Toward Semantic Interoperability in oneM2M Architecture

Prior standards, and also oneM2M, while focusing on achieving interoperability at the communication level, do not achieve interoperability at the semantic level. An expressive ontology for IoT called IoT-O is proposed, making best use of already defined ontologies in specific domains.

Mahdi Ben Alaya, Samir Medjiah, Thierry Monteil, and Khalil Drira

Abstract

The oneM2M standard is a global initiative led jointly by major standards organizations around the world in order to develop a unique architecture for M2M communications. Prior standards, and also oneM2M, while focusing on achieving interoperability at the communication level, do not achieve interoperability at the semantic level. An expressive ontology for IoT called IoT-O is proposed, making best use of already defined ontologies in specific domains such as sensor, observation, service, quantity kind, units, or time. IoT-O also defines some missing concepts relevant for IoT such as thing, node, actuator, and actuation. The extension of the oneM2M standard to support semantic data interoperability based on IoT-O is discussed. Finally, through comprehensive use cases, benefits of the extended standard are demonstrated, ranging from heterogeneous device interoperability to autonomic behavior achieved by automated reasoning.

COMMUNICATIONS STANDARDS

Introduction

In the past few years, machine-to-machine (M2M) communications have witnessed the emergence of various and different standardization initiatives. Indeed, several applications sectors are pushing standards that are often targeting mainly a specific application domain such as smart meters standards developed by IEC or IEEE (EN 13757, IEEE 1888-2011, etc.). Different standardization defining organizations (SDOs) have tackled this problem by focusing on the definition of a horizontal service platform that fits different verticals. This work was consolidated later on into a global initiative called oneM2M.

It is worth noting that all these initiatives have focused on the communication interoperability between system entities (servers, devices, applications, etc.). Indeed, these standards have defined a horizontal service layer that enables seamless communication between heterogeneous entities independently of the underlying network and vendor-specific device technologies. It is thus possible to reach and deliver a message to any entity in the system. However, no standard has tackled the "meaning" of the message content being exchanged. Although the SmartM2M [1] standard has introduced some recommendations for supporting semantics [3], a generic data model has not been specified. This has been left to the appreciation of the service provider, system developer, or the system user. Such standards have achieved interoperability at the communication level only. It is important to investigate their extension to semantic interoperability. This will lead to efficient systems, where autonomic management could be achieved.

Semantic data is brought through the definition of a common set of ontologies that describe all system entities but also the data items produced, exchanged, and consumed by these entities. Various information models have been defined for the Internet of Things (IoT), ranging from specialized models such as the Zigbee or KNX data models to more general models such as W3C SSN [3]. These solutions suffer from two main issues: they are either too specialized and focus on a specific application domain, or they lack some concepts mainly related to actuators.

In this article we discuss and propose an ontology model called IoT-O that handles both sensing and actuating concepts of M2M devices, as well as concepts related to services. We then discuss the extension of the oneM2M standard to support semantic data based on the proposed ontology. Finally, through comprehensive use cases, we show the use of IoT-O following the oneM2M standard.

The oneM2M Standard

The oneM2M global initiative [4] is an international partnership project established in June 2009 by the seven most important SDOs in the world and various alliances and industries. The main goal is to define a globally agreed M2M service platform by consolidating currently isolated M2M service layer standards activities. The oneM2M standard is organized into five technical working groups focusing on M2M requirements, system architecture, protocols, security, and management, abstraction and semantics.

As described in Fig. 1, the oneM2M system architecture is composed of the following four functional entities: the application dedicated node (ADN); the application service node (ASN); the middle node (MN); and the infrastructure node (IN). Each node contains a common services entity (CSE), an application entity (AE), or both. An AE provides application logic, such as remote power monitoring, for end-to-end M2M solutions. A CSE comprises a set of service functions called common services functions (CSFs) that can be used by applications and other CSEs. CSFs includes registration, security, application, service, data and device management, etc.

The oneM2M standard adopted a RESTful architecture, thus all services are represented as resources to provide the defined functions. In the rest of the article we will focus on oneM2M, but the proposed solution also works for SmartM2M since the two standards share a similar architecture.

Mahdi Ben Alaya, Samir Medjiah, and Thierry Monteil are with CNRS and Univ de Toulouse.

Khalil Drira is with CNRS.

 0163-6804/15/$25.00

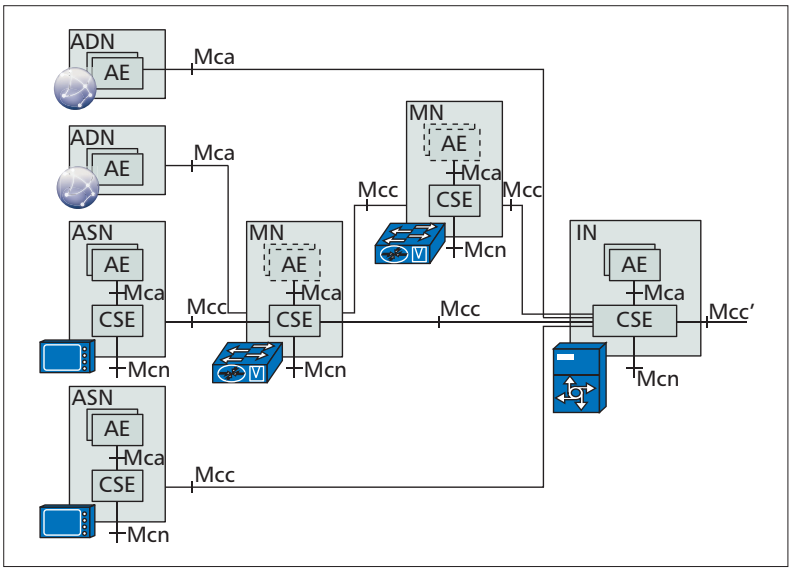

Figure 1. oneM2M system architecture [4].

Full Interoperability Challenge

Full interoperability is a desirable property to achieve in M2M systems, and will pave the way to the ultimate goal, i.e. autonomic management. Indeed, interoperability between heterogeneous devices and services is required to achieve self-configuration, self-healing, self-optimization, and self-repairing.

As introduced earlier, almost all standardization initiatives have not efficiently tackled the issue of full interoperability, i.e. considering both communication and data interoperability. Thanks to communication interoperability, M2M system entities already benefit from services such as discovery, monitoring, management, etc. Although such a service platform can be sufficient for the design and implementation of specific M2M systems, autonomic management using automated reasoning based on a knowledge oriented service platform cannot be achieved.

For example, using a service platform built upon oneM2M, an application can seamlessly discover new devices plugged into the system. This application can subscribe to the new device events, and will receive them as soon as they are triggered, even if the routing path implies crossing multiple entities and using heterogeneous communication protocols or network technologies at any segment of the communication path. This has been made possible thanks to the interoperability at the communication level. Now that device events have been successfully reported, the application does not have any means to understand the reports' content without prior conventions (data formats, encapsulation, and semantics) set up between the application and the device application developers.

IoT Ontology Principles

Ontologies have proven to be beneficial for intelligent information integration, information retrieval, and knowledge management. They enable the indexing of resources' content using semantic annotations that can result in the representation of explicit knowledge that cannot be assessed and managed because of their mess. Ontologies are very popular and useful to overcome challenges fixed in the proposed study because they provide an efficient way of cleverly structuring a domain, making use of semantic hierarchical and property/value relationships based on a vocabulary of concepts/instances [5].

In an M2M system, users and applications should be able to discover, monitor, and control sensors and actuators offering particular services and having particular properties with a high degree of automation. To reach this goal, an ontology for IoT shall represent a variety of concepts such as platform, deployment system, thing, device, node, service, sensor, actuator, sensing and actuating capabilities, observation, operation, time, unit, kind, and their relationships.

Since ontologies are designed to be reusable and extensible, it is possible to define a complete ontology for IoT by reusing existing ontologies. New concepts should be designed only when needed. This approach helps reduce the ambiguity of IoT terminology and makes it possible to converge quickly to a common vocabulary.

IOT-O: An Ontology for IOT

Since there is no single model that covers all IoT concepts, a set of well-defined ontologies were carefully selected to create an efficient ontology for IoT called IoT-O.

IoT-O Concepts and Relationships

IoT-O consists of five main parts: sensor, observation, actuator, actuation, and service models. Figure 2 shows how the selected ontologies are merged together to form this new ontology.

The DUL upper ontology is selected to describe very general concepts that are the same across all knowledge domains, and so facilitate reuse and interoperability. It is a lightweight foundational model for representing either physical or social contexts. The SSN ontology, which is aligned with DUL, is selected to represent sensors in terms of measurement capabilities and properties, observations, and other related concepts. However, it does not describe actuator devices.

Since currently there is no ontology that accurately describes actuators, we designed a new ontology called SAN, which is inspired by SSN and aligned with DUL, to describe actuators in terms of actuating capabilities and properties, actuation, and related concepts. The QUDV ontology was selected to represent quantities, units, dimensions, and values. The OWL-TIME ontology was selected to provide a vocabulary for expressing facts about topological relations among instants and intervals, together with information about duration, and about date time information.

Given that oneM2M aims to enable seamless interactions between business applications and services, it is important to represent how these services can be requested without any ambiguity in order to reduce the amount of manual effort required for discovering and using them. The MSM ontology was selected to describe services since it provides a common vocabulary based on existing web standards able to capture the core

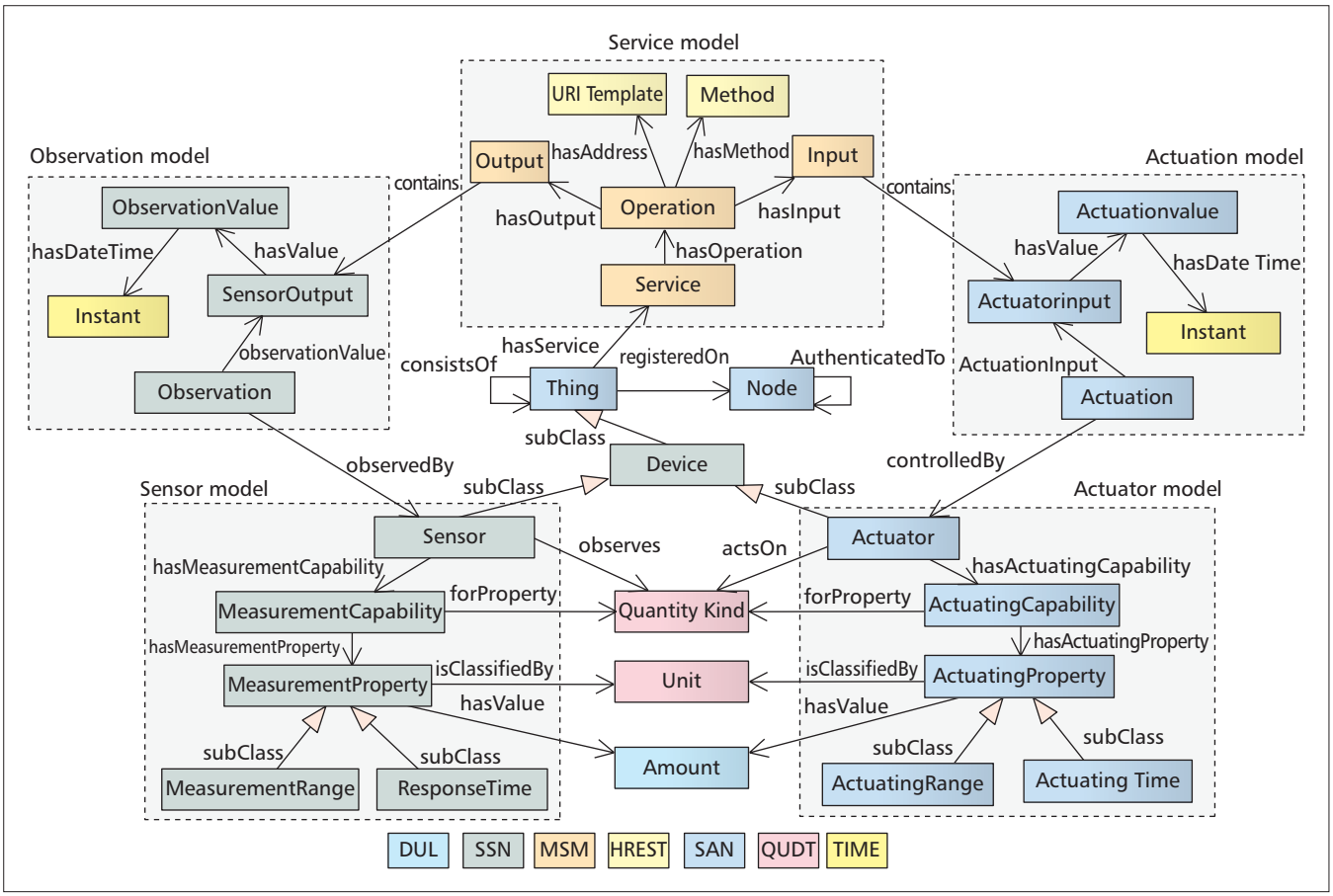

Figure 2. IoT-O ontology model.

semantics of both Web services and Web APIs in a common model. Each service is described using a number of operations that have address, method, input, and output message contents.

Actuator Model Instance According to IoT-O

Let's consider a concrete example representing a real actuator using the IoT-O ontology. The "HUELUX" actuator is a digitally dimmable wireless lighting bulb from Philips having power ranging from 0 Watt to 50 Watt. The luminosity level can be dimmed by requesting the required power value. The light bulb offers a web service to enable remote luminosity control. The luminosity can be dimmed instantaneously by sending a create request to the address "/HUELUX_APP/dimming" with a message body containing the required power. Figure 3 illustrates the corresponding ontology instance. It shows how the actuator, actuation, and service information are inserted into the IoT-O ontology. The actuator model represents the light bulb information and actuating capabilities, including power range. The actuation model represents the dimming command. The service model represents the light bulb web service, including the luminosity dimming operation, address, and method.

Semantics Extension to the oneM2M Standard

Through oneM2M working group 5, semantics is already envisioned for the oneM2M standard. However, as of its first release, semantics aspects are not yet tackled. In this subsection we will discuss a possible extension to oneM2M in order to support semantic interoperability. Two options are available.

The first option, which we called *reference-based integration*, uses the ontology reference attribute *ontologyRef* of *type xs:anyURI*, already defined in the oneM2M standard to include the semantic meta data. This attribute contains a unique reference of an ontology concept that describes the semantic meaning of a specific resource. It can be used to enhance the discovery mechanism to find a specific resource based on a semantic concept. This option is very limited as it informs only about a concept, and does not provide any semantic meta-data about the concept relationships, which is important for semantic interoperability.

The second option, which we called *inline integration*, consists of defining a new sub-resource called a "descriptor" to the oneM2M resource architecture. The "descriptor" sub-resource contains a complete semantic description of the resource defined using the resource description framework (RDF). This semantic description is exposed and shared across different applications, thus enabling semantic interoperability. This option requires the use of a triple store within a CSE to store the ontology instance.

We do not have to chose between the two options as they are complementary and can coexist in the same architecture. Such a solu-

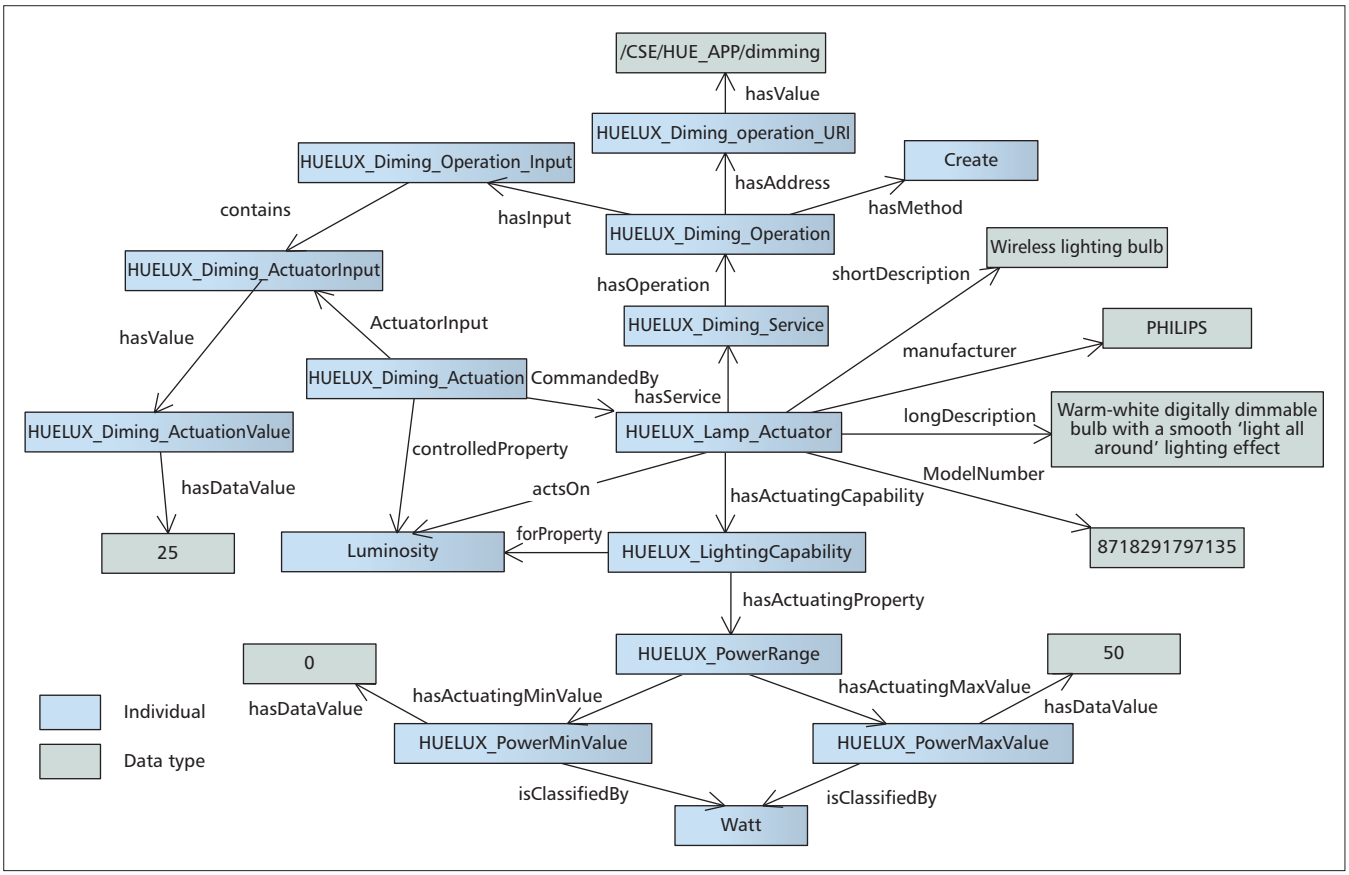

Figure 3. Actuator model instance according to IoT-O.

tion offers more flexibility to applications. In fact, depending on the scenario, an application may decide to read only the ontology reference attribute or go further and retrieve the full semantic description from the descriptor resource.

Semantic Data Interoperability

In this section, and through comprehensive use cases, we present the usability of the IoT-O ontology. The use case will also feature our living lab, i.e. the ADREAM smart building, and tackle the addressed challenge in a real scenario. Our OM2M [6] platform, which offers an open source implementation of the oneM2M standard, is used to validate the proposed approach. The OM2M high level architecture is described in Fig. 4.

LAAS Smart Building: ADREAM

ADREAM is the LAAS-CNRS smart experimental building. The main originality of this instrumented building compared to already existing buildings is that it is a "living lab" of 1700m^2, as it is both a research tool and a building with offices for the researchers.

The building includes 500m^2 of technical platforms (IoT, robotics, ambient intelligence, and energy) and 700m^2 of offices, with the remaining space devoted to a garden. It hosts our smart apartment equipped with various sensors and actuators connected using different networking technologies. The device set includes different sensors (temperature, humidity, luminescence, presence, etc.), as well as actuators such as electric plugs attached to different elements, e.g. lamps, fans, humidifier, etc., with all these devices gathered around different gateways. Each gateway is specialized in one or two networking technologies. Finally, these gateways are connected to one central server.

Seamless Device Discovery and Interaction

In order to demonstrate the interoperability achieved by the OM2M platform through its compliance with the oneM2M standards and its support of a generic data model IoT-O, we propose a simple scenario where the software platform is able to discover newly plugged devices such as sensors and actuators, browse the exposed attributes and methods, and finally interact with these devices by retrieving sensed data or triggering actions.

The scenario setup includes different devices attached to an M2M gateway through local network technologies that are either wireless, e.g. ZigBee and 6lowpan, or wired, e.g. KNX. The M2M gateway is connected to an M2M server. The M2M gateway entity is equipped with mapping modules that translate every communication with a specific networking technology into a generic communication protocol that is completely independent from the transport protocol or the network access. Thus, the support of new technologies or protocols is simply achieved through the implementation of the translation module, i.e. the interworking proxy unit (IPU).

When the IPU discovers a new device through the specific technology discovery mechanism, it will expose this device along with its attributes and methods to other entities in the M2M system. From the M2M system perspective, any data or action request is routed to this IPU in order to be translated to the specific technology operations. In this way, any application present in the M2M system can access the newly discovered resources (device, device's attributes, device's actions, etc.) using standardized restful operations. This can be achieved without any knowledge of the underlying network technology or its low-level mechanisms.

Furthermore, as all exchanged messages are augmented with semantics, an application not only has access to the data being generated by the device or the actions it exposes, but it can also understand the meaning of the data. Indeed, as the application has access to the ontology that defines the data, it can browse the ontology and map the received data elements into this ontology and perform the appropriate processing.

Toward Autonomic M2M Systems

In this section we demonstrate how the IoT-O ontology can be used to develop autonomic M2M systems [7, 8] capable of self-management to hide the intrinsic complexity from administrators and users. The main goal here is to use IoT-O ontology to dynamically reconfigure CSE resources to interconnect applications according to their semantic description.

Autonomic Service for Self-Configuring M2M Resources

In general, an application must manually perform a set of complex requests to discover relevant resources and perform several subscriptions to monitor the evolution of interesting resources. In addition, as an application has only a partial view of the M2M environment, it becomes difficult to find relevant services, especially in huge and highly distributed M2M environments.

An autonomic computing service is integrated into oneM2M as a new CSF is integrated into a CSE. A CSE control loop enables the dynamic reconfiguration of the oneM2M resource architecture when needed. The objective here is to help applications discover relevant devices and exchange data with the correct communication mode based on its description, role, and relationships.

To meet this goal a representative model of M2M system knowledge is required to assist the execution of the management process. Since the IoT-O ontology covers all required M2M concepts needed for this use case, it will be used as a knowledge model by the autonomic service.

ADREAM Smart Building Use Case

The ADREAM building contains two middle nodes, "MN-CSE-1" and "MN-CSE-2", registered to an infrastructure node "IN-CSE". A luminosity sensor is connected to the "MN-CSE-1", where it created the "LumSensorApp" application. A lamp actuator is connected locally to "MN-CSE-2", where it created the "LampActuatorApp" application.

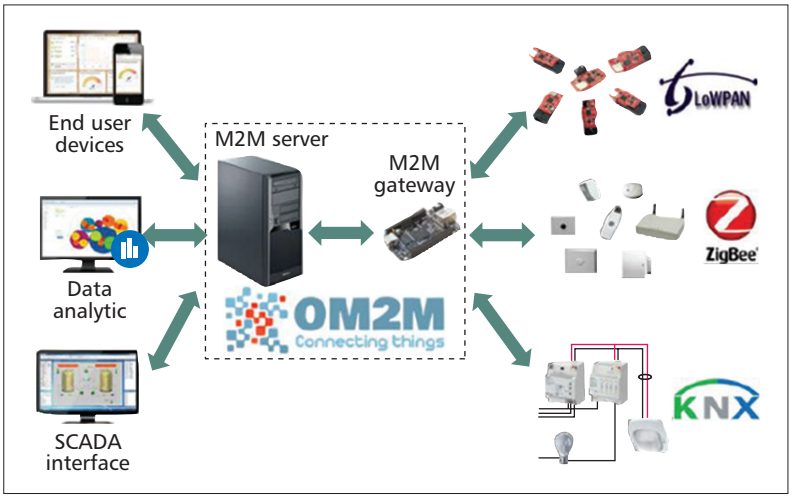

Figure 4. OM2M high level architecture.

The autonomic service discovers and monitors all registered applications, reasons on the IoT-O ontology model using inference rules to find relevant matching, and finally reconfigures the resource architecture accordingly to set up the required connections.

Monitoring the Smart Building System Entities

The autonomic service is responsible for detecting all existing nodes within the M2M system and reflecting the corresponding configuration into its knowledge base. It adds the infrastructure node as an individual of the "IN-CSE-1" node into the IoT-O instance. It then adds the two discovered middle nodes as individuals of the "MN-CSE" class. For each discovered node, the autonomic service retrieves the registered applications with their corresponding services and operations, and adds them to the ontology instance. Figure 5 shows the IoT-O instance as generated by the autonomic service.

The "LuminositySensor" individual is added as an instance of the "Sensor" class. It is registered to the "MN-CSE-1" node and is linked to the "Luminosity" quantity kind with the "observes" relationship. This sensor provides a service called "LuminosityService" that offers several operations to measure the luminosity level. In particular, the operation "RetrieveLuminosityOperation" informs about the address, method, and output required for retrieving data from to the sensor.

The "LampActuator" individual is added as an instance of the "Actuator" class. It is registered to the "MN-CSE-2" node and is linked to the "Luminosity" quantity Kind using the "actsOn" relationship. This actuator provides a service called "LampService" that offers several operations to configure the luminosity level. In this example the lamp is capable of adjusting its intensity according to ambient luminosity received as input. The operation "LampOperation" informs about the address, method, and input required to send to update the actuator state.

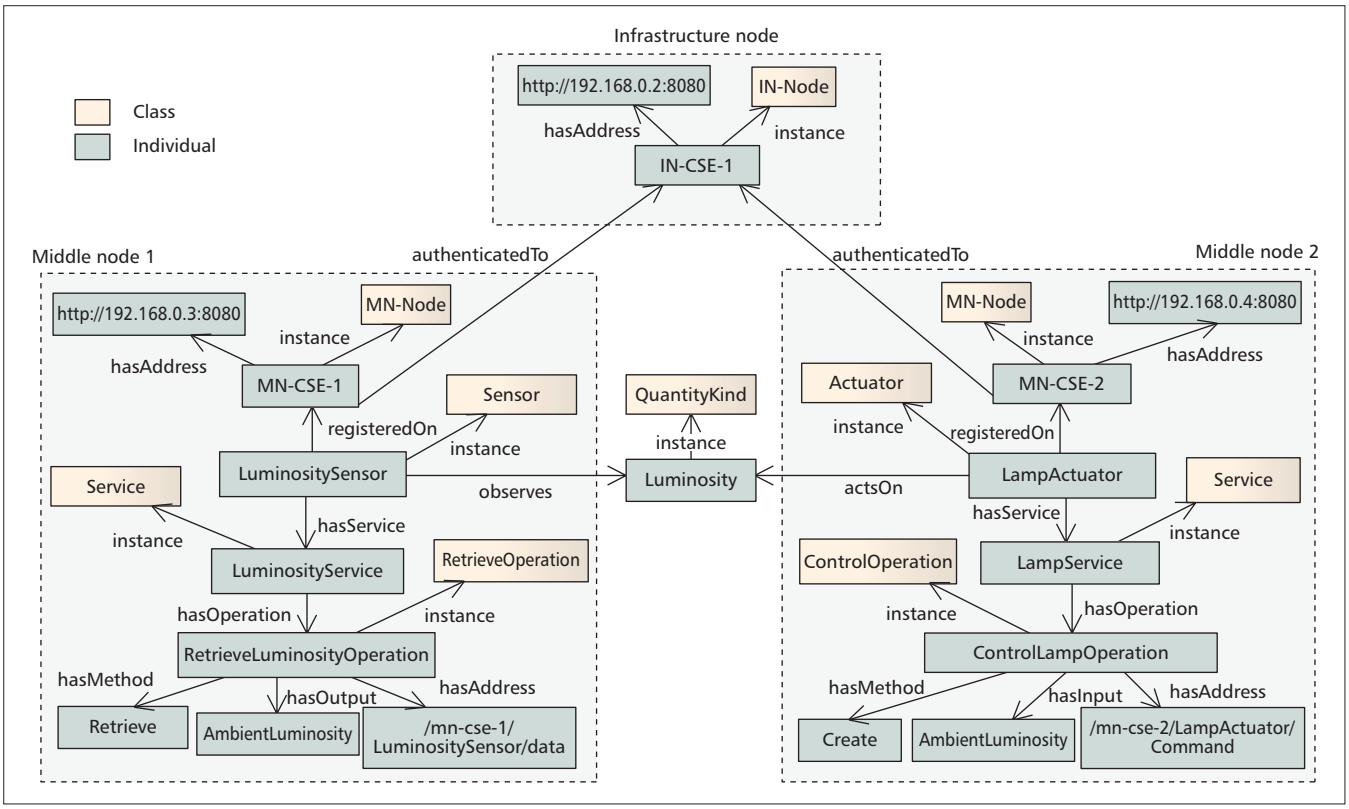

Figure 5. IoT-O smart building use case instance example.

Analyzing Applications Semantic Matching Using Inference Rules

Inference rules can be applied to infer new knowledge and so enrich the IoT-O instance with new individuals and relationships. This new knowledge is necessary to understand the role of each application in the M2M architecture. It allows each application to take maximum advantage of the services offered by other applications.

In this example the following SPARQL [9] rule is applied by the analyzer to find semantic matching between available things. It makes it possible to connect sensors with relevant actuators within the M2M system. For example, if there is a sensor that observes a particular quantity kind, and there also exists an actuator that acts on the same quantity kind, and if this sensor is not already matched to that actuator, then as a result a new "matches" relationship is inferred to link these two devices together. In our example, one semantic matching is detected: the luminosity sensor is matched with the lamp actuator:

```
CONSTRUCT {
        ?sensor iot-a:matches ?actuator
}
WHERE {
        ?actuator rdf:type san:Actuator.
        ?sensor rdf:type ssn:Sensor.
        ?quantityKind rdf:type qu:QuantityKind.
        ?actuator san:actsOn ?quantityKind.
        ?sensor ssn:observes ?quantityKind.
        FILTER EXISTS {
                ?sensor iot-a:matches ?actuator
        }
}
```

Using the same approach, more advanced rules can be applied including more constraints such as the device location, temporal requirements, and quality of services parameters.

Planning of Resource Reconfiguration Actions

The autonomic service plans the list of required actions to establish the communication between the matched things. Concretely, for each matching, a specific subscription action must be created with the correct parameters such as method, input, and also source and subscriber addresses. The autonomic service can extract these parameters from the IoT-O instance by processing the list of operations provided by each thing.

To configure the detected matching, the autonomic service extracts, from the operation "RetrieveLuminosityOperation", the method, input, and the source address where the subscription should be applied "/mn-cse-2/LuminositySensor/data". The operation "ControlLampOperation" makes it possible to extract the address of the subscriber (/mn-cse-1/LampActuator/Command). As a result, the autonomic service creates a subscription action to subscribe the lamp actuator to the luminosity sensor.

Establish the Communication between Matched Applications

Finally, the autonomic service converts each planned action to a Restful request, including all required parameters, using a specific communication protocol such as HTTP or CoAP. Then it sends each request to the IN-CSE. If a subscription request is executed successfully, the auto-

nomic service adds the "manages" relationship between the corresponding actuator and sensor; otherwise, an alert including the error details is reported. As soon as all requests are executed and reported, a new control loop can start.

The lamp actuator is dynamically subscribed to the luminosity sensor. It receives luminosity notifications and updates its level accordingly. Using the same approach, the current lamp actuator subscription can be cancelled and new subscriptions can be dynamically created to connect to more adapted sensors according to changes in the M2M environment.

Conclusion

Current M2M standards aim to provide a horizontal service platform to enable communication interoperability between machines. However, semantic data interoperability is not achieved, which brings into question the horizontality of such a platform. To overcome this challenge, a dedicated ontology for IoT, called IoT-O, has been defined. IoT-O merges together a set of popular ontologies and is enriched with new relevant concepts and relationships. Two possible integrations with the oneM2M standard are discussed. The main concepts and relationships of IoT-O are described using different use cases. An autonomic service making use of IoT-O and inference rules for resource architecture dynamic reconfiguration was also explained.

In future work, we propose to calculate the overhead cost of the proposed solution and the resulting overload. We also propose to validate IoT-O in various vertical M2M domains such as e-health, transport, and smart grid. The use of semantics makes it possible for end users to perform advanced discovery requests based on a SPARQL endpoint. However, the current oneM2M security solutions are not sufficient to support such a capability, the authorization mechanism must be rethought. Terminology for cloud data management and service lifecycle management can also be used to extend IoT-O concepts. A set of contributions will be sent to oneM2M for integrating IoT-O concepts into the standard to move forward toward semantic data interoperability.

References

[1] D. Boswarthick, O. Elloumi, and O. Hersent, *M2M Communications: A Systems Approach,* John Wiley & Sons, 2012.

[2] ETSI, "TR 101 584–Study on Semantic support for M2M Data," ETSI M2M, 2013, v2.1.1, pp. 1–34.

[3] M. Compton *et al.*, "The SSN Ontology of the W3C Semantic Sensor Network Incubator Group," *Web Semantics: Science, Services and Agents on the World Wide Web*, 2012, vol. 17, pp. 25–32.

[4] J. Swetina *et al.*, "Toward a Standardized Common m2m Service Layer Platform: Introduction to onem2m," *IEEE Wireless Commun.*, 2014, vol. 21, no 3, pp. 20–26.

[5] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, 2001, vol. 284, no 5, p. 28–37.

[6] M. Ben Alaya *et al.*, "OM2M: Extensible ETSI-Compliant M2M Service Platform with Self-Configuration Capability," *Procedia Computer Science*, 2014, vol. 32, pp. 1079–86.

[7] J. O. Kephart and D. M. Chess. "The Vision of Autonomic Computing," *Computer*, 2003, vol. 36, no 1, pp. 41–50.

[8] M. Ben Alaya and T. Monteil. "Frameself: An Ontology-Based Framework for the Self-Management of M2M Systems," *Concurr. Comput. Pract. Exp*, 2006, vol. 18. no 6, pp. 1412–26.

[9] J. Perez, M. Arenas, and C. Gutierrez. "Semantics and Complexity of SPARQL," *ACM Trans. Database Systems (TODS)*, 2009, vol. 34, no. 3, pp. 1–45.

Biographies

Mahdi Ben Alaya (ben.alaya@laas.fr) is a computer science engineer and IoT solutions architect. He obtained a Ph.D. in networks, telecommunications, systems, and architecture from the University of Toulouse, France. He is vice-chairman of the oneM2M Testing Group. He is co-founder and technical manager of the open source project OM2M at the Eclipse Foundation. His research interests include M2M interoperability, autonomic computing, IoT semantic, and information centric networking.

Samir Medjiah (medjiah@laas.fr) received a Ph.D. (2012) in computer science from the University of Bordeaux, France. He is an associate professor at Paul Sabatier University in Toulouse (France) and a research scientist at the Laboratory for Analysis and Architecture of Systems (LAAS-CNRS). His main research interests include application-network co-optimization, software-defined networking, network virtualization, M2M communications, and IoT applications. He is actively working on R&D projects related to M2M/IoT.

Thierry Monteil (monteil@laas.fr) has been an associate professor in computer science since 1998 at INSA Toulouse and a researcher at LAAS-CNRS. He works on parallel computing, cloud resource management, autonomic middleware, and machine-to-machine and Internet of Things architecture. He is member of ETSI (European Telecommunication Standards Institute). He also represents CNRS in the Eclipse Foundation and co-leads the OM2M open source project. He has authored more than 50 regular and invited papers in conferences and journals.

Khalil Drira (drira@laas.fr) obtained the Ph.D. and HDR degrees in computer science from UPS, University Paul Sabatier Toulouse, in October 1992 and January 2005, respectively. From October 1992 to September 2010 he was Chargé de Recherche, and since October 2010 he has been Directeur de Recherche, a full-time research position, at the French National Center for Scientific Research (CNRS). His research interests include formal design, implementation, testing and provisioning of distributed communicating systems and cooperative networked services.

The authorization mechanism must be rethought. Terminology for cloud data management and service lifecycle management can also be used to extend IoT-O concepts. A set of contributions will be sent to oneM2M for integrating IoT-O concepts into the standard to move forward toward semantic data interoperability..