Semantic Internet of Things

Amarnath Palavalli
VMware
Email: apalavalli@vmware.com

Durgaprasad Karri
VMware
Email: dkarri@vmware.com

Swarnalatha Pasupuleti
VMware
Email: spasupuleti@vmware.com

***Abstract*—Semantic IoT Framework is a flexible data management and processing platform for any type of data, built on Semantic Data Platform (SDP). SDP is based on declarative fact oriented approach for modeling, that enables the models to be executable themselves. The declarative nature of data semantics enables defining or modifying of data models dynamically with new concepts, relationships or constraints without having to change any other parts of the systems. The Event Manager, Data Processing and Analytics engine forms the core part of the proposed semantic IoT framework, which enables consumers to listen to metadata and data changes, derive analytics and fire domain specific business rules or policies. Therefore, Semantic IoT approach provides great agility in defining, modifying and interpreting metadata. A simple home automation usecase is presented to demonstrate how a device can publish its metadata as semantic facts including business rules or policies as constraints as well as publish the data to semantic data service running on the cloud infrastructure. The future work presented includes extending the semantic approach to collaboratively evolve IoT data model standard among device vendors.**

Keywords - IoT, Ontology, Object Role Modeling, Fact Oriented Modeling, Semantic Data Platform, Cloud Computing

I. Introduction

Internet of Things (IoT) is driving interconnection of uniquely identifiable embedded computing devices within the existing infrastructure. IoT enables collection of huge data from billions of interconnected devices. This data explosion requires highly agile data models, which can enable monitoring, processing, optimizing and analyzing data to gain insights, to make better decisions or any other actionable results.

The key drivers of IoT include new types of devices with their own data model being added every day. This has resulted in enormous growth of the structured data both in size and complexity. *IoT paradigm created a huge business opportunity and to realize the full potential, there is a need for a common way to abstract the heterogeneity of devices so that their functionality can be represented as virtual computing platform*[15][16][17]. Extending beyond heterogeneity abstraction, *Semantic IoT* framework is capable of accepting heterogeneous models that can co-exist in isolation as well be reconciled into a common model. This common model powers not only data sharing among devices and also forms the foundation for persistence, querying and building inferences on the shared data.

We propose *Data Centric* approach based on *Ontology*[10], where all device vendors participate to evolve the *Ontology*, which serves as common data model. That means new data or entities available in ontological model are available for consumption for other components and devices in the ecosystem by extending its visibility into the changed *Ontology*. 'The formal structure of Ontology makes it a natural way to encode domain knowledge'[12], therefore *Ontology* is an effective mechanism for representing and sharing of data.

A. Semantic Data Model based on Ontology

Object Role Modeling(ORM)[1] is a powerful approach for designing and querying models at the Conceptual level. *ORM* enables describing application data as *Semantic Data Model* containing Terms, Facts and Rules, that are easily understood by all. The modeling process is further simplified by using natural language and intuitive diagrams which can express information in terms of simple real life concepts and facts. Therefore, ORM is a great tool for representing *Universal Relationships*[11].

Building a data-centric software abstraction platform can take the data away from the applications. This is accomplished by creating common *Semantic Data Platform*(SDP) based on *ORM* and *Gellish*[2]. It is easy to define, modify and maintain the models using verbal expressions of *Gellish*[2].

B. Semantic IoT Framework

The proposed *Semantic IoT Framework* is shown in Fig. 1, based on the *IoT Reference Architecture* from *java.dzone.com* [6]. Our focus in this paper is on data semantics, responsible for definition, management and processing of data[6]. *Device Management* layer manages various types of devices connected to internet. Each device comes with its own meta data

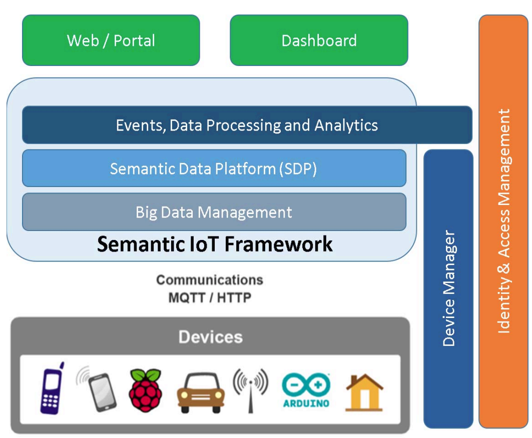

Fig. 1. IoT Reference Architecture [6]

978-1-5090-0662-5/16 $31.00 © 2016 IEEE
DOI 10.1109/ICSC.2016.35

including unique identification. *Identity and Access Management*(IAM) is another common layer responsible for security aspects. *Communication Layer* enables the connectivity of devices through well known protocols. The *Web/Portal* and *Dashboard* enables the users to communicate with devices from outside of the device-oriented system. *Semantic IoT* framework is a set of layers, responsible for aggregation, persistence, analytics and serving of data. *Semantic IoT* framework requires robust cloud infrastructure, hence an area with high significance for cloud infrastructure providers.

The rest of the paper is organized as, Section 2 provides literature survey, Section 3 presents the *Semantic IoT* framework based on *Semantic Data Platform* and provides simple usecase showing how simple it is to build with Semantic IoT Framework, Section 4 describes simple usecase leveraging Semantic IoT framework, Section 5 provides details about building with proposed framework, Section 6 explains the simple usecase of *Semantic IoT*, in Section 7 we briefly summarize the future scope of work and Section 8 concludes with a brief note on potential of *Semantic IoT* framework in making the IoT simple for various verticals.

II. Related Work

Due to complex nature of the relationships among IoT objects that change with time, it requires an efficient mechanism to manage and update the meta-data. In this direction, the work *IoT-DS*[13] proposes an IoT directory supporting semantic description, discovery and integration of IoT objects. The operations like adding/modifying/removing semantic Facts, capturing the entities and relationships for building and reconciliation of ontology are fundamental to *SDP*.

The management of devices in localized environment and their communication over internet to cloud platforms hosting backend infrastructure is not the focus of our discussion as it is addressed in other work like *Information Broker for Localized Computing Environments*[18] and *Semantic Middleware*[**?**][23]. The work[16] presents an approach to solve the discovery and interaction among heterogeneous IoT devices using transformation models for proper transformation of device syntax and semantics.

The Publish and Subscribe(pub/sub) dissemination paradigm has emerged as a popular means of disseminating time sensitive or filtered information in Service-Oriented Architectures(SOA). This work [14] proposes concrete approach to map client subscriptions to the semantic context of the published data, enabling the subscribers to receive notifications with enforcement of security policies across shared semantically related data.

Cloud computing environments dealing with large scale data with complex relationships comes with several challenges like scalability, data consistency, security and flexibility of mapping from different sources [21]. Building persistence capabilities with support for big data in *Semantic Data Platform* can serve as highly flexible big data platform driven by semantic data models reconciled from several sources. Hence, *Semantic Querying* for data analytics can be an attractive alternative to *NoSQL* and traditional *SQL* based databases[19].

The work[22] proposes semantic service matching using lament semantic analysis with a weighted-link analysis based on logical signature matching to provide functionality such as service recommendation, composition and provisioning in IoT. However, our approach thrives on the fact that each vendor and device type can potentially come with its own data model. Semantic data model reconciliation is the inherent feature of *SDP* and leverages the domain expertise of users to build ontology by reconciling the facts in a highly collaborative manner. Almost all inconsistencies and incompatibilities are discovered during the model building activity.

IoT development frameworks like *Kura*[8], *AllJoyn* [7] require significant implementation efforts to build support for new devices, as each type of device or any change in the capabilities of the devices, due to lack of agility in modifying the metadata dynamically. Also, these frameworks mostly focus on easing the development around device integration into the IoT framework, leaving the data persistence and analysis to cloud infrastructure and application providers.

III. Semantic Data Platform

Fact-Oriented Modeling[1] based on ORM, forms the foundation of *SDP*. In *Fact-Oriented Modeling*[1], a *Fact* is represented as an association between *Concept*s.

(HeadConcept) (role)/(co-role) (TailConcept)

This fact has the two reading directions left and right respectively.

(HeadConcept) (role) (TailConcept)

(TailConcept) (co-role) (HeadConcept)

In practice, *Concept* is abstract and independent of language it is expressed in. Hence, language specific noun form for a *Concept* is known as *Term*. As an example, consider,

Person has / of Name

defines a *Fact* expressing relation between *Concept*s *Person* and *Name*, expressed in natural language like English, where the left and right reading directions given respectively,

Person has Name

Name of Person

There is a need to express relations like synonym forms, taxonomy among the *Concept*s.

(role)/(co-role) is (relation type)

Example, *is a / subsumes is Taxonomy*

The relation plays an important role in reconciliation and mapping of *Concept*s across data models that are provided by various type of devices from different vendors [20]. Therefore, the reconciled concepts can be used interchangeably in all semantic constructs. *SDP* engine generates such semantic constructs as part of reconciliation process on adding new *Fact*s.

Semantic Path is an expression constructed consecutively using reading directions from *Fact*s or even with just one *Concept*.

Constraint is applied on reading direction of *Fact* or a *Semantic Path*, expressed as

(Concept) (role) (constraint-type) (Concepts)
(Concept) (constraint-type) (role) (Concepts)
Example, *Person has unique SSN*
Person has exactly one First Name and Last Name

Consider a simple device like temperature sensor, with data points like identifier (can be *IPv6Address*), description text, reading, etc. This information can be represented as *Facts* given here.

Facts:context=www.vmware.com/semantic-iot/device/tempsensor
TempSensor has/of Id
TempSensor manufactured by/manufactures Vendor
TempSensor has/of Description
TempSensor has/of Name
TempSensor has/of ReadingInterval
TempSensor generates/generated by TempReading
TempReading has/of Timestamp
TempReading has/of Temperature

Some of the *Constraints* can be like,
Constraints:context=www.vmware.com/semantic-iot/device/tempsensor
TempSensor is uniquely identified by Id
Id is of type IPv6Address
TempReading has exactly one Timestamp

Similarly an air-conditioning system data *Facts* are:
Facts:context=www.vmware.com/semantic-iot/device/ac
AirCondition has/of Id
AirCondition manufactured by/manufactures Vendor
AirCondition has/of Description
AirCondition has/of Name
AirCondition has/of PowerState
AirCondition has/of TempReading

The *Constraints* can be given as,
Constraints:context=www.vmware.com/semantic-iot/device/ac
AirCondition is uniquely identified by Id
PowerState value is defined by set$\{$*ON, OFF*$\}$

The *Context* is unique in which facts, relations and constraints are represented and collision is avoided in metamodel.

As seen from the model, *AirCondition* also has the unique *Id* and *TempReading* from the inbuilt *TempSensor*. So the vendors are free to extend or reuse the *Facts* published for devices. The advantage of this approach is, there is no binding at the implementation elements like APIs or libraries to integrate their devices with the *Semantic IoT* framework.

The data associated with the *Semantic IoT* model(Fig. 3) is continuously pushed to *SDP* by the devices through *Messaging Middleware* and persisted by *SDP*. Semantic model based persistence enables *SDP* to use semantic queries to fetch the data, when requested by *Applications* or *Analytics Engine*.

A simple semantic query to fetch readings of a specified temperature sensor can be given as,
"GET *TempReading has Temperature*
WHERE (*TempSensor has Id* = 'xxx-xxx...' AND *TempReading has Timestamp* $>=$ TODAY)"

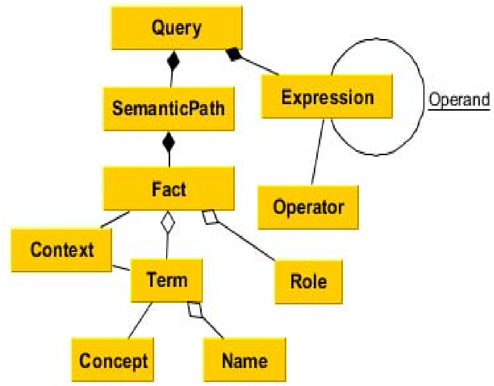

Fig. 2. Semantic Query Design

The *Semantic Query* implementation details are given in Fig.2 that follows the format given here.
Verb SemanticPath(s)/Instances Expression
where, *Verb* is in $\{$GET,PUT,DELETE$\}$, and *Expression* is a recursive construct to express the query criteria, given as
Expression = (Expression <AND|OR|NOT> Expression) |
(<SemanticPath> (< | > | = | >= | <=) <Literal>)

Expressing the *Semantic Data Model* in a form close to natural language enables device vendors to create, edit and even combine models written as plain text. A *Gellish* [2] parser can read semantic model in text form and emit executable runtime structure and vice versa. This forms the basis of *SDP*, an engine to interpret and drive data semantics for managing data on cloud.

IV. Semantic IoT Framework

Fig 3 shows the components of *Semantic IoT* framework. Web-portal interacts with *SDP*, which glues the various subsystems together, using semantic queries to pull the required data and presents to the user.

A. Event Manager

Event Manager(Fig. 3) manages notification of data change events to the devices. The devices and applications can subscribe to the data change events for the selected *Facts* from the semantic model[14]. It is the responsibility of device vendors to build part of the system that translate the notifications as commands to the devices to take appropriate actions. For example, the *AirCondition* system should be switched on if the temperature from the sensor crosses 25 degrees. This can be achieved as conditional value change as given below. The *Device Manager* can listen to data change notifications and issues power-on command to *AirCondition* system.
"ON (*TempSensor has TempReading has Temperature* $>$ 25 AND *TempSensor has Id = 'xxx.xxx...'*)
SET *AirCondition has PowerState* = 'ON'
WHERE *AirCondition has Id* = 'yyy.yyy...'"

Event Manager subscribes to data changes on all semantic paths that are participating in semantic *Constraint(s)*. Also, the

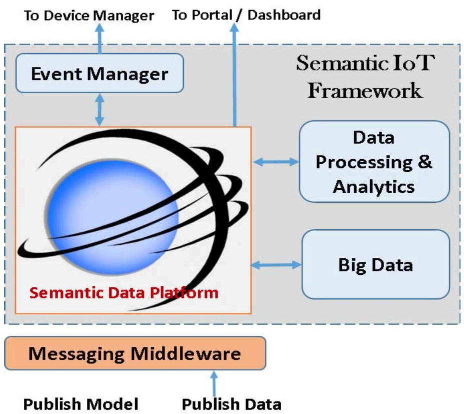

Fig. 3. Semantic IoT

devices subscribe to semantic paths representing their states through middleware infra. In our example, air conditioner device listens to its own state represented by semantic path *AirCondition has PowerState* and Event Manager listens to data changes for semantic path *TempSensor has TempReading has Temperature*. If a data change notification from *SDP* satisfies the specified constraint (in our case the temperature exceeding 25 degrees) then perform the action specified as part of the rule (in our case, setting air condition unit to 'ON' state for the specified device id in 'WHERE' clause). The device performs the appropriate action on receiving the notification of state change.

B. Data Processing and Analytics

In *SDP*, each entity in the model is reachable through a semantic path and the entity instance has corresponding path instance (semantic path with values embedded). Therefore, it is easy to define derived semantic path instances based on the constraints and associated actions. For example, if the temperature sensor measures higher than 25 degrees then consider the state of the room as 'WARM', represented in our approach as

"ON (*TempSensor has TempReading has Temperature* > 25 AND *TempSensor has Id = 'xxx.xxx...'*)
SET *Room has State* = 'WARM'
WHERE *Room has Address* = 'office3.floor1.room2...'"

This semantic data enrichment capability of *SDP* enables transformation and enrichment of raw data into information suitable for analytics and persistence. Therefore, the query engine of *SDP* can retrieve this data with same semantic queries. Extending further, smart semantic queries can result in rich analytics. Any change in the model requires only changing associated queries. It is easy to extend the semantic query framework to build out of the box functions to provide rich analytics, without having to rely on APIs or any other tight coupling. These analytics are also stored by *SDP* alongside the data from devices. There by, one can easily query and build custom analytics both on the raw and processed data.

For example, "ON *COUNT(Building includes Room has State = 'WARM')* $>= 50$
SET *FireStation has Alert* = 'ON' WHERE *Building has Address has ZipCode in (FireStation serves Area has ZipCode)*

V. Building with Semantic IoT Framework

Steps to build with Semantic IoT framework includes

- Define the semantic model of devices by modeling *Facts*, *Relations*, *Constraints* and *Rules*.
- Publish the model to *SDP*, ensure to resolve errors if any, as it is very crucial for the functioning.
- Enable the devices to push the data based on the model to *SDP* using semantic constructs.
- Provide the *Device Manager* implementation integration required for issuing commands to devices.
- The *Device Manager* will subscribe to the events relevant for the device.
- Applications can pull data and analytics from *SDP* using semantic queries.

VI. Simple Usecase

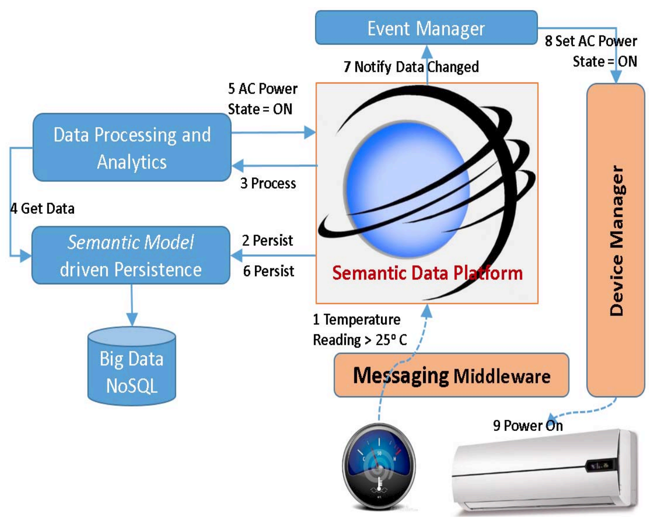

Fig. 4. Semantic IoT - Simple Usecase

Simple *Home Automation* usecase shown in the Fig. 4, where the devices like temperature sensor and air conditioner have published their model to *SDP*. The following steps demonstrate the event handling mechanism based on the rules or constraints defined.

- Step 1 - Sensor pushes reading at regular intervals along with its identification information like IPv6 address.
- Step 2 - *SDP* persists the readings from the temperature sensors into data store.
- Step 3 - *SDP* detects that a *Rule* has been defined for the Fact *TempReading has Temperature* from the semantic model of temperature sensor and forwards the data to *Data Processing and Analytics* engine for further processing.

- Steps 4,5 - *Data Processing and Analytics* engine figures out that the *Rule* must be triggered as the condition is satisfied (the temperature crossed 25 deg C in this case). The engine then pulls the target device (in this case *AirCondition* system) information associated with the sensor and set aircondition system *PowerState* to *ON* and notify SDP.
- Step 6 - SDP persists *AirCondtion* power state.
- Steps 7,8 - SDP also notifies about the data changes to *Event Manager*, which in turn notifies *Device Manager*.
- Step 9 - As the *DeviceManager* is listening for *AirCondition* system state issues *PowerOn* command to the *AirCondition* system.

VII. Future Work

The entire IoT stack spans hardware devices, sensors, gateways, cloud connectors, message protocols, middleware, bigdata management, analytics and many more. *SDP* can play an important role in evolving data model standards for the devices. The semantic model can be externalized using domain specific language like *Gellish*[2], that makes it very easy for vendors to adopt and extend the data model.

Adding new *Facts* to the semantic model is easy and non-invasive. The operations like change or delete *Facts* and *Constraints* is non-invasive. We propose to mark old *Facts* as deprecated and retain them in the same semantic model for backward compatibility. Later deprecated *Facts* need to be cleaned up once all device vendors move away.

VIII. Conclusion

IoT is an important area for cloud vendors, as the demand for large data storage and analysis leads to significant investments into cloud infrastructure. We believe*Semantic IoT* service can attract large IoT community for all data needs. Since, semantic model based on *SDP* can be externalized and shared among the users, it is very intuitive and easy to build application verticals like healthcare, logistics, public infrastructure, retail, industrial and many more [9]. Building a new vertical is effectively building a new semantic model through collaboration and creating queries for data and analytics.

Acknowledgment

The authors would like to thank *Vipin Balachandran*, *Jagannathan Srinivasan* and *Kalyan Venu Gopal Abbaraju* for providing valuable feedback. We would also like to thank *Balaji Katika* for his support in implementation of proof of concept.

References

[1] Terry Halpin, *Fact-Oriented Modeling*. LogicBlox, Australia and INTI International University, Malaysia, http://www.dama.org.au/wp-content/uploads/2012/04/DAMA-Sydney-3-April-2012-Terry-Halpin-Fact-Oriented-Modelling.pdf.

[2] Van Renssen, A.S.H.P, Dietz, J.L.G, *Gellish: a generic extensible ontological language - design and application of a universal data structure*, 3rd ed. Electrical Engineering, Mathematics and Computer Science, TU Delft, NL, http://repository.tudelft.nl/view/ir/uuid

[3] Peter Bollen, *Fact-oriented modeling in the data-, process-and event perspectives*, Department of Organization and Strategy, Faculty of Economics and Business Administration, University of Maastricht, Maastricht, The Netherlands, http://dl.acm.org/citation.cfm?id=1781008.

[4] Terry Halpin, *Fact-Oriented Modeling: Past, Present and Future*, Conceptual Modelling in Information Systems Engineering, http://link.springer.com/chapter/10.1007

[5] Wei Fan, Zhengyong Chen, Zhang Xiong, Hui Chen, *The Internet of Data: a new idea to extend the IOT in the digital world*, Frontiers of Computer Science, http://link.springer.com/article/10.1007/s11704-012-1036-z.

[6] *IoT Reference Architecture*, http://java.dzone.com/articles/internet-things-iot-reference.

[7] *AllJoyn*, https://allseenalliance.org/.

[8] *Kura*, https://eclipse.org/kura/.

[9] *IoT Vertical Markets*, http://www.im2mc.org/imcmarkets.

[10] N. Guarino, *Formal Ontology in Information Systems*, in FOIS'98, International Conference on Fonnal Ontology in Information Systems, lOS Press, 1998, http://dl.acm.org/citation.cfm?id=521669.

[11] Fabricio Henrique Rodrigues, Cecilia Dias Flores, Liane Nanci Rotta, *An Ontology Design Pattern to Represent Universal Relationships*. IEEE International Conference on Semantic Computing, 2015.

[12] Dejing Dou, Hao Wang, Haishan Liu *Semantic data mining: A survey of ontology-based approaches*. IEEE International Conference on Semantic Computing, 2015.

[13] Sejin ehun, Seungmin Seo, Byungkook Oh, Kyong-Ho Lee, *Semantic Description, Discovery and Integration for the Internet of Things*. IEEE International Conference on Semantic Computing, 2015.

[14] Norman Ahmed, Jason Bryant, Gregory Hasseler, Matthew Paulini, *Enabling Semantic Technologies in Publish and Subscribe Middleware*. IEEE International Conference on Semantic Computing, 2015.

[15] Jussi Kiljander, Alredo Delia, Francesco Morandi, Pasi Hyttinen, Janne Takalo-Mattila, Arto Ylisaukko-Oja, Juha-Pekka Soininen, Tullio Salmon Cinotti, *Semantic Interoperability Architecture for Pervasive Computing and Internet of Things*. IEEE International Conference on Semantic Computing, 2014.

[16] Guangyi Xiao, Jingzhi Guo, Li Da Xu, Zhiguo Gong, *User Interoperability With Heterogeneous IoT Devices Through Transformation*. IEEE International Conference on Semantic Computing, 2014.

[17] Janne Takalo-Mattila, Jussi Kiljander, Ferry Pramudianto, Enrico Ferrera, *Architecture for mixed criticality resource management in Internet of Things*. IEEE International Conference on Semantic Computing, 2014.

[18] Ivan V. Galov, Aleksandr A. Lomov, Dmitry G. Korzun, *Design of Semantic Information Broker for Localized Computing Environments in the Internet of Things*. 17TH Conference of Open Innovations Association (FRUCT), 2015.

[19] Ted Garcia, Taehyung Wang, *Analysis of Big Data Technologies and Methods*. 17TH Conference of Open Innovations Association (FRUCT), 2015.

[20] Ted Garcia, Taehyung Wang, *Scalability in Semantic Computing: Semantic Middleware*. IEEE International Conference on Semantic Computing, 2013.

[21] Yunchuan Sun1, Cheng Lu, Hongli Yan, Rongfang Bie, *Theory for Semantic Relation Computing and its Application in Semantic Link Network*. IEEE International Conference on Semantics, Knowledge and Grids, 2012.

[22] Jianying Zhang, Hongbo Liu, Hong Yu, *The motive for constructing a high level data model on cloud databases*. IEEE International Conference on Semantic Computing, 2012.

[23] Gilbert Cassar, Payam Barnaghi, Wei Wang, Klaus Moessner, *A Hybrid Semantic Matchmaker for IoT Services*. IEEE International Conference on Semantic Computing, 2012.

[24] Zhexuan Song, Alvaro A. Cardenas, Ryusuke Masuoka, *Semantic middleware for the Internet of Things*. IEEE Internet of Things (IOT), 2010.