Towards a Common Data Model for the Internet of Things and Its Application in Healthcare Domain

Rıza Cenk Erdur, Özgün Yılmaz, Onurhan Çelik, Anıl Sevici
Department of Computer Engineering
Ege University
İzmir, Turkey
e-mail: cenk.erdur@ege.edu.tr

Olgun Cengiz, Cem Pancar, Tuğçe Kalkavan, Gizem Çelebi, Hasan Basri Akırmak, İlker Eryılmaz, Arda Güreller
Ericsson Turkey
İzmir/İstanbul, Turkey
e-mail: ilker.eryilmaz@ericsson.com

***Abstract*— In the Internet of Things (IoT) environment, there exist a lot of devices, such as mobile phones, tablets and sensors, which are connected to each other. Huge amount of data is being generated from those connected devices. One of the challenges in developing IoT platforms and/or IoT applications is the representation and storage of this data. Towards this aim, in this paper, a standards based common data model has been proposed. The proposed data model basically depends on IoT-Architectural Reference Model (IoT-ARM), but is also extended with some concepts coming from the European Telecommunications Standards Institute (ETSI) Machine to Machine (M2M) functional specification. To show its applicability, we have instantiated the data model for the healthcare domain.**

Keywords-internet of things; data model; IoT-ARM; ETSI.

I. INTRODUCTION

In the IoT environment, there are a lot of interconnected devices, such as mobile phones, tablets and sensors. Those interconnected devices produce vast amount of data. Representation and storage of that data is one of the hot topics in the IoT area. Accordingly, the main objective of this paper is to define a common data model which is compliant with the current standards in the IoT research area.

There are two main standardization efforts in the IoT area. One of them is the efforts of the ETSI [1] as a result of which a M2M functional architecture specification has been published among many other specifications. The other one is IoT-A (IoT-Architecture), a European Union's Seventh Framework project consortium with several partners both from academia and industry. The objective of IoT-A is to create the architectural foundations of IoT environments as a result of which the IoT-ARM had been released [2]. The common data model presented in this paper basically depends on the IoT-ARM reference model, but is extended with some concepts taken from the ETSI specification.

To show the applicability of the proposed data model, we have instantiated the data model for the healthcare domain. We then mapped that instantiation to MongoDB [3], which is a document oriented NoSQL database.

The rest of the paper is organized as follows. In Section II the IoT-ARM reference model is overviewed to provide a background. Section III explains the IoT-ARM compatible data model which is extended using some concepts taken from the ETSI specification. Section IV discusses the implementation of the data model on MongoDB. Finally, Section V includes conclusion.

II. IOT-ARCHITECTURAL REFERENCE MODEL

The IoT-A aims to establish an architecture for the Internet of Things. An architectural reference model has been established to form a common ground for developing IoT applications [2]. The layers of this reference model are shown in Fig. 1.

Since the main focus of this paper is to define a common data model, we are interested in the domain and information models.

Part of the IoT domain model is given in Fig. 2. The concepts of service, resource, virtual entity, physical entity and device constitute the main part of the IoT domain model [2] and they are explained in the following paragraphs. Please refer to chapter 7 of [2] for the full UML (Unified Modeling Language) representation and explanation of the IoT domain model.

In the IoT domain model, the physical entity can be any physical object from humans to animals. For example, physical entities can be a kind of flower loaded into a truck for transportation and these flowers are subject to environmental monitoring. Physical entities are used in the digital world as virtual entities [2]. In our model, we have *Virtual Entity* class to include these physical entities.

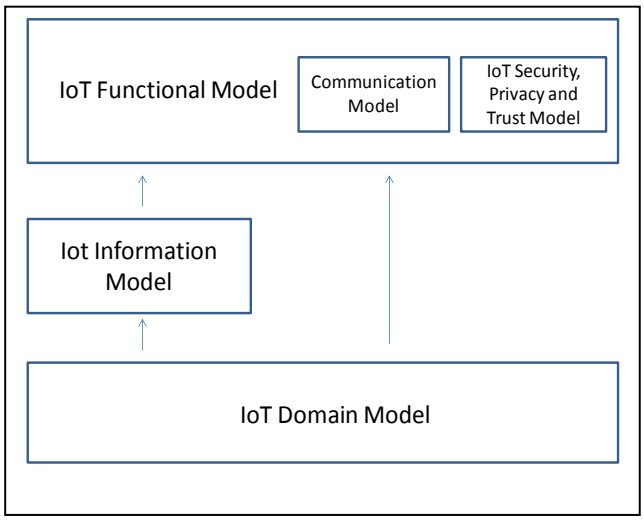

Figure 1. IoT –A Reference Model

 ISBN: 978-1-61208-507-4

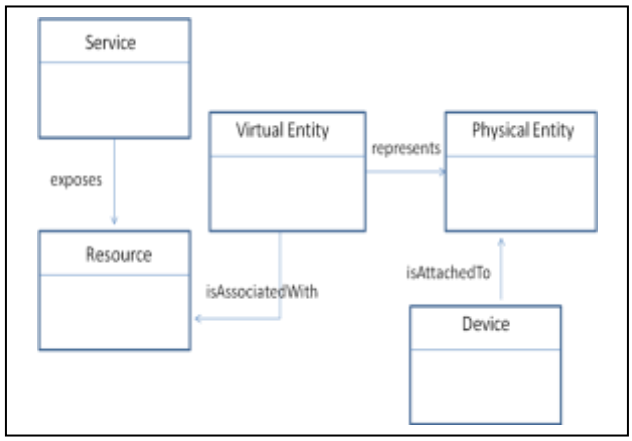

Figure 2. Part of the IoT-ARM domain model (adapted from chapter 7 of [2])

Also, there are devices that can be sensors or actuators in the IoT domain model. They have a relationship with physical entities, and hence with virtual entities [2]. In the example mentioned above, there are sensors that measure the temperature of the truck that carries the flowers (e.g., orchids). Using our data model, which is presented in the next section, the developers can define devices within the sensor list in the *Virtual Entity* class.

Resources are software components that provide data from physical entities. A virtual entity can be associated with resources. IoT domain model also defines service. A service exposes resources and provides a standardized interface to present all of the functionalities for interacting with the resources/devices related with physical entities. For each virtual entity there is an association with different services. These services may provide different functionalities, such as getting information or carrying out the execution of actuation tasks [2].

The information model specifies how the information related with virtual entities is represented. It consists of attributes, their values and meta-data. Please, also refer to chapter 7 of [2] for detailed explanations and UML based representations of IoT information model.

In the following section, the common data model is presented. The data model basically depends on the IoT domain model and is extended with some concepts coming from the ETSI functional specification.

III. THE DATA MODEL

The main objective in defining the data model is to form a common data model infrastructure for IoT applications in different domains. For this purpose, open standards in this area, which are IoT-A and ETSI based, have been inspected and a common data model has been defined.

The idea that underlies the process of modeling IoT data is to represent physical entities and the devices (i.e., sensors and actuators) in an abstract way. In the application layer, these abstract entities can be grouped together with the required properties to be used in the form of an upper data model.

The IOT-ARM and ETSI compatible data model that we propose is shown in Fig. 3. *Virtual Entity* (*VE*), *Resource* and *Service* tables that exist in the model represent the virtual provisions of the real entities and physical devices in the IoT-A standard.

Subscription, *Registration* and *Group* tables that are provided by the ETSI standard are to help the application layer. These tables are used by applications accessing the services offered by the device.

Subscription represents a request from the application (an actor performing a request) to be notified about the changes made on the parent resource. A subscription resource can also be used as a timer to trigger other actions. In this case, the subscriber is notified at the expiration and receives a timeout reason defined at the subscription resource creation [1].

In order to subscribe to a resource a subscription should be created in the subscriptions collection. The child resource will be notified about the change of the parent resource of the subscriptions resource where the subscription is added [1].

Group table is a representation of the resources that are same or mixed. Manipulation can be made to all members in the group [1].

ETSI M2M adopts a RESTful architecture style. Information is represented by resources that are structured to form a tree. A RESTful architecture provides the transfer of representations of uniquely addressable resources. ETSI M2M standardizes the resource structure that resides on a M2M Service Capability Layer (SCL) by providing each SCL with a resource structure where the information is kept [1].

Resource is a uniquely addressed entity in the RESTful architecture. A resource should be accessed via a Universal Resource Identifier (URI) [1].

Issuer is the actor performing a request. An issuer can be an application or a SCL [1].

As a result of a successful registration of an application with the local SCL, application resource is created. In this scheme, applications should only register to their local SCL [1].

The data model in Fig.3 also illustrates the use of it in the healthcare domain. In the following section, the implementation of this data model on MongoDB is discussed.

IV. IMPLEMENTATION

IoT technologies are suitable for using in e-health systems, because in e-health systems, there is a huge amount of time series data coming from many devices similar to IoT domain.

DataPoint table is the most important table when the data model is mapped to a document oriented database like MongoDB, since data generated by the resources affects this table, and operations on this table are very intensive.

 ISBN: 978-1-61208-507-4

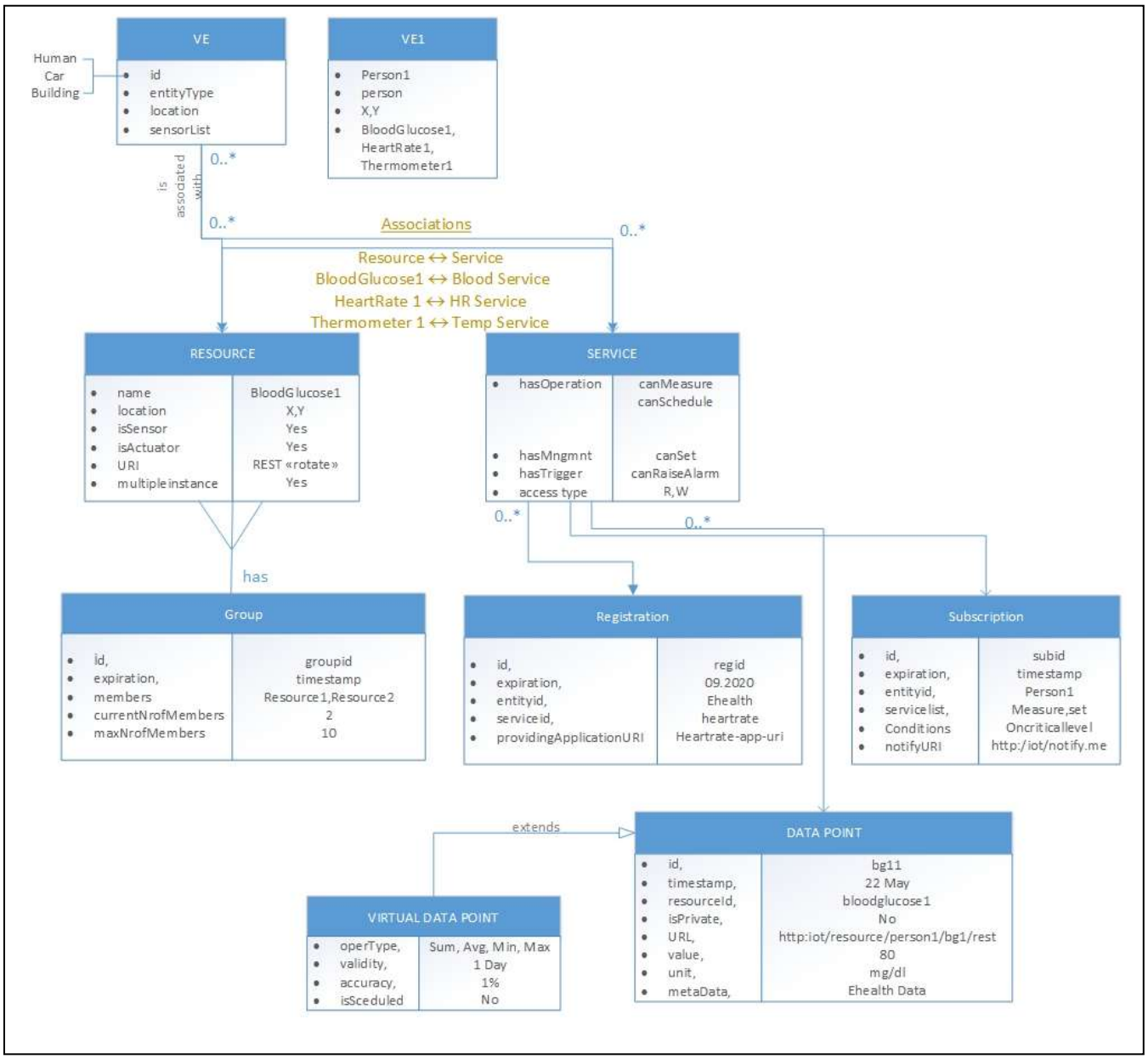

Figure 3. Generic data model which is IOT-ARM and ETSI compatible

Comparing the relational database and non-relational database, collections could be considered analogous to tables and documents analogous to records. The most of the computing is performed on *Data Point* collection in our data model. All data from devices/sensors will be added to this collection directly.

The most important point is how to store that data in an efficient way in terms of both access time and memory space. Hence, *Data Point* collection is designed to meet these constraints.

The data that come from sensors are in the form of time series data. Time series data is a great fit for MongoDB. If we store data in the form of time series, we can obtain high performance.

A row in a relational database management system (RDBMS) is a single data record. On the other hand, in MongoDB a document corresponds to a row. However, we can expand a document with more data records and store them in different formats. Thus, we can design each document to include data in a specified time interval.

Fig. 4 shows how a single data can be held in a single document. The sample data has been formed using the proposed data model.

 ISBN: 978-1-61208-507-4

```
dataPoint(collection)
{
        "_id" : ObjectId("5740a7ee35fe8d03d821bd52"),
        "resourceId" : "thermometer1",
        "timestamp" : ISODate("2016-02-10T23:00:00.000Z"),
        "isPrivate" : "Yes",
        "URL" : " http:iot/resource/human1/thermometer1/rest",
        "metaData" : "Health",
        "unit" : "celsius",
        "values" : "36.2"
}
```

Figure 4. One document storing one data value

The document given in Fig. 4 shows the storage of single data that comes from a resource called "thermometer1" at a certain time. If we use such a document, the document needs to be repeated for all the temperature values coming in a specific time period.

To be more efficient, the design in Fig. 4 needs to be changed. The new model involves all the data values for a specific time period in one document. This document has been shown in Fig. 5. In particular, it includes the data coming from "thermometer1" resource during an interval of one hour, sampled in periods of one second. As shown in Fig. 5, each hour's data has been represented as a different document.

In this way, instead of keeping each second's measured temperature data in a different document, all the data coming from that sensor during a specific period is stored in a single document.

This approach has the following advantages from different perspectives:

- Write performance advantages,
  - The first approach performs 3600 insert operations within the period of one hour. But the second approach performs 1 insert and 3599 update operations.

```
dataPoint(collection)
{
        "_id" : ObjectId("5740a7ee35fe8d03d821bd52"),
        "resourceId" : "thermometer1",
        "timestamp" : ISODate("2016-02-10T23:00:00.000Z"),
        "isPrivate" : "Yes",
        "URL" : " http:iot/resource/human1/thermometer1/rest",
        "metaData" : "Health",
        "unit" : "celsius",
        "values" : {
           0: { 0: 36.1, 1: 36.1, …, 59: 36.2 },
           1: { 0: 36.2, 1: 36.2, …, 59: 36.2 },
           …,
           58: { 0: 37.6, 1: 37.6, …, 59: 37.7 },
           59: { 0: 37.7, 1: 37.7, …, 59: 37.7 }
         }
}
```

Figure 5. Storing time series data

  - By avoiding unnecessary rewriting of the entire document and index entries, less disk I/O is performed. Because field-level updates are much more efficient.
- Read Performance advantages,
  - In the second approach, reads are also much faster. If we need the measurements during a specific hour, using the first approach we need to read 3600 documents. On the other hand, using the second approach we only need to read a single document. Reading fewer documents has the benefit of fewer disk seeks.
- Indexing advantages,
  - Another important advantage is from the indexing perspective. In the first approach, the great number of insert operations increases the size of the index. In the second approach, the size of the index will be reduced significantly due to performing a less number of insert operations.

Also, data can be kept in a document with different time periods. This period can be set depending on the device or sensor. For example, if a sensor/device produces data once per minute, then the document stores 60 data value on an hourly basis.

Sample data for *Data Point* collection is shown in Fig. 6. Two different resources are created with different identity numbers. These devices are blood glucose monitor and thermometer. Each of the devices has different data units and range of values. All of these data are generated randomly.

V. RELATED WORK

In this section, similar studies are reviewed. IPSO Smart Objects provide a common design pattern and an object model for communication between devices and applications. Resources and services in IPSO are used in the same way as the ETSI and IoT-A standards. But IPSO tends to model with RESTful protocol [4].

Resources, devices and services have their own identifier URLs (Uniform Resource Locator). For example, thermometer is a device and it is defined as an object URL. Every object has items that are services. Thermometer has a sensor value and sensor value is an item. Items have their own object URLs. After these definitions, the instance of the object is created and measurements are calculated with the object and the item values.

In our data model, thermometer is a sensor and it is stored in the resource table and it has a defined sensing temperature service in the service table. We also keep data in JSON format like IPSO object model, but there are differences in the definitions of the collections. We have registration and subscription tables originated from the ETSI specification. Resources will be registered to entities and the subscriptions are connected to these registrations. After registration, data streaming will be in the *Data Point* table having all resource data on the same collection.

Bui and Zorzi [5] propose the Internet of Things communication framework as the main enabler for

 ISBN: 978-1-61208-507-4

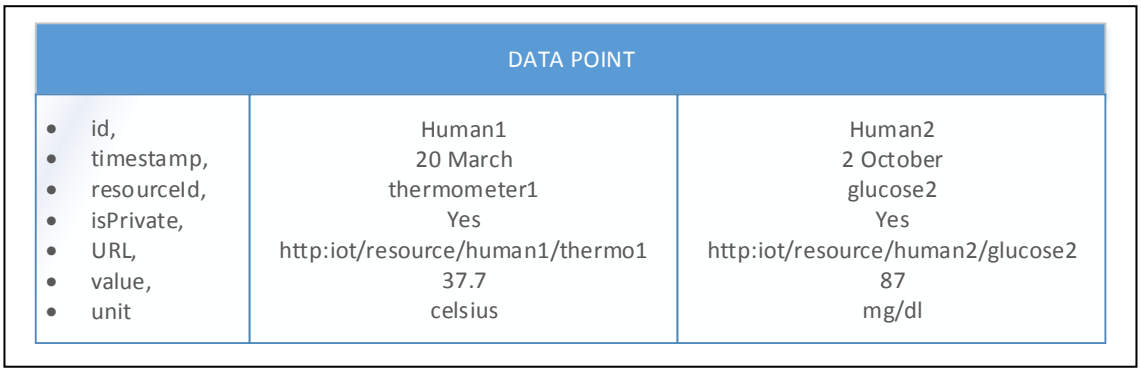

Figure 6. Example data for *Data Point* collection

distributed worldwide healthcare applications. The requirements of a unified communication framework are analyzed. Finally, the Internet of Things protocol stack and the advantages it brings to healthcare scenarios in terms of the identified requirements are presented. As for the implementation, this related work contributes to the IoT-A reference model.

De et al. [6] views the functionalities provided by objects of the IoT as 'real-world services' because they provide a near real-time state of the physical world. A semantic modeling approach for different components in an IoT framework is presented in this paper. This paper and the data model presented for modeling the IoT components contributes to the IoT-A reference model. Our data model is also based on the IoT-A reference model.

There are a number of works which use ontologies for representing sensor data. OntoSensor [6][7] defines an ontology-based specification model for sensors by borrowing parts from SensorML [8] descriptions and extending the IEEE Suggested Upper Merged Ontology (SUMO). However, a descriptive model for observation and measurement data is not provided. Reference [9] proposes an ontology-based model for service oriented sensor data and networks. However, representing and interpreting complex sensor data is not described [6]. The SensorData Ontology [10] is built based on Observations & Measurements and SensorML specifications which are defined by the OGC Sensor Web Enablement (SWE) [6][11].

W3C Semantic Sensor Network Incubator Group has constructed an ontology [12] to describe sensors and sensor networks. A high-level schema model is represented by this ontology to describe sensor devices and other related information about these devices. However, modeling aspects for features of interest and units of measurement and domain knowledge about the sensor data is not included. This domain knowledge is needed to be associated with the sensor data in order to support data communication and efficient reasoning [6].

Quwaider and Jararweh [13] discuss that there is a crucial need for efficient and scalable community health awareness in today's health care applications. In this related work, a cloud supported model for efficient community health awareness in the presence of a large scale data generation is presented. The goal is to monitor big data to be available to the end user or to the decision maker in a reliable manner. A system architecture and performance evaluation of the system is provided. But the data model of the system is not presented. So we cannot compare it directly to our system.

VI. CONCLUSION

In the IoT environment, platforms managing data and applications require more flexibility, agility and scalability to meet the requirements of IoT. A huge amount of sensor data which is schemaless, diverse in format and in scale needs to be processed. Thus, NoSQL databases should be the best candidates that can meet these requirements. Relational databases can still be used for structured data and where transaction information is important. For example, relational database systems still fit well for storing billing/charging or customer information. On the other hand, NoSQL databases are good when agility, scalability and heterogeneity are considered.

With all this information given in this paper, a document oriented NoSQL database can be preferred for storing sensor information in IoT environment. Document oriented databases provide flexibility, dynamic or changeable schema or even schemaless documents. New data structures can be added to the system easily.

In this paper, a standards based common data model for the IoT domain has been proposed. The proposed data model is compatible with IoT-A and ETSI M2M standards. Then as a proof of concept, this data model is instantiated for the healthcare domain and mapped to MongoDB. MongoDB is chosen as a document oriented NoSQL database because it provides secure, scalable, flexible solutions for IoT systems, and it is also compatible with cloud architecture.

As a future work, the developed database will be evaluated for its performance by distributing the data on several clusters on a cloud environment. The Yahoo Cloud Serving Benchmark [14] is going to be used in these tests.

REFERENCES

[1] ETSI, Machine-to-Machine communications (M2M); Functional architecture, ETSI TS 102 690 V2.1.1, 2013.

[2] A. Bassi et al., "Enabling Things to Talk", Springer, 2013.

[3] MongoDB, https://www.mongodb.com/ (retrieved: 08, 2016).

 ISBN: 978-1-61208-507-4

[4] J. Jimenez, M. Kostery, and H. Tschofenig, "IPSO Smart Objects", Position Paper for the IoT Semantic Interoperability Workshop 2016.

[5] N. Bui and M. Zorzi, "Health Care Applications: A Solution Based on The Internet of Things", Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies, pp. 131:1–131:5, 2011.

[6] S. De, P. Barnaghi, M. Bauer, and S. Meissner, "Service Modelling for the Internet of Things", Proceedings of the IEEE 2011 Federated Conference on Computer Science and Information Systems, pp. 949-955, 2011.

[7] D. J. Russomanno, C. Kothari, and O. Thomas, "Sensor ontologies: from shallow to deep models", Proceedings of the Thirty-Seventh Southeastern Symposium on System Theory, pp. 107-112, 2005.

[8] OGC, "OpenGIS Sensor Model Language (SensorML) Implementation Specification", Open Geospatial Consortium, Inc. 2007. http://www.opengeospatial.org/standards/sensorml (retrieved: 08, 2016).

[9] J. H. Kim, K. Kwon, K. D.-H, and S. J. Lee, "Building a service-oriented ontology for wireless sensor networks", Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science, pp. 649-654, 2008.

[10] P. M. Barnaghi, S. Meissner, M. Presser, and K. Moessner, "Sense and sens'ability: Semantic data modelling for sensor networks", Proceedings of the ICT Mobile Summit, 2009.

[11] OGC, "Open Geospatial Consortium (OGC) Sensor Web Enablement: Overview and High Level Architecture", OGC white paper, 2007.

[12] W3C-SSNIG, "W3C SSN Incubator Group Report". 2011. https://www.w3.org/2005/Incubator/ssn/XGR-ssn-20110628/ (retrieved: 08, 2016).

[13] M. Quwaider and Y. Jararweh, "A cloud supported model for efficient community health awareness", Pervasive and Mobile Computing, vol. 28, pp. 35–50, June 2016.

[14] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking Cloud Serving Systems with YCSB", Proceedings of the 1st ACM symposium on Cloud computing, pp. 143-154, 2010.

 ISBN: 978-1-61208-507-4