Semantic Enablement in IoT Service Layers—Standard Progress and Challenges

We believe that applying semantic technologies to IoT service layer platforms can improve data accessibility, data discoverability, and the ability to extract knowledge about the data. Therefore, this article shows how semantic technologies can be leveraged by IoT service layer platforms.

The Internet of Things (IoT) is significantly growing with an aim to make a connected world by providing numerous opportunities for many industrial sectors and domains such as smart cities, smart factories, and smart homes. Currently, however, IoT applications in these domains are not interoperable with each other. The heterogeneous nature of these applications provides justification for defining a standard way of abstracting vertical data models.1,2 IoT data is usually collected from various sources such as sensory devices and/or crowd sensing. The data is often stored in IoT platforms as resources based on different data models. This collection of data can vary in quality and context. Accordingly, a semantic approach—for example, used in the Semantic Web3—can provide great agility toward resource representation, sharing information, and inferring new knowledge from data in the IoT on a global scale.4

Standardizing a set of common functions (such as registration and discovery) across IoT applications and devices would reduce the development cost of IoT devices. This IoT service layer enables application development independent of the underlying network communication and protocols (such as HyperText Transfer Protocol [HTTP] and Constrained Application Protocol [CoAP]) by abstracting different network technologies.5 As most IoT service layer platforms simply store IoT data in a non-semantic aware fashion, the meaning of the data cannot be conveyed to IoT applications. Therefore, they are unable to understand the context of the data. Meaningful use of any IoT data requires knowledge about its context such as its geolocation, its

Komal Gilani
Sejong University

Jaho Kim
Korea Electronics Technology Institute

JaeSeung Song
Sejong University

Dale Seed
InterDigital Communications

Chonggang Wang
InterDigital Communications
units and its producer. We believe that applying semantic technologies to IoT service layer platforms can improve data accessibility, data discoverability, and the ability to extract knowledge about the data. Therefore, this article shows how semantic technologies can be leveraged by IoT service layer platforms.

Since the way of storing and managing data in IoT platforms is different compared to the Web, this article shows mechanisms of using a data modelling language such as Resource Description Framework (RDF) to semantically describe IoT data, methods of associating IoT data with this semantic metadata, and methods of handling semantic queries to discover meaningful data from IoT platforms. For this purpose, we have designed a semantic-enabled IoT service-layer platform based on oneM2M global IoT standards supporting semantic features such as annotation and discovery.

**SEMANTIC TECHNOLOGIES AND RELATED STANDARDS**

This section explains why semantic technologies are required for IoT service layer platforms and gives an overview of core semantic technologies. Semantic technologies can play a critical role in data and knowledge management for context-awareness in IoT service platforms.

**Ontology**

Ontology represents concepts as objects that have properties and relationships with other objects. An ontology describes linguistic artifacts using a shared vocabulary of basic concepts about a piece of reality. It helps to support semantic exchange and context-driven communications among people and machines by defining shared and common theories.

**RDF and RDF Schema**

RDF is a standard model and language that represents the ontological level of facts about a resource or an individual—for example, types of individuals and their relations, respectively. RDF Schema provides a vocabulary for structuring RDF resources and describing relationships among resources. This includes the modelling of classes (rdfs:Class), the rdf:type property that provides the links of instances to a class, and the rdfs:subClassOf property, which allows the specification of class hierarchies.

**OWL**

As an ontology language, RDF and RDFS have limited expressiveness, as they have difficulties describing cardinality constraints (e.g., Parking Garage A has more than 10 unoccupied parking spots). Therefore, OWL was introduced to provide greater expressiveness and even support ontological reasoning. OWL offers different sublanguages with different levels of expressiveness and related properties regarding reasoning completeness and time complexity.

**SPARQL**

SPARQL is a query language for interacting with a triple store to process stored RDF triples. SPARQL can support ontological reasoning and semantic discovery. The triple store typically provides an interface to receive SPARQL query requests from a user and to send responses back to the user. Now the question is whether and how these technologies can be leveraged in an IoT service layer platform to support semantic interoperability.
SEMANTIC-ENABLED IOT SERVICE LAYER

A common IoT service layer platform is required by the IoT market to facilitate multi-industry IoT applications. The oneM2M Global Initiative is an international partnership project to develop a globally acceptable IoT service-layer standard. The common service layer specified by oneM2M can be embedded into various IoT entities such as end devices, gateways, and servers. It provides various IoT common service functionalities such as device registration, group management, and security and privacy.

The oneM2M service layer provides a means for connecting various IoT devices regardless of their access technologies, collecting data from these devices, and managing the collected data. Through its semantic capabilities, it also supports the annotation of semantic descriptions to oneM2M resources. Figure 1 shows the high-level design of a semantic-enabled IoT service layer platform. In order to support semantic features, IoT service-layer platforms have to support at least three basic features as follows:

- **Semantic annotation**: To achieve data interoperability, the service layer first should be able to support describing the meaning of resources/data. IoT service-layer resources (i.e., data sets) can be annotated with semantic information using standardized ontologies and data structures.
- **Semantic query and discovery**: The platform can support queries from IoT applications based on a semantic query language. When a semantic query is received, the platform executes the query by retrieving semantic information for the targeted resources and processing the discovery query.
- **Semantic mashup**: Like a traditional Web mashup, a semantic mashup is used to compose a virtual IoT resource from more than one IoT resource, which can be other existing virtual resources as well.

In order to provide semantic services to users properly, it is necessary to define common vocabularies, standardized data formats and description rules that can eventually solve the interoperability challenges caused by heterogeneous IoT data. The standard RDF language can be used to describe the semantic information. Also the annotated semantic metadata is then stored by the platform in a new resource designed to accommodate semantic information in an RDF/RDFS format. The metadata can also be stored in a triple store/ontology repository.

---

![Figure 1. Semantic capabilities in an IoT service layer.](image-url)
SEMANTICS IN ONEM2M STANDARDS

In this section, we describe how the semantic IoT features mentioned in the previous section can be realized in an IoT service-layer platform.

oneM2M Resources for Storing Semantic Information

The two basic logical entities that play a major role in the oneM2M system are an Application Entity (AE) and a Common Service Entity (CSE). In the oneM2M architecture, both CSEs and AEs can reside within different nodes, such as an Infrastructure Node (IN) for a server platform, a Middle Node (MN) for a gateway, an Application Service Node (ASN) and an Application Dedicated Nodes (ADN) for a constrained device. The AE is the logical entity that provides application’s business logic. It is used for hosting sensors, applications, and it resides in the Application dedicated node, which is called AND-AE. On the other hand, the IN-CSE entity is hosted on a server. The CSE functionality is provided for utilization by various AE resources. oneM2M adopted a resource based data model, in which all services are represented as resources. A resource can be uniquely addressed by a Uniform Resource Identifier (URI) and manipulated via create, retrieve, update, delete, and notify operations (CRUD+N).

To enable semantic technologies, the oneM2M service layer defines a <semanticDescriptor> resource, as highlighted in Figure 2. This resource is responsible for storing semantic information related to its parent resource and potentially sub-resources. It is created inside an existing container resource or AE resource of CSE in the oneM2M resource structure. The contents of this resource can be provided based on ontologies. The <semanticDescriptor> resource contains various attributes—that is, ontologyRef for the URI of an ontology, descriptorRepresentation to indicate the format of the semantic information, relatedSemantics to contain the URIs of other related descriptor resources, and descriptor for semantic information itself—to facilitate semantic information management. The <subscription> resource can be added as a child resource by any CSE/AE that expects to receive automatic notifications on the changes of a <semanticDescriptor> resource.

Figure 2. oneM2M resource structure

Let’s describe an example of the semantic information management process, where two sensors measure temperature information in different units and a smartphone application makes a discovery request of relevant semantic information. These two sensors are represented as ADN-AE-1 and ADN-AE-2 in an IoT platform server and periodically store measured temperature values in the server. The measured temperature sensor values are stored to a <contentInstance> resource for each sensor reading with a <semanticDescriptor> as its child resource. The <semanticDescriptor> resource is used to store the semantic information about the temperature sensor reading and the measured value. Once the <semanticDescriptor> resource is created, the smartphone application (i.e. ADN-AE-3) sends a semantic discovery request to the IN-CSE,
which contains a semantic filter. Then the IN-CSE will use the semantic filter to discover desired resources. After this the application ADN-AE-3 receives a response in the form of unique resource identifiers. Based on the returned list of unique resource identifiers, ADN-AE-3 can make another request to the IN-CSE to retrieve one or more semantic descriptor resources.

oneM2M Base Ontology

In general, information and operations in each IoT system can be described by ontologies, which provide a vocabulary with a structure. These ontologies (with OWL representations) can be used to support interoperability between different systems via ontology integration or mapping. For this purpose, oneM2M has defined its own ontology called the oneM2M Base Ontology. Various external ontologies from other IoT systems can be mapped to the oneM2M Base Ontology (e.g., by sub-classing and equivalence) so the interworking between the oneM2M system and external systems can be achieved. The oneM2M Base Ontology contains Classes (i.e. sets of individuals) and Properties (i.e. relationships and links between individuals), but no instances since the Base Ontology only supports a semantic description of these entities in the oneM2M architecture.

Semantic Annotation

Semantic annotation, which is the first step toward a semantic IoT system, is a process of adding semantic information to resources in oneM2M IoT platforms so that an annotated resource can be discovered semantically by heterogeneous IoT applications. In the oneM2M system, semantic information is represented using RDF/RDFS (or OWL) as RDF triples. Since the oneM2M system uses a hierarchical tree structure to store and manage its resources, semantic information is added as a special semantic resource. For this purpose, an IoT semantic annotator (IoT-SA) is introduced that runs within the oneM2M IoT system to automatically annotate semantic information for various resources representing sensors/devices registered to the oneM2M system with the following five steps:

1. As inputs to the IoT-SA, users/admins select IoT resource(s) to be annotated from the IoT platform and choose ontology(s) to be used during this annotation.
2. The IoT-SA then parses the given ontology to retrieve its classes and properties. The IoT-SA also retrieves other resources having related semantic information from the platform as candidate resources to establish relationships. The related semantic information is retrieved from `<relatedSemantic>` attribute, which contains URIs of other linked descriptor resource(s).
3. Users/Admins repeat a process to define semantic information in a triple format (i.e. `subject → predicate → object`) based on the given classes and attributes/properties from the given ontology.
4. Selected resources and semantic information are then converted into the defined RDF format and the IoT-SA uploads encoded RDF triples to the `<semanticDescriptor>` resource under the target resource.
5. The semantically annotated resources can now be discoverable by IoT applications. The updated semantic information can also be seen by users/admins for other purposes.
Semantic Resource Discovery and Semantic Query

One of the key benefits of semantic descriptions is to enable semantic resource discovery. Semantic resource discovery is basically a capability for an IoT application to discover resources based on certain specified characteristics of resources it is interested in. Semantic resource discovery can be achieved by using a SPARQL query. Figure 3 shows semantic discovery procedures in oneM2M. An IoT application is notified of the discovered resources and can retrieve desired resources based on the returned URIs after a semantic query is executed in the oneM2M platform. Specifically, a semantic filter is specified in oneM2M, which is formulated as a SPARQL query and contained in a semantic resource discovery request. An IoT application that wants to discover resources using semantics has to form a semantic query statement using SPARQL based on its needs.

When a SPARQL query is received targeting a specific resource (a.k.a. target resource), the receiver (i.e. IN-CSE in Figure 3) performs Semantic Graph Scoping (SGS) to decide the scope of the SPARQL query execution (i.e. to formulate a RDF basis for executing the SPARQL query). Semantic descriptors which are distributed and are hosted in the IoT platform’s resource structure are collected together to formulate a complete RDF data basis.

Semantic Mashup

Semantic Mashup is a process to discover and collect data from more than one IoT data sources and apply relevant business logic on the collected data to generate meaningful mashup results. For example, let us consider a case where users are interested in a service called “weather comfort index,” which provides and expresses satisfaction level regarding weather conditions. The comfort levels can be calculated based on the temperature and humidity sensors deployed in a specific location together with additional weather conditions; this is actually a mashup process and can be provided as a mashup service by an IoT platform. oneM2M specifies a semantic mashup service, which is implemented via a set of mashup procedures as shown in Figure 4. In order to utilize a mashup service, an IoT application should first discover the corresponding SMJP (i.e. a <semanticMashupJobProfile> resource as defined in oneM2M (Step 1). A SMJP describes the profile and necessary information required for a specific mashup service such as input parameters, member resources, mashup function, and output parameters. The SMJP resource shall contain <semanticMashupInstance>, <semanticDescriptor> and <subscription> as child resources. Based on the profile described in the SMJP, Originators (e.g. AEs) can create corresponding semantic mashup instances where semantic mashup results will be generated and stored in <semanticMashupResult>. The Mashup Requestor may use <semanticMashupResult> to retrieve the mashup result.
Figure 4. IoT Semantic mashup procedures in oneM2M. Each specific mashup service is described by a Semantic Mashup Job Profile (SMJP) which defines all required elements (e.g. types of input parameters, types of member resources, mashup operations or business logic, etc.) in RDF triples by this mashup service.

Based on the discovered SMJP, the next step is for the IoT application to create a Semantic Mashup Instance (SMI) resource, for example, by giving appropriate input parameters and member resources (Step 2). The SMI resource is used to contain input parameters, member resources, and any generated mashup results. Basically, SMJP provides a guidance on how an SMI shall be created and how the mashup result shall be calculated. The third step is for the IoT platform (i.e. CSE in oneM2M) to discover and collect original data from each member resource (e.g. via semantic resource discovery procedures) (Step 3). After the data is collected from the identified member resources (i.e. data sources), the IoT platform calculates the mashup result according to the business logic as described in the SMJP (Step 4). The generated semantic mashup result is stored in the SMI, which can be retrieved by the IoT application or other entities (Step 5).

CONCLUSION

There is a strong need to resolve the interoperability issue in the IoT service layer using semantic technologies inspired by semantic Web. This article described a semantic-enabled IoT service layer architecture based on the oneM2M global IoT service layer standards. In this architecture, semantic descriptor resources are introduced to represent semantic information in RDF triples. This semantic descriptor resource allows an IoT service layer or an IoT application to annotate existing IoT resources/data with additional semantic information using selected ontologies and RDF/RDFS. The added semantic information is then leveraged for semantic filtering/discovery and semantic mash-up.

The proposed semantic-enabled IoT architecture also supports a semantic repository to maintain all semantic information in a centralized triple store. Then, a SPARQL query can be executed directly on the triple store against the semantic information stored there. Future work includes advanced semantic annotation with other data models, information synchronization between oneM2M service layer resource structure and the triple store, distributed semantic analytics and other functions, as well as interoperability with other standards.

ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No. B0184-15-1003, No. B0184-15-1001)
ACKNOWLEDGMENT

REFERENCES


ABOUT THE AUTHORS

Komal Gilani is an MSc student at Sejong University. Her research interests include IoT semantics, smart cities, and industrial IoT. She has a BS in computer science from Arid Agriculture University Rawalpindi. Contact her at G2017SFE052@sju.ac.kr.

Jaeho Kim is a principal engineer at Korea Electronics Technology Institute. His research interests include IoT, smart cities, and digital twin technologies. He has a PhD in electrical and electronic engineering from Yonsei University. Contact him at jhkim@keti.re.kr.

JaeSeung Song is an associate professor at Sejong University. His research interests include IoT semantics, software testing, and industrial IoT. He has a PhD in computing from Imperial College London. Contact him at jssong@sejong.ac.kr.

Dale Seed is a principal engineer at InterDigital Communications. His research interests include IoT protocols and services. He has a MS in computer science and engineering from the Pennsylvania State University. Contact him at dale.seed@interdigital.com.

Chonggang Wang is a principal engineer at InterDigital Communications. His research interests include IoT, blockchain and distributed ledger technologies, and AI-powered future networking and computing. He has a PhD in computer science from Beijing University of Posts and Telecommunications. Contact him at chonggang.wang@interdigital.com.