Adding Semantics to Internet of Things

Xiang Su\textsuperscript{1*}, Jukka Riekki\textsuperscript{1}, Jukka K. Nurminen\textsuperscript{2}, Johanna Nieminen\textsuperscript{3}, Markus Koskimies\textsuperscript{1}

\textsuperscript{1}Department of Computer Science and Engineering, University of Oulu, Finland
\textsuperscript{2}Department of Computer Science and Engineering, Aalto University, Espoo, Finland
\textsuperscript{3}TeliaSonera, Sturenkatu 16, 00510, Helsinki, Finland

SUMMARY

The development of Internet of Things 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 Internet of Things. 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 difference a data format can make in energy consumption. Copyright © 2010 John Wiley & Sons, Ltd.

Received . . .

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 and will communicate with other peers and servers in the Internet. The resulting uniform access to these devices and objects, that is, uniform access to IoT nodes, 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.

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” [2]. 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. Since 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, since IoT nodes are often small devices 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 Internet of Things without breaking the constraints on resource usage. Common semantic technologies 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. We focus on different data formats enabling semantics; we compare semantic expressivity of these formats and measure the resources needed to encode and decode them in this sensor system.

We do not explicitly discuss protocols or architectures in this article. They have a significant effect to resource usage, but we focus on resources needed to encode and decode different data formats when messages are sent and received. We do not consider ontologies either, although they determine the meanings encoded in the data format. Instead, 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 [1], and report 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, like 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 consumers of data. “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 the existing Semantic Web tools can 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, object). A triple denotes that a subject has a property whose value is the object. IoT data usually originates from devices, humans and other entities in the physical world, and refers to attributes of phenomenons, 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 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 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) and a user identifier. Other sensor data can be represented in a similar way. Table I presents in RDF/XML format the first sensor data example 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 (RDFS)[5] and Web Ontology Language (OWL)[6] provide a standard vocabulary for defining classes and relationships.
Table I. Temperature, Acceleration and Magnetic Sensor Data in RDF

RDF

<?xml version="1.0" encoding="UTF-8"?>
  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:temp>22.5</i:temp>
  </i:Sensor>
</rdf:RDF>

Table II. Location Sensor Data in RDF

RDF

<?xml version="1.0" encoding="UTF-8"?>
  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>

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.

among classes that enable high level inference. Hence, when IoT nodes express data in RDF, these languages facilitate realizing advanced semantic processing.

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 has a flexible language with strong expressivity capability going beyond the RDF model, Turtle is an RDF-compatible subset of N3 while N-Triples has constrained expressivity. N3 and Turtle have shorthand 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 consisting of attributes and an array of entries. Each entry, in turn, consists of attributes such as a unique identifier 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...
Table IV. Location Sensor Data in N3

```xml
@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.
```

Table V. Temperature, Acceleration and Magnetic Sensor Data in different formats of SenML

**Sensor Data in JSON**

```json
{ "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 XML**

```xml
<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 EXI**

```exi
0x80419cd95b ... 145 bytes ... 0801001000
```

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 identifier). Table VI presents the location sensor data in SenML. "pr" stands for prefix in Table V and VI, which can be transformed to xml:base="http://iot.fi/o" when SenML data are transformed to RDF/XML.

Entity Notation (EN) [13][14] is another lightweight data format for distributed systems. It supports Semantic Web technologies and has been designed to be compatible with RDF and OWL. EN has almost equal expressivity with RDF and N3 on the data exchange level. As can be seen from Table VII, 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, etc. The short EN format is based on templates, identifiers and variables. The upper part of Table VII presents a complete EN packet.
Table VI. Location Data in different formats of SenML

Sensor Data in JSON

```json
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 XML

```
  <e n="longitude" v="25.468">
    <e n="latitude" v="65.058">
      ...
      ...
      ...
    </e>
  </e>
</senml>
```

Sensor Data in EXI

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

Table VII. Temperature, Acceleration and Magnetic Sensor data in EN

Sensor Data in EN 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 "761"  
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 EN Short Packet

```
[urn:uuid:311b4e80-d9fd-11de-8a39-0800200c9a66  
"2012-05-18T12:00:00"  
"618" "319" "761" "123" "234" "345" "22.5" "0"]
```

Table VIII. Location Sensor data in EN

Sensor Data in EN 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 EN Short Packet

```
[urn:uuid:4e663b23-d0ef-11e2-8b8b-0800200c9a66 "Alice" "25.468" "65.058"]
```

describing temperature, acceleration and magnetic field data, while the lower part of Table VII presents a corresponding short EN packet. Similarly, Table VIII presents complete and short EN packets for location sensor data. UUID in the short packet is used to identify the template. It is an identifier that is guaranteed to be unique across space and time.

Table IX presents a template for transferring a short EN packet to a complete EN packet. This template basically contains a description of the constant part of this complete EN packet and

Copyright © 2010 John Wiley & Sons, Ltd.  
Prepared using cpeauth.cls  
DOI: 10.1002/cpe
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 identifier 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. [13] Transferring EN templates introduces extra encoding and decoding cost, but this is not considerable amount of extra cost when a template can be utilized for decoding a large amount of short EN packets.

We compare the semantic expressivity of RDF, N3, SenML and EN in Table X. RDF, N3 and EN can be mapped to conceptual graphs [15] straightforwardly, as they all have a (subject, property, object) triplet structure as the base representation. Hence, they support ontologies. SenML has a more arbitrary data structure, which cannot be mapped to a conceptual graph in a similar fashion. Hence, SenML data can not 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 short EN format has the same benefit. Describing device types is important for all data formats, because it enable a linking to higher level knowledge. All alternatives can express device types; among them it’s mandatory in EN complete packets. 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 (XSD), while SenML allows only four basic data types, i.e., floating points, numbers, 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 an ontology, 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 supports additional user defined semantic attributes.

In addition to these formats, several other representations have been suggested for semantic annotations. Semantic Sensor Web [16] enables semantic annotations in terms of time, location, and thematic data into the actual sensor data by using RDFa [17]. SemSOS [18] is a similar solution for adding semantic annotations into sensor observations. Finally, semantic extensions are being built for the Product Markup Language (PML)[19], 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 like EXI, X.694 ASN.1 [20], WAP Binary XML Content Format [21], Fast Infoset [22], and Xebu [23] can be used to transfer data from embedded sensors. W3C 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 [24]. Using a relatively simple algorithm, it produces encodings of XML event streams. Its simplified mode of operation called schemainformed mode allows embedded devices to work 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.

For representations that utilize XML, communication load can be decreased also by compressing XML. For example, XMLPPM can compress an XML file into 8.25% of the original size [25]. However, these solutions increase the computational load and some compression methods also loose some content. Data serialization languages like JSON are not as such suitable for adding semantics to IoT, as they do not have any general format that can be transformed into a knowledge representation. However, they can be used by other formats, like in SenML/JSON.

RDF Header-Dictionary-Triples (HDT) [26][27] is a binary format for RDF, especially for large RDF data sets. RDF HDT provide 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 comprises the pure structure of an RDF graph in a compressed form. Similar to EN, unique IDs are assigned to each element in RDF and prefixes are utilized to shorten URIs. Hasemann et al. [28] 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 e.g. Yick et al. [29], Sohrabi et al. [30], and Akyildiz et al. [31] 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. [32] and Siekkinen et al. [33] 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. [34][35]. 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, 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 [34][33][36] we can have 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 (BLE), 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 e.g. the BLE energy utility is rather constant until bitrate drops below 0.1 kBytes/s. We have to notice 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 [37], that communication is over one thousand times more expensive in terms of energy than performing a trivial aggregation operation. Ni et al. [38] 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 [39] present one of the few studies 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. [40] review data management in sensor network with Semantic Web technologies. Energy consumption is one of the key criteria of their analysis.

On a general level, Madden et al. [41] 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-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 and SenML in XML and JSON are considered as easy-to-use formats, so Sensor B simply forwards them. All these packets are send from Sensor B to a knowledge processing component on a PC and integrated in to OntoSensor [42] ontology. As a result, the data generated by IoT nodes is compatible with knowledge systems which can reason additional knowledge and actions based on this data.

Sensor A measures acceleration and magnetic field, both in three dimensions, and temperature as well. The node consists of an Atmel’s 8-bit ATmega32 microcontroller with 256kB flash and 128kB 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, so 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 yet use any specific protocol, but simply created a message and sent it. All other messages were created by filling the data values into 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 Sensor A and Sensor 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. RDF/XML is the longest packet format, while the other formats produce somewhat shorter packets. The shortest format (Short EN) is about 31.38% of the longest format (RDF/XML).
Figure 2. Comparison of packet lengths of different data formats communicated between Sensor A and Sensor B.

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

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.

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

\[ 0.0011A \times 3V \times 0.000001s = 3.3 \text{ nJ} \]

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. The maximum current of the Bluetooth module is 60 mA at 3V 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 micro J energy. The data rate of WP12 Bluetooth module is 350 kbps when RFCOMM 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 the lowest ones. When comparing overall energy consumption on Sensor A, the short EN format requires the least energy and other alternatives consume at least double of that amount. Generating short EN messages only consumes about 25.82% of generating SenML/EXI messages, which consume the largest amount of energy.

Sensor B decodes 3 data formats to easy-to-use representations. Values from a short EN packet are filled in a template to produce a complete EN packet. The globally unique identifiers in the short EN
messages enable this processing to be done 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 MCU energy needed for decoding operation on Sensor B and Figure 6 shows a comparison of overall energy consumption for both sensors, including encoding and decoding operations. SenML/EXI processing generates XML and requires additional processing for producing RDF. We can 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 most complex one. The transmission energy consumption of different formats scales linearly with the payload size. Short EN format requires smallest amount of transmission energy, while SenML/EXI requires second least amount of transmission energy.
However, SenML/EXI requires so much computation that the total energy consumption is larger than for the two times longer RDF/XML messages. RDF, N3, SenML/XML and complete EN consume similar amount of energy. Short EN packets only consume less than half of the second best alternative (N3), and consumes the least energy among these formats. 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.

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. [43]

Adding semantics to IoT is one step in reaching this vision, but it is still in its early days. In general, 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. However, as Barnaghi et al.[44] has pointed “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.” Producing a generic solution on a global scale is a truly challenging task.

We are studying the best way to add semantics to IoT data. So far we have studied data formats, their expressive abilities and resource consumption. Our experiment shows that short EN packets have the most compact format and require least energy. Other formats require more than double of the amount short EN packets need. SenML/EXI requires so much computation that the total energy consumption is larger than for the two times longer RDF/XML messages. Although this experiment was quite simple, it illustrates how big difference a data format can make in energy consumption.

As pointed out by Anastasi et al.[45], many schemas have an effect to energy consumption, including data reduction, energy-efficient data acquisition, topology control, power management etc. 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, like OntoSensor [42] and Standard Ontology for Ubiquitous and Pervasive Applications (soopa)[46], offer a good starting point for this work. 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.

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. 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 is 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

We believe that Semantic Web technologies, especially RDF, will become the de-facto standard in Internet for representing physical world phenomena and activities accessed from IoT nodes, regardless 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 shows the variability a data format can make in packet length, MCU circles and energy consumption. We will continue this work with studying more complex potential scenarios and messaging patterns.

6.1. Acknowledgements

This work was funded by the Internet of Things (IoT) program funded by TIVIT and Tekes. TIVIT is the Strategic Centre for Science, Technology and Innovation in the Field of ICT and Tekes is the Finnish Funding Agency for Technology and Innovation. The first author would like to thank HPY Research Foundation and Tauno Tönnin in Säätiö for funding. We thank Janne Haverinen for 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

5. RDF Vocabulary Description Language 1.0: RDF Schema, http://www.w3.org/TR/rdf-schema/ [accessed on March 2013]
6. OWL 2 Web Ontology Language Primer (Second Edition), http://www.w3.org/TR/owl2-primer/ [accessed on March 2013]
34. Nurminen, J.K. Parallel Connections and their Effect on the Battery Consumption of a Mobile Phone, 7th IEEE Consumer Communications and Networking Conference, Jan, 2010, pp. 1-5.
35. Xiao, Yu and Savolainen, Petri and Karppanen, Arto and Siekkinen, Matti and Ylä-Jääski, Antti, Practical power modeling of data transmission over 802.11g for wireless applications, Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking, pp.75-84.