperability.
Validation of specifications Reduce ambiguities in specifications and development time.	Specifications development time could be too long. Ambiguities in specifications could lead to major non-interoperability issues. Quality, time and cost factors lead to the needs of models and automation.
Test specifications Provide market accepted test specifications ensuring minimum accepted level of interoperability.	The absence of test specifications leads inevitably to different specifications implementation and interoperability issues. Development of test specifications is often too expensive for a limited set of stakeholders and effort should be collectively shared.
Tools and validation programs Develop market accepted and affordable test tools used in market accepted validation programs.	Development of test tools is expensive. Available test tools may not be sufficient and not optimized for all tests needs. The full chain of specifications to tool development not considered. Providing final confidence to end users with consistent tests not always considered.
Integration Support multiple resources (sensors, actuators) and relevant types of data sources (independently of vendor and resource location).	Enable scalable sharing and integration of distributed data sources. All IoT applications involve multiple heterogeneous devices. Orchestrate resources in order to automatically formulate composite workflows as required by end-user applications.
Annotation Enable the (automated) linking of relevant data sources.	Linking of data sources facilitates application integration and reuse of data. Enable interactions between sensors and between IoT services. Built on the standards.

Management Enable the creation and management of virtual sensors and virtual resources based on the composition and fusion of multiple data sources.	Application development and integration involve multiple distributed and heterogeneous data sources to be processed in parallel.
Discovery Provide the means for discovering and selecting resources and data sources pertaining to the application.	End users need a high-level interface to be accessed. Provide the means for describing/formulating IoT services and applications according to high-level descriptions.
Analysis and Reasoning Provide analytical and reasoning tools on top of semantic level capabilities.	IoT addresses large-scale environments with numerous resources featuring different functionalities and capabilities.
Visualisation Optimize usage of info across multiple users sharing these resources.	For a better understanding and reporting of resource interactions or interactions between services.

2.1.2. Why using Semantics?

Many of the problems present in current Internet technology are going to remain in the Internet of Things systems and mainly generated by interoperability problems, thus there are three persistent problems:

1. Users are offered relatively small numbers of Internet services, which they cannot personalize to meet their evolving needs; communities of users cannot tailor services to help create, improve and sustain their social interactions;
2. The Internet services that are offered are typically technology-driven and static, designed to maximize usage of capabilities of underlying network technologies and not to satisfy user requirements per se, and thus cannot be readily adapted to their changing operational context;

3. Network operators cannot configure their networks to operate effectively in the face of changing service usage patterns and rapid networking technology deployment; networks can only be optimized, on an individual basis, to meet specific low-level objectives, often resulting in sub-optimal operation in comparison to the more important business and service user objectives.

Towards Internet of Things, there is an increasing focusing on how to evolve communications technologies to enable the "Internet of Things", but addressing the evolution of networking technologies in isolation is not enough; instead, it is necessary to take a multi-domain adaptable broader view of the evolution of services, their requirements and the heterogeneous infrastructures. Find solutions to information interoperability challenge issues, Internet of Things systems must be able to exchange information and customize their services.

2.2. Semantic Models for Smart Object Platforms and Services in Internet of Things

There are several semantic descriptions designed for the IoT domain. The SSN ontology [3] is one of the most significant and widespread models to describe sensors and IoT related concepts. The SSN Ontology provides concepts describing sensors, such as outputs, observation value, and feature of interest. However, it is a detailed description, containing concepts and properties that enable flexible descriptions over a very wide range of applications, but including non-essential components for many use cases that can make the ontology heavy to query and process if it is used as it is.

The IoT-A model and IoT.est [5] are some of the projects that extend the SSN ontology to represent other IoT related concepts such as services and objects in addition to sensor devices. IoT-A provides an architectural base for further IoT projects. The only implementation of a purely IoT-A semantic model known by the authors is described in [6]. The IoT-A model is overly complex for fast user adaptation and re-

sponsive environments. The IoT.est model extends the IoT-A model with extended service and test concepts.

The Open Geospatial Consortium's (OGC) Sensor Web Enablement [7]initiative has defined a framework of services and data models to enable the publication, discovery and access to sensor data. The key service thereby is the Sensor Observation Service (SOS) [8] that offers an XML-based protocol and application programming interface (API). The usage of XML means that ad-hoc mapping of different schemas is needed to integrate data. Here, neither semantic interoperability is enabled nor reasoning is supported. As an attempt to account for that issue, a semantic extension to SOS, SemSOS [9], has been developed.

Semantic Web technologies address such weaknesses by using machine-understandable descriptions of resources, which do not require any ad-hoc schema. The de-facto standard as a representation model is RDF and the meaning of each term can be determined automatically by checking the corresponding vocabulary definition. The Semantic Sensor Web [10] studies the application of these technologies to enable the Sensor Web. The OGC defined a vocabulary for sensors, SensorML, which has also been included into the SSN-XG ontology [3].

Several research projects have followed the Semantic Sensor Web vision and are currently using the W3C SSN-XG ontology, e.g. SENSEI [11], the Spanish Meteorological Agency, Sensor Masher [12] , the work done by the Kno.e.sis Center, CSIRO, the 52°North initiative, SemSorGrid4Env, as well as the Exalted project.

3. Semantic Model of BIG IoT Resources

3.1. BIG IoT Domain Concepts

This subsection is intended to clarify the core terminology used during in the BIG IoT project. This initial step is intended to clarify all of the important terms used, in order to minimize misunderstandings when referring to specific parts involved in the generation of data and the BIG IoT concepts. The following definitions (listed below) were set regarding the domain area of BIG IoT (from D2.4.a and D4.1.a)

In the following sections, we will propose the BIG IoT Semantic Core Model based on the terminology and definitions table. These core concepts and their relationships are explained in more detail, including additional sub-concepts.

3.2. BIG IoT Semantic Core Model

3.2.1. Schema Namespaces

This Section will list all the potential namespaces that will be used in BIG IoT Semantic Core Model.

Table 2. Schema Namespaces

Pre-fix	Ontology/Language	Namespace
bigiot	BIG IoT Ontology	http://big-iot.eu/core#
sche-ma	Schema.org ontology	http://schema.org/
td	Things description	http://w3c.github.io/wot/wot.owl#
xsd	XML schema Definition	http://www.w3.org/2001/XMLSchema#
rdf	RDF Concepts Vocabulary	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	RDF Schema ontology	http://www.w3.org/2000/01/rdf-schema#

3.2.2. Modularization of BIG IoT Semantic Core Model

One of the key aspect when designing a semantic model is reusing of knowledge. Once a semantic model is created for a domain, it should be (at least to some degree) reusable for other applications in the same domain. To simplify both semantic model development and reuse, a modular design is beneficial. Based on the project specification and the domain model in Deliverable 2.4.1, the semantic models can be modularized according to their scope, as follow:

- Organization module
- Provider module
- Offering module
- Consumer module

- Query module

Figure 2 illustrates the high level of BIG IoT Semantic Core Model which includes all the basic conceptual entities and their relationship of all modules. Details of each module will be presented in the next subsections.

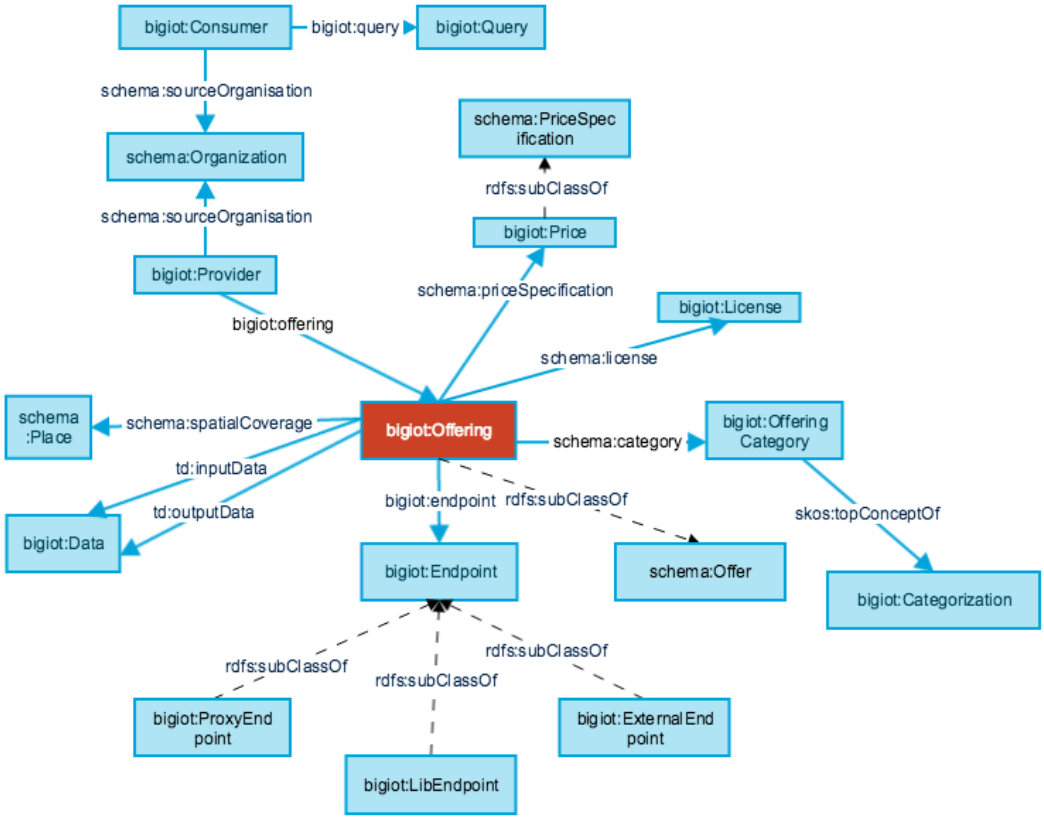

Figure 2. BIG IoT semantic model

3.2.3. Provider Module

A provider can be either a platform or a service instance that offers available offerings. A provider is described through **bigiot:Provider** class. At this stage, each provider will have a name (*schema:name*) and id (bigiot:providerId) and its organization as shown in Table 3. More information about the provider can be added in the future.

Table 3. Provider Properties

Property Name	Data Types	Optional/ Mandatory	Example	Description
schema:name	string	m	"UPC"	Name of the provider
bigiot:providerId	string	m	"145sfwr"	Provider id
schema:sourceOrganization	schema:Organization	o	"Barcelona_City"	A provider's organization
schema:description	string	o		A description of the provider

3.2.4. Organization Module

A provider may also describe its organization. The provider's organization will be an instance of the **schema:Organization** class. As illustrated in Figure 2, the connection between the Provider and the Organization is the schema:sourceOrganization property. Table 4 below presents some basic properties of the organization, which are taken from the schema:Organization.

Table 4. Organization properties

Property Name	Data Types	Optional/ Mandatory	Example	Description
bigiot:organizationId	string	m	"Barcelona_City"	Organization id
schema:name	string	m	"Barcelona City Council"	Name of the organization
schema:address	string	o		Physical address of the organization
schema:contactPoint	schema:ContactPoint	o		A contact point for organization
schema:description	string	o		A description of the organization

3.2.5. Offering Module

As illustrated in Figure 2, the Offering module represents the initial conceptualisation which is built around the Offering and its metadata. All the core concepts of this module are defined as follows:

A provider registers its offerings on the marketplace by providing an offering description. An offering description is an instance of the **bigiot:Offering** class. It contains the information about the service area (schema:region), type of the offering (td:resourceType). All relevant communication metadata are provided on how the offering can be accessed through the endpoint which will be instantiated from

bigiot:EndPoint class. This instance includes several properties, such as the end-point URL (schema:url), streaming (td:supportStreaming) and subscription supports (td:supportSubscription). The offering description also includes an identifier (bigiot:offeringId) and a name of the offering (schema:name).

Each offering will have the price information for accessing the offering's resources. This information plays an important role in the BIG IoT billing module. We use the property schema:priceSpecification for linking **bigiot:Offering** and **bigiot:Price** classes. The **bigiot:Price** class is a subclass of **schema:PriceSpecification**. Therefore, the price description will have the currency (schema:currency) and the amount of money (schema:amount). The total cost will be determined based on the accounting model (bigiot:accoutingModel) that specifies the method for calculating (PER_ACCESS, FREE or PER_MONTH).

Details of all classes and their properties in the Offering module are presented in Section 4 Offering Description.

3.2.6. Consumer Module

A consumer can be either an application or service instance that requires access to IoT resources in order to implement an intended service or function. In the consumer model, we create the **bigiot:Consumer** class that represents the BIG IoT consumers. Same as the provider, the consumer is also linked to the Organisation. The Table 5 below presents some basic properties of the bigiot:Consumer.

Table 5. Consumer properties

Property Name	Data Types	Optional/ Mandatory	Example	Description

Property Name	Data Types	Optional/ Mandatory	Example	Description
schema:name	string	m	"Consumer A"	Name of the consumer
bigiot:consumerId	string	m	"cons_A"	Consumer id
schema:sourceOrganization	schema:Organization	o	"Company A"	A consumer's organization
schema:description	string	o		A description of the consumer
bigiot:query	bigiot:Query	o		Query to BIG IoT marketplace of consumer

3.2.7. Query Module

The query module presents an abstraction for a query for Offerings. It is used to describes the properties of Offerings a consumer is interested in (Offering type, input & output data, price, license, ...). Through the BIG IoT marketplace, the consumer discovers the offerings of interest by providing an (offering) query. In this schema, the query information can be described through the **bigiot:Query** class. For example, a consumer can register a query by providing a description of the desired resources (such as the type of parking information), and also define the maximum price, the desired license types, the region, etc. The properties of bigiot:Query is shown in the Table 6 below.

Table 6. Query properties

Property Name	Data Types	Optional/ Mandatory	Example	Description
bigiot:queryId	string	m	"query1"	Query id
bigiot:queryTemplate	spin:Query	m		A query template. For example, the Exchange query or Offering query…
schema:description	string	o		A description of the query
schema:name	string	m	"Parking Query"	Name of the query

4. Offering Description

To reach semantic interoperability between different IoT domains and platforms there has to be a clear understanding and an agreement of a semantic model. In this section we introduce a semantic concept that allows to describe the capabilities and interfaces of a BIG IoT Provider and how this information can be used for discovery and access purpose.

The sections start with a motivation based on a short reflection of the BIG IoT do-main model. There we will introduce a semantic description that is called the Offering Description. Next, we give an overview of the goals and requirements of this description.

Please note, the Offering Description will be improved over the BIG IoT project runtime based on the implementation experiences as well as of the development of the Thing Description that is currently be standardized by the W3C Web of Thing working group (see D6.2). The following sub-sections reflect the current working assumptions in BIG IoT.

4.1. Motivation and Offering Description Concept in BIG IoT

Each provider within the BIG IoT ecosystem is intended to provide some valuable data and/or functionalities that can be used by another participant as the consumer. To reach interoperability between these participants we need a common understanding what and how a provider can offer and how the consumer can interpret it and how to setup the right interaction. For this common understanding there is an agreed description needed which is reflected on a shared model. This, the BIG IoT Semantic Core Model, was introduced in the previous Section 3. All the data and functionalities that are served by a BIG IoT Provider are reflected within the Offering

concept. To exchange this information, BIG IoT introduces the Offering Description (OD).

The role of the OD within the BIG IoT ecosystem is illustrated in Figure 3: As described in the BIG IoT domain model (see D2.4.a) there are 3 major parties, namely Provider, Consumer, and the BIG IoT Marketplace with its eXchange. The starting point is the Provider that serves a number of information/properties and/or functions/actions. Each of them is reflected by the OD.

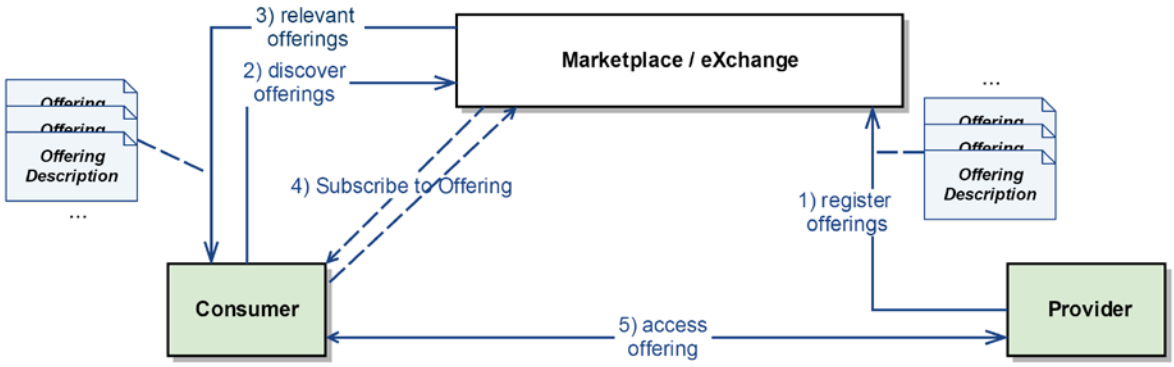

Figure 3. Role of the Offering Description within the BIG IoT ecosystem

The following steps will be:

- One or more ODs will be register at BIG IoT Marketplace by the Provider. There, the ODs will be stored and managed within the eXchange (RDF Store).
- A consumer is interested in some particular kind of offerings if they exist. In that case, the consumer starts to make a query / discovery at the BIG IoT Marketplace / eXchange. Within the query concrete details are reflected which the searched OD should fulfil.
- If there is a match for one more ODs in the eXchange, those are returned to the Consumer to be as relevant for its search.

- for all offerings relevant for consumer's application, consumer subscribes to them on the marketplace where a security token is returned (see D3.3.a)
- Within the OD there all metadata provided that is needed to access the provided Offering by the corresponding Provider

In the next sections, we will go in more detail about the goals, requirements, and contents of the OD as well as how the serialization format is used in the BIG IoT ecosystem.

4.2. Goals of Offering Description

As already reflected in the previous sub-section, the Offering Description is designed with three major goals in mind:

1. reflecting the latest offered resources (Information/Properties and Functions/Actions) of the Provider which is provided in the marketplace/eXchange
2. defining the semantics/meanings of the offering, which can then be discovered through querying
3. comprising all relevant information to setup a communication between Consumer and Provider

Based on these goals of the OD the interoperability between different IoT platforms should be guaranteed. In the following, the details about OD's content are presented.

4.3. Basic Structure of the Offering Description

4.3.1. Offering Object

The basic structure of the OD is derived from the BIG IoT Semantic Core Model (see Section 3.2 or Figure 2). The central point is the Offering class.

An offering has several vocabulary fields, listed in the following tables. There you find the vocabulary/Property name with its associated context/namespace as prefix (also see Section 3.2.1), its simplified form as used field name in a OD serialization (see next section), the associated data type, if this information is optional or mandatory as well as a detail description of this vocabulary. Vocabularies which are de-fined complex (contain embedded vocabularies) are explained in separated tables:

Table 7. Offering object

:

Property Name	Field Name	Data Types	Optional /Mandatory	Description
schema:name	Name	string	m	Name of the offering
bigiot:offeringId	Id	string	m	Concatenation of the ID of organization, provider, and offering.
schema:category	Category	string	o	String (compact URI) referring to one of the BIG-IoT Categories (see Section **Error! Reference source not found.**) or URI pointing to an external category.
td:hasInput / td:hasOutput	inputData / outputData	bigiot:Data	o	Gives a detailed description of the transported payload data between consumer and provider for this offering.

Property Name	Field Name	Data Types	Optional /Mandatory	Description
				This includes at least the name of a data variable and, optionally, its value type (e.g., Boolean, number, etc.) and a semantic annotation by the usage of the rdfType which can point to an RDF concept. The "members" field can be used to define complex types (nested structures of input/outputData). See Data object definition below.
bigiot:endpoint	endpoint	bigiot:Endpoint	m	The endpoint includes all relevant information which is required to setup right communication to the provider's offering. This includes the description of what kind of endpoint is the underlying infrastructure (e.g., HTTP, Web Socket, CoAP, etlc), the URI for addressing the corresponding resources, the method which as to ap-

Property Name	Field Name	Data Types	Optional /Mandatory	Description
				plied (e.g., GET, PUT, etc.), mediaType used for the payload data (e.g., JSON, XML, ...), the different BIG IoT access interfaces (e.g., is the BIG IoT Lib used for the resource access), and if streaming and subscription is supported. See Endpoint object definition below.
schema:spatialCoverage	region	schema:Place	o	This optional field gives some location information that is relevant for that offering such as geographical coordinates (lat=latitudes and lng=longitudes) or by generic place by providing the city name.
schema:priceSpecification	price		m	See Price object definition below.
schema:license	license	bigiot:License	o	See License object definition below
-	resourceType	string ("Information", "Property",	m	Characterized this offering if it only provides some data

Property Name	Field Name	Data Types	Optional /Mandatory	Description
		"Action", "Event")		(information/property) or some actions (functions/processes).

4.3.2. Endpoint Object

Table 8 Endpoint object

Property Name	Field Name	Data Type	Optional/Mandatory	Description
-	uri	string (URI)	m	Endpoint URI.
rdf:type	type	string ("HTTP", "WS", "CoAP", "MQTT", "REST-HTTP", "REST-CoAP", or "REST-oDATA")	m	Protocol to use when querying the endpoint.
-	method	string ("GET", "PUT", "POST", ...)	o	Method to use when querying the endpoint. Methods are proto-

Property Name	Field Name	Data Type	Optional/Mandatory	Description
				col-specific.
-	mediaType	string ("application/json", "application/xml", ..)	o	Media type(s) to expect for the representation of the offering resource. Possible media types are listed in a IANA registry.
-	accessInterfaceType	string ("BIGIOT_LIB_ACCESS", "BIGIOT_EXTERNAL_ACCESS", "BIGIOT_PROXY_ACCESS", ...)	o	Type of the access interface. See D2.4.a: High-level Architecture for more details.
-	supportsStreaming	boolean	o	
-	supportsSubscription	boolean	o	

4.3.3. Input/Output Data Object

Table 9 Input/Output Data Object

Property Name	Field Name	Data Type	Optional/Mandatory	Description
schema:name	name	string	m	Name of the input/output data
-	valueType	string ("string", "number", "boolean", "null")	o	Data type, as interpreted in JSON.
bigiot:rdfType	rdfType	string (URI)	o	Semantic entity that relates to
-	members	bigiot:Data	o	Nested input/output data objects. The structure of each nested object should be the same as for their parent (with name, valueType, rdfType, members). Data defining nested data is expected to be serialized in JSON.

4.3.4. Price Object

Table 10 Price Object

Property Name	Field Name	Data Type	Optional/Mandatory	Description
bigiot:accountingModel	accounting	string	m	e.g. per access or flat rate.
schema:price	amount	double	m	amount for each transaction as per accounting
schema:priceCurrency	currency	string	m	currency, in ISO 4217 format: "EUR", "USD", "JPY", etc.

4.3.5. License Object

Table 11 License Object

Property Name	Field Name	Data Type	Optional/Mandatory	Description
bigiot:licenseType	type	string	m	e.g. "OPEN DATA LICENSE".
-	description	string	o	free text description of the license, if of a specific license type

4.4. Encodings of Offering Description

The BIG IoT Offering Description (OD) relies on RDF as an underlying data model. As a current serialization format of RDF, JSON-LD has been proposed which provides the semantic description of a BIG IoT provider. For transportation of this information between a Provider and the eXchange, GraphQL is used.

4.4.1. JSON-LD Encoding

JSON-LD is a relatively new serialization format that was standardized by the W3C in 2014 [13]. JSON-LD has been proposed in BIG IoT in order to benefit from both widely used JSON-based format and JSON-LD's concept of @context. This latter mentioned mechanism provides the power of mapping from JSON to an existing RDF model such as from smart parking or smart vehicle routing. BIG IoT provides a default JSON-LD context file located at http://gib-iot.eu/ctx, that maps RDF properties and classes to the field names given above. Given this default context, the basic OD structure by using the JSON-LD encoding follows from the object definitions. It is illustrated in the following snippet:

Table 12 Offering Description JSON-LD Serialization

Offering Description JSON-LD Serializaton

```
{
  "@context": "http://big-iot.eu/ctx",
  "name": "...",
  "category": "...",
  "inputData": [],
  "outputData": [],
  "endpoint": {},
  ...
}
```

In the first line, the context is introduced for this JSON-LD document. BIG IoT's default context (http://big-iot.eu/ctx) shall always be used to identify this document as a BIG IoT Offering Description. Depending on the Offering's context application, additional namespaces can be added (e.g., "datex" for smart parking). After the context field the presented mandatory fields (e.g., name, id, resourceType) and optional fields (e.g., category) are used as presented in Table 8. Complex structures such as endpoint and region have nested field definitions.

4.4.2. GraphQL Encoding

As an alternative RDF-based serialization format, BIG IoT introduces a new mechanism with GraphQL [14]. In general, GraphQL is a data query language using JSON-style syntax and provides an alternative to REST and ad-hoc web service architectures. GraphQL is one of the main components in the BIG IoT marketplace and is introduced there in more detail (see D4.1.a). We defined the GraphQL schema in that way, that we can embed all relevant information of the OD in a GraphQL document. In the following, a snipped is shown:

Table 13 Offering Description GraphQL Serialization

Offering Description GraphQL Serializaton

```
mutation registration {
  createOffering ( newOffering: {
    {
      context: "http://big-iot.eu/ctx",
      name: "...",
      category: "...",
      inputData: [],
      outputData: [],
      endpoint: {},
      ...
    }
  )
}
```

Since GraphQL does not have the semantic *@context* concept as JSON-LD, we introduced a workaround by using the key-value pair. Using this, the same meaning and usage can be activated as it is done in an OD represented in JSON-LD. In the same manner, the rdfType is introduced as a replacement of the *@type* field mechanism in JSON-LD. Since GraphQL is structured similarly to a JSON document, all defined fields as listed in Table 8 occur in the same manner as in the JSON-LD document.

4.5. Examples of Offering Descriptions

Example 1

The example below gives one possible definition for an offering to list available parking spots in a given area. This offering is described in D5.1 and is part of the different pilot scenarios. This Offering object could be created entirely programmatically through the BIG IoT SDK (see tutorial section in D3.1.a).

Table 14 Parking Spot Availability Offering (JSON-LD)

Parking Spot Availability Offering (JSON-LD)

```
{
  "@context": "http://big-iot.eu/ctx",
  "name": "Parking Spot Availability Service",
  "inputData": [
    { "name": "longitude" },
    { "name": "latitude" },
    { "name": "radius" }
  ],
  "outputData": [
    { "name": "longitude" },
    { "name": "latitude" },
    { "name": "parkingSpotID" },
    { "name": "timestamp" },
    { "name": "status" }
  ],
  "endpoint": {
```

```
        "uri": "http://example.org/pspot",
        "method": "HTTP_GET",
        "accessInterfaceType": "BIGIOT_LIB"
      },
      "license": { "type": "OPEN_DATA_LICENSE" },
      "price": {
        "amount": 0.002,
        "currency": "EUR",
        "accounting": "PER_ACCESS"
      },
      "region": "http://sws.geonames.org/3128760/"
    }
```

Example 2

Also taken from the list of pilot offerings as specified in D5.1, we show here another OD example for a Parking Spot WMS Service. This offering builds on top of the offering described in Example 1. It aggregates information about parking spots in a specific area in order to display it on a geographical map (conform to an OGC Web Map Service, WMS).

Table 15 Parking Spot WMS Offering (JSON-LD)

Parking Spot WMS Offering (JSON-LD)

```
    {
      "@context": ["http://big-iot.eu/ctx",
                   {"mobility" : "http://big-iot.eu/mobility#"}],
      "name": "Parking Spot Web Map Service Offering",
      "category": "mobility:Parking",
      "inputData": [
        {“name": "area", "rdfType": "schema:GeoCircle",
          "members": [
            {“name": "radius", "valueType": "number","rdfType":"sche
ma:geoRadius" },
            {“name": "coords", "rdfType": "schema:GeoCoordinates",
              "mem-
bers":[{ "name": "latitude", "rdfType": "schema:latitude", "valueTyp
e" : "number"},

{“name": "longitude", "rdfType": "schema:longitude", "valueType" : "
number"}]}]}
```

```
      ],
      "outputData": [
        {“name": "geoCoordinates", "rdfType": "schema:GeoCoordinates
",
          "mem-
bers":[{ "name": "latitude", "rdfType": "schema:latitude", "valueTyp
e" : "number"},
                  {“name": "longitude", "rdfType": "schema:longitu
de", "valueType" : "number"}]},
        {“name": "parkingSpotID", "valueType": "string", "rdfType":"
mobility:parkingSpaceOrGroupIdentifier" },
        {“name": "timestamp", "valueType": "string", "rdfType":"mobi
lity:parkingSpaceStatusTimeStamp" },
        {“name": "status", "valueType": "string", "rdfType":"mobilit
y:ParkingSpaceStatus" }
      ],
      "endpoint": {
        "uri": "http://example.org/pspot",
        "method": "HTTP_GET",
        "accessInterfaceType": "BIGIOT_LIB"
      },
      "license": "OPEN_DATA_LICENSE",
      "price": {
        "amount": 0.02,
        "currency": "EUR",
        "accounting":"PER_ACCESS"
      },
      "region": "http://sws.geonames.org/3128760/"
    }
```

This example illustrates the use of the “members” key to specify complex input/output data structures. For example, this second offering has a single input parameter corresponding to an area specification (a circle). In JSON, this input parameter is expected to have the following structure, while accessing the offering:

Table 16 Input parameter at access time for the Parking Spot WMS Offering

Input parameter at access time for the Parking Spot WMS Offering

```
{
  "area": {
    "radius": 50,
    "coords": {
      "latitude": 24.5
      "longitude": 35.1
    }
  }
}
```

This OD is also semantically annotated with terms from the mobility domain. More details are given in D4.2 (including this example). Although they were required to provide a consistent modelling of the semantics of the various offerings defined for the pilots, complex data structures are not supported yet by the BIG IoT SDK.

4.6. Analogy to W3C Web of Thing & Thing Description

Deliverable 6.2.a (D6.2.a) presents the concept of the W3C Thing Description (TD). In general, the TD is defined in a very generic way which can be more specialized for any (domain) applications. The TD can be used to define the capabilities and interfaces of any kind of IoT representative such as from devices (physical or virtual), platforms, and cloud solutions. In one of the use cases, the TD can be used for the same discovery purpose as introduced in the OD (see Section 4). To be compatible with a standard, the BIG IoT OD is mainly building upon of the current working assumption of W3C TD. Many concepts and definitions of the W3C TD are reused or mapped to the OD concept. The commonly used basic vocabularies are:

- name
- inputData

- outputData
- valueType
- mediaTypes

Specialized vocabularies due to the BIG IoT ecosystem such as category, accessInterfaceType, and price (and its content) are added in addition to the core TD vocabulary set. In the following, an example is shown, how the OD is directly represented as a TD (please note, the RDF model is identical to the OD and TD representation):

Table 17 W3C Thing Description for Parking Availability

W3C Thing Description for Parking Availabliity

```
{
   "@context": [
      "http://w3c.github.io/wot/w3c-wot-td-context.jsonld",
      {"schema" : "http://schema.org/",
      "bigiot":"http://big-iot.eu/core#",
      "mobility": "http://big-iot.eu/mobility#"}
   ],
   "@type": "Thing",
   "name": "Parking Spot Availability Service",
   "interactions": [{
      "@type": [
         "Property",
         "mobility:parkingSpaceAvailable"
      ],
      "name": "available",
      "inputData": {
         "valueType": {"properties": {"area": {"properties": {
         "radius": {"type": "number"},
         "coords": {"properties": {
            "latitude": {"type": "number"},
            "longtitude": {"type": "number"}
         }}
      }}}}},
      "outputData": {
         "valueType": {"properties": {
         "parkingSpotID": {"type": "string"},
         "timestamp": {"type": "string"},
         "status": {"type": "string"},
         "coords": {"properties": {
```

```
            "latitude": {"type": "number"},
            "longtitude": {"type": "number"}
        }}
    }}},
    "writable": false,
    "links": [{
        "href": "http://example.org/pspot",
        "mediaType": "application/json"
    }],
    "schema:license": "OPEN_DATA_LICENSE",
    "schema:priceSpecification": {
        "schema:price": 0.02,
        "schema:priceCurrency": "EUR",
        "bigiot:accountingModel":"bigiot:PER_ACCESS"
    },
    "schema:spatialCoverage": "http://sws.geonames.org/3128760/"
  }]
}
```


The BIG IoT software libraries will support both variants of encodings for offerings, as an Offering Description representation (JSON-LD or GraphQL) or as a W3C Thing Description representation.

5. Conclusions and Future Work

This deliverable presents the results of the 1st iteration of the work related to the specification of a Semantic Core Model to describe BIG IoT resources and to enable the semantic interoperability for smart object platforms and services in a BIG IoT ecosystem, which has been carried out from M5 to M12 of the project. The work was carried out by Task 3.2, but also involved interactions and coordination with representatives from other project activities, in particular Task 4.2 (Semantic interoperability for smart object platforms and services), with inputs from Task 2.3 (Requirements analysis and specification), and will be reflected in the development of the work of all the different tasks of WP3, WP4 and WP5.

We presented the first iteration process act to define the core model and specifications of the BIG IoT Ecosystem domain. In particular, we presented the latest version of the BIG IoT Semantic Core Model, revolving around the concept of Offering. It

captures most of the semantics associated to offerings defined during specification of BIG IoT's high-level architecture, with respect to endpoint definition, billing, licensing as well as offering providers. The document further describes how to exchange information about offerings, using the so-called Offering Description (OD) format, which is aligned with the idea of Thing Description, as specified by the W3C Web of Things interest group.

At the time of the writing, ODs can be generated and processed by the BIG IoT SDK. However, the OD format remains a work in progress. First, endpoint definition contains minimal information about how to access to the exposed data, which does not cover the case where a consumer uses external access, without SDK (mode 3 access). Because the endpoint definition is likely to change in future versions, the semantic core model lacks precise semantics for it in the present version. Second, although semantic annotations have already been included in the ODs provided in this release, they are not exploited by the SDK at their full potential. It has been observed throughout the modeling work that the semantics for atomic pieces of data exchanged at access time (input and output data) deserved more effort. The modeling of input and output data will be the main element to consider for the next release.

The 2nd and 3rd evolution of these application domain models will be reported in next deliverables D3.2.b and D3.2.c.

6. References

Hans van der Veer and Anthony Wiles, "Achieving Technical
1] Interoperability - the ETSI Approach," in *ETSI*, 2008.

Mike Ushold, Christopher Menzel, and Natasha Noy. Semantic Integration
2] & Interoperability Using RDF and OWL. [Online]. https://www.w3.org/2001/sw/BestPractices/OEP/SemInt/

M. Compton et al., "The SSN ontology of the W3C semantic sensor
3] network incubator group," *JWS*, 2012.

"IoT Semantic Interoperability: Research Challenges, Best Practices,
4] Recommendations and Next Steps," 2015.

Wei Wang, "A comprehensive ontology for knowledge representation in the
5] internet of things," *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, 2012.

Suparna De, "An internet of things platform for real-world and digital
6] objects," in *Scalable Computing: Practice and Experience*, 2012, pp. 45-58.

Arne Bröring et al., "New generation sensor web enablement," *Sensors*,
7] vol. 11, no. 3, pp. 2652-2699, 2011.

Arne Bröring, Christoph Stasch, and Johannes Echterhoff, *OGC sensor
8] observation service interface standard, Version 2.0, OGC 12-006*.: Open Geospatial Consortium, 2012.

Cory A Henson, "A comprehensive ontology for knowledge representation
9] in the internet of things," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, 2012.

A., Sheth, C. Henson, and S Sahoo, "Semantic sensor web," in *IEEE*

10] *Internet computing*, pp. 78-83.

Vlasios, Tsiatsis, "The SENSEI Real World Internet Architecture," in *Future* 11] *Internet Assembly*, 2010.

Danh Le-Phuoc and Manfred Hauswirth, "Linked open data in sensor data 12] mashups," in *2nd International Conference on Semantic Sensor Networks*, 2009.

Manu Sporny, Gregg Kellogg, and Marku Lanthaler. (2012, Dec.) JSON-LD 13] Syntax 1.0. [Online]. http://json-ld.org/spec/latest/json-ld-syntax/

Facebook GraphQL. (2015, July) graphql/graphql-js. [Online]. 14] https://github.com/graphql/graphql-js