CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. (2014)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3203

SPECIAL ISSUE PAPER

Adding semantics to internet of things

Xiang Su [1,*,†], Jukka Riekki [1], Jukka K. Nurminen [2], Johanna Nieminen [3] and Markus Koskimies [1]

[1] *Department of Computer Science and Engineering, University of Oulu, Oulu, Finland*
[2] *Department of Computer Science and Engineering, Aalto University, Espoo, Finland*
[3] *Teliasonera, Sturenkatu 16, 00510, Helsinki, Finland*

SUMMARY

The development of Internet of Things (IoT) applications can be facilitated by encoding the meaning of the data in the messages sent by IoT nodes, but the constrained resources of these nodes challenge the common Semantic Web solutions for doing this. In this article, we examine enabling technologies for adding semantics to the IoT. Especially, we analyze data formats, which enable IoT applications consume semantic IoT data in a straightforward and general fashion, and evaluate resource usage of different alternatives with a sensor system. Our experiment illustrates encoding and decoding of different data formats and shows how big a difference a data format can make in energy consumption. Copyright © 2014 John Wiley & Sons, Ltd.

Received 9 March 2013; Revised 10 October 2013; Accepted 1 December 2013

KEY WORDS: Internet of Things; semantics; data formats; energy efficiency

1. INTRODUCTION

Internet of Things (IoT) is expected to bring the Internet truly into our everyday lives by connecting a vast amount of devices and objects to the Internet. All these devices and objects—from white goods, bicycles, and sport watches to environmental sensors, traffic lights, and tools used in factories—will have their unique identifiers (IDs) and will communicate with other peers and servers on the Internet. The resulting uniform access to these devices and objects will introduce significant possibilities for applications that help people to achieve their goals, companies to improve their processes—generally, the society to improve its citizens' quality of life.

While the main challenges of connecting IoT nodes in low level are being solved, integrating and interoperating huge amounts of information provided by IoT nodes are becoming increasingly important. Even more can be achieved if we add semantics to the information produced by the IoT nodes. As pointed out by Berners-Lee *et al*. in their landmark article about the Semantic Web, 'developments will usher in significant new functionality as machines become much better able to process and understand the data' [1]. We see this significant new functionality possible when IoT nodes send data directly in a format that contains semantics in addition to the raw data. Because the meaning of the data is encoded in the message, the receiver of the message can utilize the data in a straightforward and general fashion. The receiver does not need node-specific knowledge but can process data from all nodes in a similar way. However, because IoT nodes are often small devices

*Correspondence to: Xiang Su, Department of Computer Science and Engineering, P.O. Box 4500, University of Oulu, Oulu, Finland.
†E-mail: Xiang.su@ee.oulu.fi

with modest computing, communication, memory, and energy resources, they introduce challenges not present in the common scenarios of Semantic Web.

In this article, we tackle the challenge of adding semantics to IoT without breaking the constraints on resource usage. Common semantic technologies offer expressive representations. However, these representations require a considerable amount of resources, which are not available in IoT systems, and this conflict introduces a considerable challenge. We present the enabling technologies for adding semantics to IoT, compare the different approaches, and measure their resource usage, especially energy consumption with a sensor system. The main contributions of this article are (1) comparing the semantic expressivity of different data formats adding semantics to IoT, (2) measuring the resources needed to encode and decode them in a sensor system, and (3) suggesting data formats offering the best compromise between the usually conflicting characteristics of good expressivity and modest resource requirements.

We focus on different data formats enabling semantics. Hence, we assume that each encoded message is delivered under similar conditions (i.e., with the same protocol and along the same path) and do not discuss the effect of protocols or architectures on resource usage. Under these assumptions, a simple architecture suffices for the experiments. Moreover, we utilize one well-known public access ontology to determine the meanings encoded in the different data format. Other ontologies may give slightly different meanings to data items based on their own knowledge structures. We concentrate on studying generally how to map data values to ontologies, without referring to any specific ontology.

We published a brief comparison of data formats and preliminary evaluation in [2] and reported detailed introduction of different formats and approaches and careful analysis with new experiments here. The rest of this article is organized as follows: Section 2 presents data format alternatives, and Section 3 discusses energy efficiency issues. We present our system and evaluation results in Section 4, discuss the future work in Section 5, and conclude the paper in Section 6.

2. DATA FORMATS

One of the main challenges of IoT data formats is mapping between data formats and models used for constrained devices and data formats and models used in the Web and Semantic Web, such as eXtensible Markup Language (XML), Javascript Object Notation (JSON) [3], and Resource Description Framework (RDF) [4].

A data format should set minimal requirements for both IoT nodes and the data consumers, which are IoT applications and services utilizing data and supporting functionalities. ‘Minimal requirements for IoT nodes’ means that the solution should increase resource consumption as little as possible. ‘Minimal requirements for consumers’, in turn, means that the solution should be general, and any consumer should be able to interpret the data with minimal effort and *a priori* knowledge. Moreover, the data format should be compatible with Semantic Web, as only then, can the existing Semantic Web tools be used for inference, knowledge bases, ontology alignment, and semantic queries. A data format fulfilling these requirements allows application developers to easily utilize nodes implemented and deployed by others. Such a lightweight and easy-to-use data format could even bridge the current gap between different IoT domains and applications.

Research on Semantic Web has produced well-established specifications for formal knowledge representations. These knowledge representations allow logical reasoning that is able to infer new information from existing assertions and rules. Standard representations are potential candidates for representing sensor data. Among them, RDF is the most widely used data model for representing semantic data. RDF represents data as triples in the form (subject, property, and object). A triple denotes that a subject has a property whose value is the object. IoT data usually originate from devices, humans, and other entities in the physical world. It refers to attributes of phenomena and to relations among these entities. The simplest way of semantically representing IoT data, like a measurement made by an IoT device, is denoting the IoT device as the subject, the measured quantity as the property, and the measured value as the object. For example, ‘sensor 1’ is the subject, ‘temperature’ is the property, and ‘25’ is the value. The unit of measurement, for example, ‘Celsius’, can

DOI: 10.1002/cpe

be defined separately. Similarly, when Alice is in campus, ‘Alice’ is the subject, ‘isLocated’ is the property, and ‘Campus’ is the value.

We utilize a running example of two kinds of sensor nodes. One sensor node sends a time stamp value together with temperature, acceleration, and magnetic field values. A second one sends location data (longitude and latitude) with a user ID. Other sensor data can be represented in a similar way. Table I presents in RDF/XML format the first sensor data example of temperature, acceleration, and magnetization. This is produced by a sensor in our sensor system. Table II presents in RDF/XML format the second sensor data about location. A clear advantage of RDF is that the existing higher level languages RDF Schema [5] and Web Ontology Language [6] provide a standard vocabulary for defining classes and relationships among classes, which enables high-level inference. Hence, when IoT nodes express data in RDF, these languages facilitate realizing advanced semantic processing.

Notation3 (N3) [7], Turtle [8], and N-Triples [9] are alternatives for RDF. They are also based on triplet structure, but they differ in expressivity. They all can be transformed into RDF in a straightforward manner and are in most cases more lightweight than RDF/XML. Among these alternatives, N3 is a flexible language with strong expressive capability going beyond the RDF model, Turtle is an RDF-compatible subset of N3 while N-Triples has constrained expressivity. N3 and Turtle have short-hand syntaxes. These syntaxes shorten the descriptions but, on the other hand, require more computing resources when the descriptions are processed. Table III presents the temperature, acceleration, and magnetic sensor data in N3 format. Table IV presents location sensor data in N3 format.

RDF, N3, Turtle, and N-Triples are designed to be used by Web applications; hence, resource usage was not the main issue when these languages were designed. SenML [10], on the other hand, is a sensor data description language for representing simple sensor measurements and device parameters. It is targeted for resource-constrained devices, and hence, the amount of processing and the size of data were considered when it was designed. A SenML description carries a single base object

Table I. Temperature, acceleration, and magnetic sensor data in Resource Description Framework.

Resource Description Framework
<rdf:RDF xml:base=“http://iot.fi/o” xmlns:i=“http://iot.fi/o#”
xmlns:rdf=“http://www.w3.org/1999/02/22-rdf-syntax-ns#”>
<i:Sensor rdf:ID=“accmagSensor01”>
<i:timeStamp>2012-05-18T12:00:00</i:timeStamp>
<i:accX>618</i:accX> <i:accY>319</i:accY>
<i:accZ>671</i:accZ> <i:magX>123</i:magX>
<i:magY>234</i:magY> <i:magZ>345</i:magZ>
<i:temp>22.5</i:temp>
</i:Sensor> </rdf:RDF>

Table II. Location sensor data in Resource Description Framework.

Resource Description Framework
<rdf:RDF xml:base=“http://iot.fi/o” xmlns:i=“http://iot.fi/o#”
xmlns:rdf=“http://www.w3.org/1999/02/22-rdf-syntax-ns#”>
<i:LocationSensor rdf:ID=“locaSensor767”>
<i:ownerID>“Alice”</i:ownerID>
<i:longitude>25.468</i:longitude>
<i:latitude>65.058</i:latitude>
</i:LocationSensor>
</rdf:RDF>

DOI: 10.1002/cpe

Table III. Temperature, acceleration, and magnetic sensor data in N3.

N3
@prefix i:<http://iot.fi/o#>. @prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>. i:accmagSensor01 i:accX “618”; i:accY “319”; i:accZ “671”; i:magX “123”; i:magY “234”; i:magZ “345”; i:temp “22.5”; i:timeStamp “2012-05-18T12:00:00”; a i:Sensor.

Table IV. Location sensor data in N3.

N3
@prefix i: <http://iot.fi/o#>. @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>. i:locaSensor767 i:ownerID “Alice”; i:longitude “25.468”; i:latitude “65.058”; a i:LocationSensor.

consisting of attributes and an array of entries. Each entry, in turn, consists of attributes such as a unique ID for the sensor, the time the measurement was made, and the current value. SenML can be represented in JSON, XML, and Efficient XML Interchange (EXI) [11]. The SenML format can be extended with further custom attributes. For example, the resource type (rt) attribute can be used to define the meaning of a resource. Other semantic attributes can be defined in a similar way. Finally, additional information can be made available by including in a SenML description a link in the CoRE Link Format [12], but then additional communication is required to fetch that information. Table V presents the temperature, acceleration, and magnetic sensor data in SenML, using JSON, XML, and EXI, respectively (‘bt’ is base time and ‘bn’ is base name, in this case, it denotes a device ID). Table VI presents the location sensor data in SenML. ‘pr’ stands for prefix in Tables V and VI, which can be transformed to xml:base=“http://iot.fi/o” when SenML data are transformed to RDF/XML.

Another emerging attempt is to enable RDF level semantics for JSON format. Several proposals, including RDF/JSON [13], JSN3 [14], JTriples [15], RDFj [16], and JSON for Linked Data (JSON-LD) [17], have been presented. They allow an RDF graph to be written in a format compatible with JSON. To achieve this, the essential idea is to introduce universal IDs for JSON objects via the use of uniform resource identifiers (URIs), a mechanism to serialize a set of RDF triples as a series of nested data structure in JSON, and a mechanism to associate data types with values. RDF working group compared these formats with examples in [18]. Among alternatives, JSON-LD seems to be a most promising one and became World Wide Web Consortium candidate recommendation. JSON-LD is designed to be completely compatible with JSON, and it expresses semantics slightly over RDF model. This means in practice it can be considered to be a JSON serialization for RDF. JSON-LD requires minimal effort from developers to transform normal JSON to JSON-LD, in its easiest way, only two key words (@context and @id) need to be known for utilizing basic feature JSON-LD supports. The upper part of Table VII presents a compact JSON-LD packet describing temperature, acceleration, and magnetic sensor data, while the lower part of Table VII presents a corresponding context referred JSON-LD packet. Contexts describe short-hand terms for JSON-LD, and it can be directly embedded into data packets or be referenced. IoT nodes can agree contexts at design time or send full contexts on request between nodes for decreasing communication overhead in this way. For example, a node receiving context-referenced JSON-LD data (shown in the lower part of Table VII) can retrieve context information at http://iot.fi/json-ld/contexts. Table VIII shows location sensor data in compacted and context-referenced JSON-LD format.

DOI: 10.1002/cpe

Table V. Temperature, acceleration, and magnetic sensor data in different formats of SenML.

Sensor data in Javascript Object Notation


```
{“e”: [ {“n”: “accX”, “v”: 618},
{ “n”: “accY”, “v”: 319},
{ “n”: “accZ”, “v”: 671},
{ “n”: “magX”, “v”: 123},
{ “n”: “magY”, “v”: 234},
{ “n”: “magZ”, “v”: 345},
{ “n”: “temp”, “v”: 22.5}],
“bn”: “accmagSensor01”,
“pr”: “http://iot.fi/o#”,
“bt”: “3296120023”,
“rt”: “Sensor” }
```


Sensor data in eXtensible Markup Language


```
<senml xmlns=“urn:ietf:params:xml:ns:senml”
pr=“http://iot.fi/o#”
bn=“accmagSensor01”
bt=“3296120023”
rt=“Sensor”>
<e n=“accX” v=“618” />
<e n=“accY” v=“319” />
<e n=“accZ” v=“671” />
<e n=“magX” v=“123” />
<e n=“magY” v=“234” />
<e n=“magZ” v=“345” />
<e n=“temp” v=“22.5” />
</senml>
```


Sensor data in Efficient XML Interchange

0x80419cd95b ... 145 bytes ... 0801001000

bn, base name; bt, base time; pr, prefix; rt, resource type.

Entity Notation (EN) [19, 20] is another lightweight data format for distributed systems. It supports Semantic Web technologies and has been designed to be compatible with RDF and Web Ontology Language. EN has almost equal expressivity with RDF and N3 on the data exchange level. As can be seen from Table IX, the complete EN format resembles the triple structure of these representations. ‘Sensor’ and ‘accmagSensor01’ define the type and ID of the sensor; each line below contains a property and an object (i.e., value) for that subject. Type information about the subject is mandatory for complete EN packets, because it enables linking this packet to higher level ontology knowledge defining entity hierarchies and relations, types of properties, and so on. The short EN format is based on templates, IDs, and variables. The upper part of Table IX presents a complete EN packet describing temperature, acceleration, and magnetic field data, while the lower part of Table IX presents a corresponding short EN packet. Similarly, Table X presents complete and short EN packets for location sensor data. Universally unique ID in the short packet is used to identify the template. It is an ID that is guaranteed to be unique across space and time.

Table XI presents a template for transferring a short EN packet to a complete EN packet shown in Table IX. This template basically contains a description of the constant part of this complete EN packet and placeholders for the variable items. The corresponding short packets contain a value for each variable item. A sequence of complete EN packets can also be shortened with one template. A short packet sent over communication links needs to contain only a template ID and the variable items. A template can be stored locally in an IoT node decoding EN packets. Otherwise, it can be transferred between IoT nodes with a sequence of EN packets [19]. Transferring EN templates introduces extra encoding and decoding cost, but this is not a considerable amount of extra cost when a template can be utilized for decoding a large amount of short EN packets.

DOI: 10.1002/cpe

Table VI. Location data in different formats of SenML.

Sensor data in Javascript Object Notation

```
{“e”: [ { “n”: “longitude”, “v”: 25.468},
{ “n”: “latitude”, “v”: 65.058}],
“bn”: “locaSensor767”,
“pr”: “http://iot.fi/o#”,
“bt”: “3296123968”,
“rt”: “LocationSensor” }
```

Sensor data in eXtensible Markup Language

```
<senml xmlns=“urn:ietf:params:xml:ns:senml”
pr=“http://iot.fi/o#”
bn=“locaSensor767”
bt=“3296123968”
rt=“Sensor”>
<e n=“longitude” v=“25.468”>
<e n=“latitude” v=“65.058”>
</senml>
```

Sensor data in Efficient XML Interchange

```
3c73656e6d6c ... 59 bytes ... 303538223e
```

bn, base name; bt, base time; pr, prefix; rt, resource type.

Table VII. Temperature, acceleration, and magnetic sensor data in Javascript Object Notation for Linked Data.

Sensor data in compacted Javascript Object Notation for Linked Data format

```
{
“@context”:
{
“i”: http://iot.fi/o#”,
“accX”: “i:accX”, “accY”: “i:accY”, “accZ”: “i:accZ”,
“magX”: “i:magX”, “magY”: “i:magY”, “magZ”: “i:magZ”,
“temp”: “i:temp”, “timeStamp”: “i:timeStamp”
},
“@id”: “i:accmagSensor01”,
“@type”: “i:Sensor”,
“accX”: “618”, “accY”: “319”, “accZ”: “671”,
“magX”: “123”, “magY”: “234”, “temp”: “22.5”,
“timeStamp”: “2012-05-18T12:00:00”
}
```

Sensor data in context-referenced Javascript Object Notation for Linked Data format

```
{
“@context”: “http://iot.fi/json-ld/contexts”,
“@id”: “i:accmagSensor01”,
“@type”: “i:Sensor”,
“accX”: “618”, “accY”: “319”, “accZ”: “671”,
“magX”: “123”, “magY”: “234”, “temp”: “22.5”,
“timeStamp”: “2012-05-18T12:00:00”
}
```


DOI: 10.1002/cpe

Table VIII. Location sensor data in Javascript Object Notation for Linked Data.

Sensor data in compacted Javascript Object Notation for Linked Data format

{
 "@context":
 {
 "i": "http://iot.fi/o#",
 "ownerID": "i:ownerID",
 "longitude": "i:longitude", "latitude": "i:latitude"
 },
 "@id": "i:locaSensor767",
 "@type": "i:LocationSensor",
 "ownerID": "Alice",
 "longitude": "25.468", "latitude": "65.058"
 }

Sensor data in context-referenced Javascript Object Notation for Linked Data format
{ "@context": "http://iot.fi/json-ld/contexts",   "@id": "i:locaSensor767",   "@type": "i:LocationSensor",   "ownerID": "Alice",   "longitude": "25.468", "latitude": "65.058"   }

Table IX. Temperature, acceleration, and magnetic sensor data in Entity Notation.

Sensor data in Entity Notation complete packet
[http://iot.fi/o#Sensor   http://iot.fi/o#accmagSensor01   http://iot.fi/o#timeStamp "2012-05-18T12:00:00"   http://iot.fi/o#accX "618" http://iot.fi/o#accY "319" http://iot.fi/o#accZ "671"   http://iot.fi/o#magX "123" http://iot.fi/o#magY "234" http://iot.fi/o#magZ "345"   http://iot.fi/o#temp "22.5"]
Sensor data in Entity Notation short packet
[urn:uuid:311b4e80-d9fd-11de-8a39-0800200c9a66   "2012-05-18T12:00:00" "618" "319" "671" "123" "234" "345" "22.5"]

Table X. Location sensor data in Entity Notation.

Sensor data in Entity Notation complete packet
[http://iot.fi/o#LocationSensor   http://iot.fi/o#locaSensor767   http://iot.fi/o#ownerID "Alice"   http://iot.fi/o#longitude "25.468"   http://iot.fi/o#latitude "65.058"]
Sensor data in Entity Notation short packet
[urn:uuid:4e663b23-d0ef-11e2-8b8b-0800200c9a66 "Alice" "25.468" "65.058"]

We compare the semantic expressivity of RDF, N3, SenML, JSON-LD, and EN in Table XII. RDF, N3, JSON-LD, and EN can be mapped to conceptual graphs [21]. Hence, they support ontologies straightforwardly. RDF and N3 have a triple centric structure as the base representation.

DOI: 10.1002/cpe

Table XI. A template example for shortening Entity Notation packets.

A template for transferring EN short packet to EN complete packet shown in Table IV
[http://iot.fi/o#Sensor http://iot.fi/o#accmagSensor 01 http://iot.fi/o#timeStamp ?1 http://iot.fi/o#accX ?2 http://iot.fi/o#accY ?3 http://iot.fi/o#accZ ?4 http://iot.fi/o#magX ?5 http://iot.fi/o#magY ?6 http://iot.fi/o#magZ ?7 http://iot.fi/o#temp ?8]

Table XII. Data format comparison.

	RDF	N3	SenML	JSON-LD	EN
Conceptual graphs	Y	Y	N	Y	Y
Triplet centric structure	Y	Y	N	N	N
Entity centric structure	N	N	Y	Y	Y
Device type	Y	Y	Y	Y	Y
Data types	XSD	XSD	four types	XSD	N
External semantics	Y	Y	Y	Y	Y

RDF, Resource Description Framework; JSON-LD, Javascript Object Notation for Linked Data; EN, entity notation; XSD, XML Schema data types.

JSON-LD and EN follow entity centric approach, but they support mechanisms of transforming to triple structure. SenML has a more arbitrary data structure, which cannot be mapped to a conceptual graph in a similar fashion. Hence, SenML data cannot be utilized by knowledge-based systems as easily as the other alternatives. On the other hand, SenML may be easy to produce by IoT nodes, because it resembles the basic data structures of programming languages. The JSON-LD and short EN format have the same benefit. Describing device types is important for all data formats, because it enables a linking to higher level knowledge. All alternatives can express device types; but EN complete packets require this as a mandatory element. The type of the data (i.e., the physical quantity) can be defined with all these formats, which facilitates associating measured data values to concepts. RDF and N3 support rich XML Schema data types, while SenML allows only four basic data types, that is, floating points, integers, Boolean values, and strings. EN packets do not include data type information, but such information can be accessed from advanced knowledge bases, for example, from some ontologies, when EN packets are integrated into them. All these data formats support external semantic information. RDF and N3 support mechanisms to import additional knowledge; EN does not have a similar mechanism, but its packet structure enables a natural way of knowledge integration. SenML and JSON-LD support additional semantics via Web linking.

In addition to these formats, several other representations have been suggested for semantic annotations. Semantic Sensor Web [22] enables semantic annotations in terms of time, location, and thematic data into the actual sensor data by using Resource Description Framework in Attributes [23]. SemSOS [24] is a similar solution for adding semantic annotations into sensor observations. Finally, semantic extensions are being built for the Product Markup Language [25], which is an XML-based language for describing physical objects in Electronic Product Code Networks. However, XML-based solutions have limitations in supporting semantic interoperability and linking resources to knowledge.

Binary formats for XML such as EXI, X.694 ASN.1 [26], WAP Binary XML Content Format [27], Fast Infoset [28], and Xebu [29] can be used to transfer data from embedded sensors. World Wide Web Consortium recommends EXI, which is a compact representation for the XML Information set and is intended to simultaneously optimize the performance and utilization of computational resources [30]. By using a relatively simple algorithm, it produces encodings of XML event streams. Its simplified mode of operation called schema-informed mode allows embedded devices to work

DOI: 10.1002/cpe

directly with the encoding without the need to work with a full XML parser. Binary formats themselves do not support any semantics, but semantic information in RDF/XML and SenML, for example, can be encoded in binary formats to decrease communication load. We have measured the amount of computation required to produce messages in the EXI binary format that is targeted for resource-constrained environments. These measurements are presented later in this article. There are other solutions for compressing XML. For example, XMLPPM can compress an XML file into 8.25% of the original size [31]. However, many of these solutions increase the computational load, and some compression methods also lose some content. They are not as such suitable for adding semantics to IoT.

Resource Description Framework Header-Dictionary-Triples (HDT) [32, 33] is a binary format for RDF, especially for large RDF data sets. RDF HDT provides a method for encoding RDF documents in a compact manner and supports splitting large RDF documents into chunks. RDF graphs are reorganized into header (optional), dictionary, and triples. HDT Dictionary organizes all vocabularies and HDT Triples, which enables the compression of an RDF graph in a compact form. Similar to EN, unique IDs are assigned to each element in RDF, and prefixes are utilized to shorten URIs. Hasemann *et al*. [34] reported an approach to enable IoT nodes act as services providing sensor data in the RDF HDT format.

3. ENERGY EFFICIENCY

Energy consumption is a key issue for small devices like IoT nodes. Hence, when semantics is added into IoT, energy efficiency is a key criterion when comparing alternative solutions. Energy consumption together with other limited resources is one of the key drivers in wireless sensor network research. However, widely cited surveys, for example, Yick *et al*. [35], Sohrabi *et al*. [36], and Akyildiz *et al*. [37] do not have any explicit discussion on adding semantics to the data. It seems that integrating sensors into Semantic Web has not yet attracted much attention from wireless sensor network researchers.

Lee *et al*. [38] and Siekkinen *et al*. [39] have studied and compared the energy consumption of sensor radios. These results allow estimating the energy consumption of data formats with semantics. Similarly, a large body of knowledge about mobile phone energy consumption is available (e.g., [40, 41]). However, the mobile phone energy consumption is not trivial to quantify as it depends on a number of attributes, such as the wireless interface used (WiFi, 3G, and 4G), the bitrate (higher bitrate saves energy), the shape of the traffic (especially with 3G the tail energy is high), influence of other users, and distance to base station. These are often application and context dependent parameters.

Collecting comparable data from different studies is not easy, but estimating from the data in [39, 40, 42], we can have a rough comparison of the energy utilities of different technologies. Energy utility, that is, the amount of data transferred with one unit of energy, is strongly dependent on the transfer speed. If we assume the uplink data rate 10 kBytes/s, then the energy utility is 900 kBytes/J for Bluetooth, 500 kBytes/J for Bluetooth low energy, 300 kBytes/J for 802.15.4, 25 kBytes/J for WiFi, and 8 kBytes/J for 3G. When the bitrate is slower, the cellular radio and WiFi energy utilities drop quite rapidly while, for example, the Bluetooth low energy utility is rather constant until bitrate drops below 0.1 kBytes/s. It should be noted that these estimations are only about the active transfer phase and they are influenced by many other factors.

In wireless sensor networks, communication is easily the most energy consuming operation. It is reported in [43] that communication is over one thousand times more expensive in terms of energy than performing a trivial aggregation operation. Ni *et al*. [44] observe that energy can be saved by aggregating data when routing sensory data in networks. The semantics of sensory data can be used as a basis for aggregation decisions. Bista and Chang [45] present one of the few studies available that aimed to quantify the energy consumption of in-network data aggregation. They analyze the energy consumption both via analytical models and simulations. Zafeiropoulos *et al*. [46] review data management in sensor network with Semantic Web technologies. Energy consumption is one of the key criteria of their analysis.

DOI: 10.1002/cpe

Figure 1. Architecture of the system in our experiment.

On a general level, Madden *et al*. [47] studied distributed data management in sensor networks. They observe a trade-off between energy consumption and the answer accuracy. Moreover, the sensor type should be taken into consideration when defining the sampling policy. Slow changing data like temperature should be sampled much less frequently than some fast changing data. The energy consumption of pull and push based solutions are also very different.

4. EXPERIMENTS

We measured the resource usage of encoding and decoding for different data formats of the same data (shown on Tables I, III, V, VII, and IX) in a sensor system. As shown in Figure 1, this system consists of two sensors and a knowledge processing component on a PC. Sensor A encodes the different formats and sends them to sensor B. Sensor B decodes the received data formats to easy-to-use formats. For instance, EN packets are converted to RDF triples, and EXI packets are converted to XML documents. RDF, N3, SenML in XML and JSON, and JSON-LD are considered as easy-to-use formats, so sensor B simply forwards them. All these packets are sent from sensor B to a knowledge processing component on a PC and integrated into OntoSensor [48] ontology. As a result, the data generated by IoT nodes are compatible with knowledge systems, which can reason additional knowledge and actions based on these data.

Sensor A measures temperature, as well as acceleration and magnetic field in three dimensions. The node consists of an Atmel's 8-bit (Atmel Corporation, San Jose CA, USA) ATmega32 microcontroller (MCU) with 256 kB flash and 128 kB SRAM memory, a 3-axis accelerometer, a 3-axis magnetometer, a thermometer, and a short range radio link. The node has a real-time clock; thus, it can send a timestamp together with the measurements. The firmware of the sensor node has been implemented as a standalone application, and no operating system has been used.

As we were interested in the payload only, we did not use any specific lightweight protocol, but simply created a message and sent it with Bluetooth. All other messages were created by filling the data values in a string, but SenML/EXI messages were encoded using the 'Embeddable EXI implementation in C' software (http://exip.sourceforge.net/). The available EXIP software was used as such, so smaller memory footprint could be achieved by leaving out the functionality not needed in this experiment. All messages were sent to the receiver by calling the write function, which is associated with the low-power radio interface of the sensor node.

Figure 2 presents the packet lengths of different formats communicated between sensors A and B. These packets contain the same data and semantic information. Short EN format is the most compact format, while SenML/EXI is the second shortest packet. Compact JSON-LD is the longest packet format, while the other formats produce somewhat shorter packets. The shortest format (Short EN) is about 28% of the longest format (Compact JSON-LD).

Figure 3 presents the amount of CPU cycles needed to generate the messages by sensor A. It is clear that generating EXI messages requires much more computation. All other messages are produced by filling measurement values in a string.

A typical IoT sensor node is made up of several components, including sensing electronics, a MCU, a transmission chip, and other devices such as LEDs and flash RAMs. Each component may be in one of a set of activity. We consider the energy consumption in each state to be constant. We focus on energy consumptions of two operations, that is, coding/decoding data formats with the MCU and sending/receiving data packets with the transmission chip. The overall energy consumption thus consists of computing energy and communication energy. We leave energy consumptions of sensing electronics, RAMs, and LEDs out of overall energy consumption, because they are independent of data formats.

DOI: 10.1002/cpe

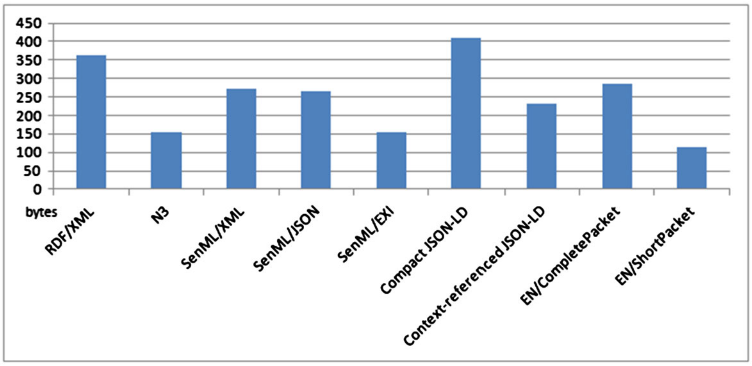

Figure 2. Comparison of packet lengths of different data formats communicated between sensors A and B.

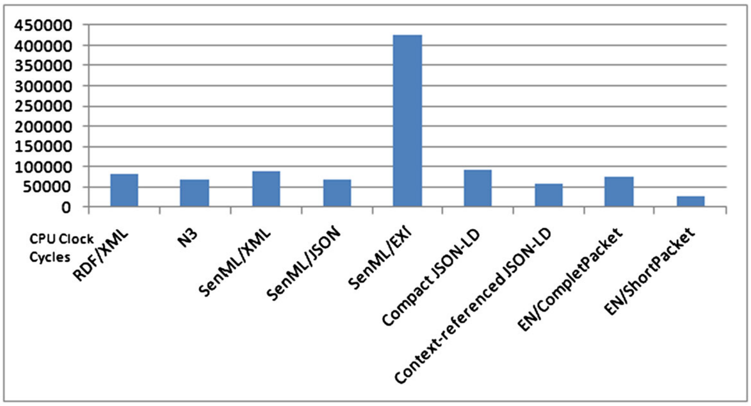

Figure 3. Comparison of CPU cycles for generating different messages by sensor A.

$$E_{Overall} = E_{Comp} + E_{Comm}$$

Computing energy is strongly dependent on MCU cycles and communication energy on message lengths.

$$E_{Overall} = f(Cycle) + f(Length)$$

The electric current needed for the MCU is 1.1 mA at 1 Mhz with 3 V operating voltage, which means an average of

$$0.0011\ A \times 3\ V \times 0.000001\ s = 3.3\ nJ$$

energy consumption for each executed instruction assuming that each instruction takes one clock cycle to execute. The exact number of clock cycles the execution of the code takes depends on the C compiler and the code optimization settings.

The radio interface of the sensor node A is implemented using Bluegiga WT12 Bluetooth Module (Bluegiga, Espoo, Finland). The maximum current of the Bluetooth module is 60 mA at 3 V operating voltage. Many low-power radio solutions exist, which are superior to Bluetooth for wireless connectivity, but as we are interested in the differences between different formats, Bluetooth is a viable selection for this experiment. With WT12 Bluetooth module, each transmitted byte consumes approximately 4-μJ energy. The data rate of WP12 Bluetooth module is 350 kbps when radio frequency communication protocol stack is used. Sending one byte of data is about one thousand times more energy consuming than one process cycle.

As shown in Figure 4, generating SenML/EXI messages requires more energy (MCU Energy in the figure) than other alternatives, but transmission energy consumption for SenML/EXI is among

DOI: 10.1002/cpe

the lowest ones. When comparing overall energy consumption on sensor A, the short EN format requires the least energy, and N3 requires the second least. Generating short EN messages only consumes about 24% of generating SenML/EXI messages, which consume the largest amount of energy.

On sensor B, messages with RDF, N3, SenML in XML and JSON, and JSON-LD data format will be forwarded to knowledge processing component without any decoding. Three data formats, that is, SenML/EXI, EN complete packet, and EN short packet, are decoded to easy-to-use formats. EN packets are converted to RDF triples, and EXI packets are converted to XML documents.

Values from a short EN packet are filled in a template to produce a complete EN packet. The globally unique IDs in the short EN messages enable this processing to be performed unambiguously. Complete EN packets are transformed to RDF, which can directly utilized by knowledge processing component. SenML/EXI packets are transformed to XML. Figure 5 presents how much transmission energy is needed for reception and how much MCU energy is needed for decoding operation on sensor B. Transmission energy consumption equals to overall energy consumptions on sensor B for those data formats that only need simple forwarding. Figure 6 shows a comparison of overall energy consumption for both sensors, including data sending, receiving, encoding, and decoding operations. SenML/EXI processing generates XML and requires additional processing for producing RDF. We can also conclude that the short EN packets require the least energy to produce data in RDF format.

The overall MCU energy consumption for encoding and decoding data depends on complexity of methods, and EXI is clearly the most complex one. The transmission energy consumption of different format scales linearly with the payload size. Short EN format requires the smallest amount of transmission energy, while SenML/EXI requires the second least amount of transmission energy.

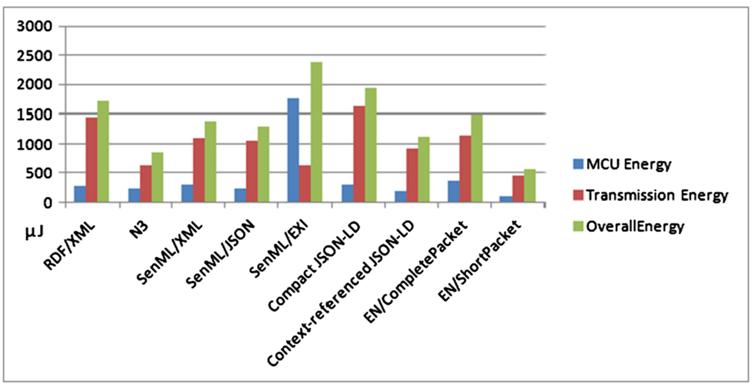

Figure 4. Comparison of energy consumption on sensor A.

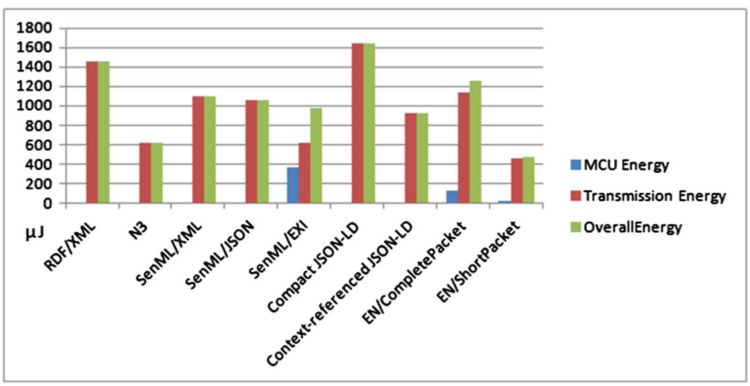

Figure 5. Comparison of energy consumption for decoding operation on sensor B.

DOI: 10.1002/cpe

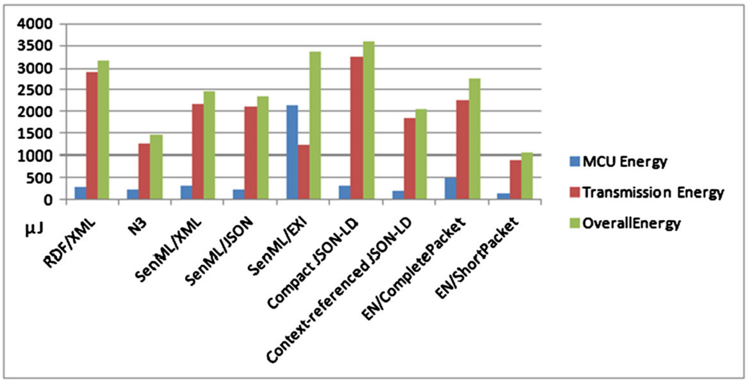

Figure 6. Comparison of overall energy consumption on the sensor system.

However, SenML/EXI requires so much computation that the total energy consumption is the second largest alternative. Two other syntaxes of SenML require a similar amount of energy in our experiment. Compact JSON-LD format consumes the largest amount of overall energy because of its longest messages. RDF/XML messages are second longest. N3 requires less than others but larger than Short EN. Short EN packets consume the least energy among these formats, it consumes 88% of the second best alternative (N3) and 29% of the worst one (Compact JSON-LD). As the messages sent by the locations sensor are quite similar when considering the amount of values and the length of the messages, the location sensor would generate quite similar energy consumption values.

As a conclusion, short EN is the best data format, when energy consumption of an IoT system has to be minimized in general, N3 being the second best. For those systems with limited communication resources but not limited with computing resources, Short EN is the best and SenML/EXI the second best. However, for those systems with limited computing resources, SenML/EXI is not an alternative. If only sensor A is required to minimize energy consumption, short EN and N3 are the best alternatives. If only sensor B is required to minimize energy consumption, for example, it is a very simple gateway without any processing capabilities, then N3 would be the best option. On the other hand, when more widely known data formats are looked for, SenML/JSON and Context-referenced JSON-LD are the best choices in cases like these.

5. DISCUSSION

At its best, research and development on IoT can produce a dynamic and universal network where billions of identifiable things communicate with each other whenever and wherever communication is needed. Things become context-aware, configure themselves, exchange information, and show intelligent behavior when exposed to a new environment and unforeseen circumstances. Intelligent decision-making algorithms enable rapid responses and revolutionize the ways business value is generated [49].

Adding semantics can help to address several challenges in building large IoT systems. Among the challenges listed in [49], semantics help to tackle the challenges related to software, services, and algorithms, that is, ‘to support interoperable machine to machine and “thing” to “thing” interaction over a network.’ Generally, semantics improves interoperability at the application layer, as communicating nodes share the meaning of the communicated data. By adding support to metadata, semantics provides tools for tackling the challenges related to discovery and search engine technologies as well. Identification technologies are also supported by expressing IoT node identities in commonly understood manner. However, as Barnaghi *et al*. [50] have pointed out, ‘the dynamic, heterogeneous and resource-constrained nature of the IoT requires special design considerations to be taken into account to effectively apply the semantic technologies in the IoT’. Adding semantics to IoT is still in its early days. Producing a generic solution on a global scale is a truly challenging task.

DOI: 10.1002/cpe

We are studying the best way to add semantics to IoT data. Thus far, we have studied data formats, their expressive abilities and resource consumptions. Data formats differ in their expressivity, but our running example shows that they all can be utilized for representing simple events in IoT systems. When the expressiveness of different formats is considered, data formats with triplets centric structure (RDF and N3) are more suitable for graph-structured data; data formats with entity centric structure (SenML, JSON-LD, and EN) show advantages when expressing basic programming languages data structures. When these results are combined with the energy consumption measurements, we can state that short EN packets and N3 are the best and the second best options. SenML/EXI can be considered when IoT systems have limited communication resources and rich computing resources. If IoT systems include a very simple gateway without any processing capabilities, N3 would then be the best option. On the other hand, SenML/JSON and context-referenced JSON-LD are the best choices in cases of looking for more widely known data formats. Although our experiment is based on a simple setup, it illustrates how big of a difference a data format can make in energy consumption.

As pointed out by Anastasi *et al*. [51], many schemas have an effect on energy consumption, including data reduction, energy-efficient data acquisition, topology control, power management, and so on. We have so far focused mainly on data formats, which is important for reducing payload of protocols. We consider data formats supporting semantics on their expressivity and resource consumption. Moreover, although a common data format supporting semantics facilitates using IoT data, it is not all that is needed. In addition, the meaning encoded in the messages needs to be shared by all entities producing and consuming the data. That is, ontologies are needed. The existing ontologies, such as OntoSensor [48] and Semantic Sensor Network Ontology [52], offer a good starting point for this work. Semantic Sensor Network Ontology is a well-designed upper level ontology for describing sensors and observations. Moreover, as IoT systems produce large amounts of data, reasoning techniques that scale and infer useful information in a reasonable amount of time are called for. These reasoning techniques need to be deployable into devices with varying computing resources.

When energy consumption is important, protocols need to be considered in addition to the data formats. Transmission Control Protocol, User Datagram Protocol, and Internet Protocol are the basic choices for transferring data in IoT systems. At the application level, hypertext transfer protocol (HTTP) protocol is an obvious choice, specifically when the representational state transfer (REST) architectural style is used. HTTP can be used to transform IoT into Web of Things—by integrating the IoT nodes into the Web and making them available as resources via standard Web mechanisms. Moreover, the RESTful architectural style is a promising approach for IoT systems because of its low complexity and loosely coupled stateless interactions. These two features enable Web servers in the RESTful architecture to be embedded in resource constrained devices and facilitate composing Web services.

However, optimized protocols such as constrained application protocol (CoAP) [53] and 6LoWPAN [54] are more suitable for resource constrained IoT networks for decreasing communication load. CoAP is complementary to HTTP as it is targeted for resource constrained networks instead of traditional IP networks. Power, memory, and computation constraints were taken into account when CoAP was designed. CoAP supports the familiar HTTP methods and a subset of HTTP compatible response codes. CoAP messages are delivered using User Datagram Protocol. It supports asynchronous transactions, although reliable transmission is provided as well with messages requiring acknowledgments. CoAP messages have a short fixed-length binary header that decreases the header overhead and parsing complexity. The fixed header is 4 bytes and can be extended with binary options. This results typically in 10–20 bytes header and even 10 times smaller communication load than HTTP. Devices can be addressed with URIs. Finally, CoAP has built-in resource discovery for discovering and advertising the resources offered by a device, and it realizes a subscribe/notify push model for messaging, in addition to the request-response model. Moreover, CoAP supports proxies that can significantly reduce energy consumption of the system.

In our experiment, we assume that encoded messages are delivered under similar circumstances, that is, using the same protocol, along the same route. In the future IoT scenarios, a large amount of devices exchange data in a variety of architectures [55, 56]. Building these architectures introduces

DOI: 10.1002/cpe

heterogeneity, connectivity, scale, energy management, self-management, privacy, and security challenges [57]. One potential scenario for our future work is a gateway receiving data from several similar sensors, aggregating the data values, and sending the resulting data forward. CoAP will be utilized and compared with other protocols in energy efficiency aspect. Comparing the energy consumption of converting the different formats into Semantic Web, compatible formats would also be interesting. Although these conversions are often made at the server side, some nodes and gateways might utilize Semantic Web technologies. An interesting task would be to study the total energy consumption, when semantics are defined in a SenML data packet with a link, and the additional data are fetched from the given location.

Moreover, we will study the different protocols. As with data formats, protocols can be expected to produce different header lengths and require different amounts of processing. Together with data formats, data aggregation and protocols, different messaging patterns will determine the overall energy consumption when an IoT system is in operation. Publish/Subscribe type messaging and adaptive sampling are two promising approaches.

6. CONCLUSIONS

Semantic technologies enable machine-interpretable representation formalism for describing objects, sharing and integrating information, and inferring new knowledge. In the IoT domain, the addition of semantics helps creating machine-interpretable and self-descriptive data. We believe that Semantic Web technologies, especially RDF-based linked data, will become the de facto standard on the Internet for representing physical world phenomena and activities accessed from IoT nodes, regardless of the application domain. In this article, we focus on investigating different approaches for adding semantics to IoT data. We also evaluate their resource usages, especially energy consumptions, by a sensor system. Our experiments show the variability a data format can make in packet length, MCU circles, and energy consumptions. We will continue this work by studying more complex potential scenarios and messaging patterns.

ACKNOWLEDGEMENTS

This work was supported by TEKES as part of the IoT program of DIGILE (Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT and digital business). The first author would like to thank HPY Research Foundation and Tauno Tönningin Säätiö for funding this work. We thank Dr. Janne Haverinen for his contribution of implementing first version of the system. Johanna Nieminen did the majority of her work share when she was working at Nokia Research Center, Helsinki.

REFERENCES

1. Berners-Lee T, Hendler J, Lassila O. *The Semantic Web*. Scientific American: New York, May 2001.
2. Su X, Riekki J, Haverinen J, Nieminen J, Nurminen JK. Enabling semantics for the Internet of Things—data representations and energy consumptions. *Internet of Things Finland* 2013; **1**(1):28–31.
3. JavaScript Object Notation. Available from: http://www.json.org/ [Accessed on March 2013].
4. RDF Working Group. Resource Description Framework (RDF), 2004. Available from: http://www.w3.org/RD/ [Accessed on March 2013].
5. RDF Vocabulary Description Language 1.0: RDF Schema. Available from: http://www.w3.org/TR/rdf-schema/ [Accessed on March 2013].
6. OWL 2 Web Ontology Language Primer (Second Edition). Available from: http://www.w3.org/TR/owl2-primer/ [Accessed on March 2013].
7. Berners-Lee T (ed.). Notation3 (N3): a readable RDF syntax. Available from: http://www.w3.org/TeamSubmission/n3/ [Accessed on March 2013].
8. Beckett D, Berners-Lee T. Turtle—Tease RDF Triple Language, W3C Candidate Recommendation. Available from: http://www.w3.org/TR/2013/CR-turtle-20130219/ [Accessed on March 2013].
9. N-Triples W3C RDF Core WG Internal Working Draft. Available from: http://www.w3.org/2001/sw/RDFCore/ntriples/ [Accessed on March 2013].
10. Jennings C, Shelby Z, Arkko J. Media types for Sensor Markup Language (SENML). Available from: http://tools.ietf.org/html/draft-jennings-senml-08 [Accessed on March 2013].
11. Schneider J, Kamiya T. Efficient XML Interchange (EXI) Format. Available from: http://www.w3.org/TR/exi/ [Accessed on March 2013].

DOI: 10.1002/cpe

12. Shelby Z. Constrained RESTful Environments (CoRE) Link Format. Available from: http://tools.ietf.org/html/rfc6690 [Accessed on March 2013].
13. Davic I, Steiner T, J Le Hors A. RDF 1.1 JSON Alternate Serialization (RDF/JSON). Available from: https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-json/index.html [Accessed on September 2013].
14. Nathan (ed.). JSN3 Unofficial Draft 27 September 2013. Available from: http://webr3.org/apps/specs/jsn3/ [Accessed on September 2013].
15. W3C. JTriples. Available from: http://www.w3.org/wiki/JTriples, http://www.w3.org/wiki/JTriples [accessed on September 2013].
16. Birbeck N. RDFj: Semantic Objects in JSON. Available from: http://markbirbeck.com/blog/2009/04/20/rdfj-semantic-objects-in-json/ [Accessed on September 2013].
17. Sporny M, Longley D, Kellogg G, Lanthaler M, Lindström N. JSON-LD 1.0 A JSON-based Serialization for Linked Data. Available from: http://www.w3.org/TR/2013/CR-json-ld-20130910/ [Accessed on September 2013].
18. W3C RDF Working Group. JSON Serializations By Example. Available from: http://www.w3.org/2011/rdf-wg/wiki/JSON-Serialization-Examples [Accessed on September 2013].
19. Su X, Riekki J, Haverinen J. Entity notation: enabling knowledge representations for resource-constrained sensors. *Personal and Ubiquitous Computing* 2012; **16**(7):819–834.
20. Su X, Riekki J. Transferring ontologies between mobile devices and knowledge-based systems. *Proceeding of the 10th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, Hongkong, China, 2010; 127–135.
21. Shadbolt N. The Semantic Web Revisited. *IEEE Intelligent Systems*; **21**(3):96–101.
22. Sheth A, Henson C, Sahoo SS. Semantic Sensor Web. *IEEE Internet Computing*; **12**(4):73–83.
23. RDFa 1.1 Primer Rich Structured Data Markup for Web Documents. Available from: http://www.w3.org/TR/xhtml-rdfa-primer/ [Accessed on September 2013].
24. Henson C, Pschorr J, Sheth A, Thirunarayan K. SemSOS: Semantic Sensor Observation Service. *Proceeding of the 2009 International Symmposium on Collaborative Technologies and Systems*, Baltimore, USA, 2009; 18–22.
25. Toma I, Simperl E, Hench G. A joint roadmap for semantic technologies and the Internet of Things. *Proceeding of the 3rd STI Roadmapping Workshop Charting the next Generation of Semantic Technology*, Heraklion, Greece, 2009.
26. International Telecommunication Union. X.694. Available from: http://www.itu.int/ITUT/studygroups/com17/languages/X694.pdf [Accessed on March 2013].
27. Martin B, Jano B. WAP Binary XML Content Format. Available from: http://www.w3.org/TR/wbxml/ [Accessed on September 2013].
28. Fast Infoset: Standard-based Binary XML FastInfoset.Net. Available from: http://www.noemax.com/products/fastinfoset/index.html [Accessed on March 2013].
29. Kangasharju J, Tarkoma S, Lindholm T. Xebu: a binary format with schema-based optimizations for XML data. *Proceeding of 6th International Conference on Web Information Systems Engineering*, New York, USA, 2005; 528–535.
30. Efficient XML Interchange Evaluation. Available from: http://www.w3.org/TR/exi-evaluation/ [accessed on March 2013].
31. Ng W, Yeung LW, Cheng J. Comparative analysis of XML compression technologies. *World Wide Web*; **9**(1): 5–33.
32. Fernández JD, Martínez-Prieto MA, Gutierrez C, Polleres A. Binary RDF Representation for Publication and Exchange (HDT), W3C Member Submission. Available from: http://www.w3.org/Submission/2011/SUBM-HDT-20110330/ [accessed on June 2013].
33. Fernández JD, Martínez-Prieto MA, Gutierrez C. Compact representation of large RDF data sets for publishing and exchange. *Proceedings of the 9th international semantic Web conference*, Shanghai, China, 2010; 193–208.
34. Hasemann H, Kroller A. Pagel M. RDF Provisioning for the Internet of Things. *3rd International Conference on the Internet of Things*, Wuxi, China, 2012; 143–150.
35. Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. *Computer Networks*; **52**(12):2292–2330.
36. Sohrabi K, Gao J, Ailawadhi V, Pottie GJ. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*; **7**(5):16–27.
37. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. *Computer Networks* 2002; **38**(4):393–422.
38. Lee J-S, Su Y-W, Shen C-C. A comparative study of wireless protocols: Bluetooth, UWB, Zigbee, and Wi-Fi. *Proceedings of the 33rd Annual Conference of the IEEE Industrial Electronics Society*, Taipei, Taiwan, 2007; 46–51.
39. Siekkinen M, Hiienkari M, Nurminen JK, Nieminen J. How low energy is Bluetooth low energy? Comparative measurements with ZigBee/802.15.4. *Proceedings of the IEEE Wireless Communications and Networking Conference Workshops*, Istanbul, Turkey, 2012; 232–237.
40. Nurminen JK. Parallel connections and their effect on the battery consumption of a mobile phone. *Proceedings of the 7th IEEE Consumer Communications and Networking Conference*, Las Vegas, USA, 2010; 1–5.
41. Xiao Y, Savolainen P, Karppanen A, Siekkinen M, Ylä-Jääski A. Practical power modeling of data transmission over 802.11g for wireless applications. *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, Passau, Germany, 2010; 75–84.
42. Balasubramanian N, Balasubramanian A, Venkataramani A. Energy consumption in mobile phones: a measurement study and implications for network applications. *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, Chicago, USA, 2009; 280–293.

DOI: 10.1002/cpe

43. Cantoni V, Lombardi L, Lombardi P. Challenges for data mining in distributed sensor networks. *Proceedings of the 18th International Conference on Pattern Recognition*, Hongkong, China, 2006; 1000–1007.
44. Ni LM, Zhu Y, Ma J, Li M, Luo Q, Liu Y, Cheung SC, Yang Q. Semantic sensor net: an extensible framework. *Proceedings of the 3rd International Conference on Computer Network and Mobile Computing*, Zhangjiajie, China, 2005; 1144–1153.
45. Bista R, Chang J-W. Energy-efficient data aggregation for wireless sensor networks. In *Sustainable Wireless Sensor Networks*. InTech, 2010, DOI: 10.5772/13708.
46. Zafeiropoulos A, Spano DE, Arkoulis S, Konstantinou N, Mitrou N. Data management in sensor networks using Semantic Web Tehnologies. In *Data Management in the Semantic Web*. Nova Science Publishers, Incorporated, 2009; 97–118.
47. Madden S, Franklin M, Hellerstein J, Hong W. TinyDB: an acquisitional query processing system for sensor networks. *ACM Transactions on Database Systems* 2005; **30**(1):122–173.
48. Russomanno DJ, Kothari RC, Thomas AO. Building a sensor ontology: a practical approach leveraging ISO and OGC models. *Proceedings of the 2005 International Conference on Artificial Intelligence*, Las Vegas, USA, 2005; 637–643.
49. Sundmaeker H, Guillemen P, Friess P, Woelfflé S (eds). *Vision and challenges for realising the Internet of Things*. Cluster of European Research Projects on the Internet of Things: Brussels Belgium, 2010.
50. Barnaghi P, Wang W, Henson C, Taylor K. Semantics for the Internet of Things: early progress and back to the future. *International Journal on Semantic Web and Information Systems* 2012; **8**(1):1–21.
51. Anastasi G, Conti M, Francesco DM, Passarella A. Energy conservation in wireless sensor networks: a survey. *Ad Hoc Networks* 2009; **7**(3):537–568.
52. W3C semantic sensor network incubator group. Semantic Sensor Network Ontology. Available from: http://www.w3.org/2005/Incubator/ssn/ssnx/ssn [Accessed on September 2013].
53. Shelby Z, Hartke K, Bormann C. Constrained Application Protocol (CoAP). Available from: http://tools.ietf.org/html/draft-ietf-core-coap-18 [Accessed on September 2013].
54. Shelby Z. *6LoWPAN: the wireless embedded Internet*. Wiley Series on Communications Networking & Distributed Systems: Hoboken, USA, 2009.
55. Ning H, Wang Z. Future Internet of Things architecture: like mankind neural system or social organization framework? *IEEE Communications Letters* 2011; **15**(4):461–463.
56. Evangelos AE, Tselikas DN, Boucouvalas CA. Integrating RFIDs and smart objects into a unified Internet of Things architecture. *Advances in Internet of Things* 2011; **1**(1):5–12.
57. Zorzi M, Gluhak A, Lange S, Bassi A. From today's Intranet of Things to a future Internet of Things: a wireless-and mobility-related view. *Wireless Communications IEEE* 2010; **17**(6):44–51.

DOI: 10.1002/cpe