

# SEMANTIC SUPPORT FOR RESOURCE-CONSTRAINED ROBOT SWARM

Xiang Su, Jukka Riekkı and Janne Haverinen

*Intelligent Systems Group and Infotech Oulu, University of Oulu, FIN-90014, Finland  
Xiang.Su@ee.oulu.fi, Jukka.Riekkı@ee.oulu.fi, Janne.Haverinen@ee.oulu.fi*

Keywords: Robot Swarm, Entity Notation, Lightweight Data Representation, Ontology, Inference.

Abstract: Semantic Web technology could offer lots of intelligent functionality to multi-robot systems. But limited processing power and storage capability of unsophisticated robots do not necessary allow them to support and process Semantic Web technology. In this paper, we propose a novel solution to provide semantic support for resource-constrained robots. Entity Notation is a lightweight data representation which can be employed to transfer data between resource-constrained robots and intelligent applications at server side. Resources-constrained robots only need to handle the lightweight Entity Notation while intelligent applications handle the more advanced knowledge representation. When the Entity Notation is used, the transfer between the robots and applications is unambiguous and lossless. In this way, an ontology and ontology-based inference at server side can improve the capabilities of the robot swarm. We present a simulator and discuss the future work.

## 1 INTRODUCTION

Research in multi-robot systems is producing more and more robot swarm systems containing up to hundreds of autonomous robots. Robot swarms can be utilized in numerous interesting domains, like space exploration, search and rescue operations, cleaning, and other everyday applications. Compared with individual complex autonomous robots, swarms of simple and cost-efficient robots, provide robustness against failure of individual robots and the power of parallel operation.

To build the complex and intelligent structure of a robot swarm, coordination among spontaneous swarms, like stigmergy (Holland and Melhuish 1999), or environment-supported coordination is essential. In this paper, we present how semantic support can be provided for resource-constrained robots to facilitate their coordination. Basically, the semantic support enables reasoning the actions for achieving the swarm's goal.

Semantic Web technology is an ideal candidate for implementing the reasoning, when information can be expressed using a Semantic Web-based formal context model – an ontology model (Studer, Benjamins and Fensel 1998). But traditionally, Semantic Web technology needs more processing power and memory than small robots have. Hence

the robots are not capable to processing knowledge representations such as Resource Description Framework (RDF) (W3C.org 2004), not to mention running the advanced reasoners or other applications utilizing the representations.

This paper focuses on how resource-constrained robots can be connected to Semantic Web-based intelligent applications that have the required resources for processing the knowledge representations and running the reasoners. We consider how information can be transferred from resource-constrained robots to an advanced knowledge representation, how actions can be deduced, and how the actions can be transferred to the robots performing them.

For transferring information from resource-constrained robots to intelligent systems we propose a lightweight representation called Entity Notation (EN). The key idea is that the robots handle the lightweight EN while the intelligent systems handle the advanced knowledge representation – and an unambiguous lossless transform can be performed between these two representations. We develop for the knowledge representation a context information ontology for describing robots, sensors, persons, and other important entities of an indoor office environment. This ontology allows us to use Semantic Web technologies in deducing actions for the robots. We utilize rule-based inference for

deducing actions for robot swarm.

The rest of this article is organized as follows. Section 2 introduces the architecture of EN-based communication. Section 3 presents the Entity Notation in brief. Section 4 presents the context information ontology model. Section 5 describes the rules and our inference mechanism. Section 6 presents the simulator for testing the system. Section 7 introduces the related work and Section 8 concludes the paper and suggests future work.

## 2 ENTITY NOTATION-BASED COMMUNICATION ARCHITECTURE

Figure 1 presents a complete loop of EN-based communication architecture for a robot system. EN packets are utilized in all communication acts between a robot swarm and a swarm server. This robot swarm system consists of cleaning robots, RFID tags, sensors, an air conditioner, and the swarm server. A cleaning robot can measure the light level and the temperature of a room and read data from and write data to RFID tags. Robots and sensors can communicate with the swarm server. The swarm server includes ontology models, EN composer/decomposer, and inference engine components. It provides semantic support to robots by receiving data from robots and sending the actions to be performed back to corresponding robots. This EN-based feedback loop is a key feature of adaptive robot system.

In the application scenario shown in Figure 1, the RFID tag in one room is marked with the timestamp of cleaning work inside this room. When a cleaning robot goes inside this room, it reads the RFID tags of the room, and measures whether light is on in this room. This information can be transferred to the swarm server as EN packets. The swarm server decomposes the EN packets and transfers them to RDF statements in a lossless way. When the context information ontology model is supplied with new RDF statements, the reasoner is triggered, the deduced statements are composed into EN format and sent to the devices. Deduced statements can be some information to a specific device, like message “Cleaning robot521 cleans the room TS354” should be sent to Cleaning robot521. The statements can also be some general information, like “Room TS354 is clean.” should be sent to all cleaning robots subscribed room cleanness information. In the process of compose/decompose EN packets, we use communicative acts ontology

model to standardize the communicative acts between robots and swarm server, which could be attached as the first EN packet to indicate the type of the following packet sequence. For example, the ‘ifRoomSituation’ packet will indicate that the following packet will inform the situation of a specific room, while the ‘ifCleanRoom’ packet will follow with a packet to inform a specific cleaning robot an action of cleaning a room.

Another use case of EN-based communication in figure 2 is that, a temperature sensor measures the temperature in room TS354, and sends the measurement data to the same swarm server by utilizing EN. The reasoner could deduce that “Air conditioner of TS354 adjusts the temperature to 23 °C.” or “TS354 has normal temperature.”

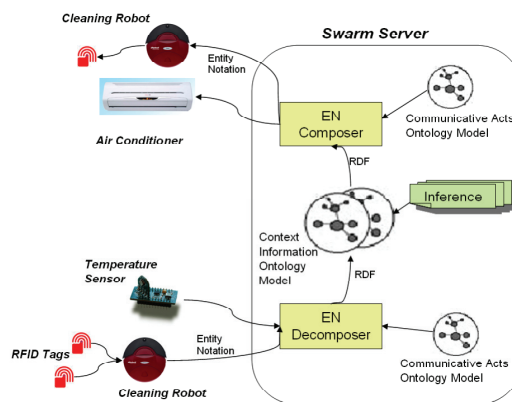


Figure 1: Loop of EN-based Communication.

## 3 ENTITY NOTATION

Entity Notation is introduced briefly in this section and more detailed description can be found in the publication by Riekk, Su and Haverinen (2008). EN was designed as a lightweight representation for exchanging information between nodes of a distributed system. We started to develop EN as a general representation that can be used by any resource-constrained device, but now we are focusing on mobile robots. It’s not a protocol, hence we assume EN packets are payload of some protocols delivering the packets, like HTTP or TCP/IP.

The most important goals for developing EN are that, on the one hand, it can be used over simple communication links by resource-constrained devices; on the other hand, the packet content can be used in a straightforward fashion by intelligent system. For example, it should be possible to

transform the packets into RDF format in an unambiguous fashion.

We achieve these goals as follows. To support resource-constrained robots and limited communication links, EN offers short packets that in their simplest form can be less than 40 bytes long. To allow the packet content to be utilized by Semantic Web applications, the EN offers complete packets that can be transferred to advanced knowledge representations – RDF unambiguously and losslessly. Hence, packet content can be delivered to a reasoner processing knowledge in Web Ontology Language (OWL) format. Complete packets and short packets can be transferred between each other straightforwardly and unambiguously.

The complete EN packet format closely follows the RDF format, which contains a sequence of (subject, predicate, object) triplets. The EN, as its name implies, describes entities. An entity is some identifiable whole, physical or digital. For example, a robot, a sensor, an RFID tag, a sending node, a recipient, a person, a measurement, a message, and a user terminal are entities.

An EN packet is a sequence of entity descriptions. Each description specifies the type of an entity, a unique entity identifier, and a number of property triplets (name, type, value). XML Schema types are used for describing the data type of property value. An entity description is of the form:

```
[EntityType EntityID
PropertyName PropertyType PropertyValue
...
PropertyName PropertyType PropertyValue
]
```

An entity description contains a set of triplets about a single entity. *EntityId* specifies the subject, *PropertyName* is a predicate, and *PropertyValue* is an object. *EntityType* and *PropertyType* specify the types of the corresponding subject and object. It is necessary that elements in EN packet are self-describing. *PropertyValue* in EN can be an identifier of the other entity, and additional statements can give information about those entities. This allows relations to be represented. The relations can be between physical entities, e.g. a robot is in a room. Or, the relations can be between digital entities. For example, the first entity of a message can specify a list and refer to all list member entities.

Here is an example in which a room entity description refers to a temperature sensor in that room. For convenience, in this and all the following examples we replace "http://ee.oulu.fi/o/" by "EE" and "http://www.w3.org" by "W3":

```
[EE#Room EE#room1879
EE#hasTempSensor EE#EntityId
EE#tempSensor234
]

[EE#TempSensor EE#tempSensor234
EE#tempValue W3/2001/XMLSchema#float
"23.5"
EE#timestamp
W3/2001/XMLSchema#dateTime
"2007-10-22T12:00:00-17:00:00"
]
```

The type EntityID is an exception in the type system as all other types are XML Schema types. EntityId specifies the corresponding property value to be an entity identifier – not a literal URI (that would be the XML Schema type anyURI). Literals are enclosed in double quotes, as some data types such as base64Binary data type can contain white space but not double quotes. EN has the ability to carry any primitive data type defined in XML Schema types, including text data, binary data, et al.

A complete packet can be transferred to a RDF description in a straightforward fashion because of their natural relationship. The entity type can be represented as a resource class (using the rdf:type property) and the entity identifier is the identifier of the resource. Property names and values can be mapped directly, and the property type can be represented by adding a rdf:datatype attribute to the property element. Here is the RDF graph corresponding to the example above.

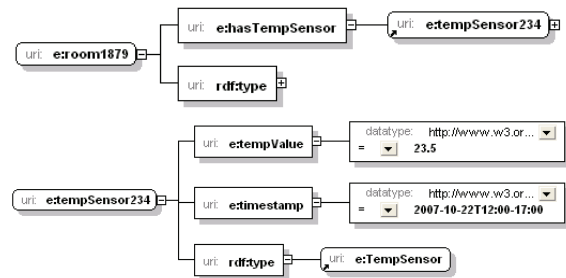


Figure 2: RDF graph representing a sensor’s data package.

Complete EN packets can be quite long because of meaningful type names and URIs. To shorten the complete packets, we suggest templates and prefixes. The basic idea is that a template contains a description of the constant part of a packet and placeholders for the variable parts that differ in each packet. The packet sent over the communication link needs to contain only the template identifier and the changing information. A complete packet can then be assembled by replacing the template’s

placeholders with the values contained in the package. Prefixes are used where the values are URIs. Such as the above example of room entity and temperature sensor inside, if we have for the packet the templates:

```
[EE#Room EE#room1879
EE#hasTempSensor EE#EntityId ?1
]
```

and

```
[EE#TempSensor EE#tempSensor234
EE#tempValue W3/2001/XMLSchema#float
?1
EE#timestamp
W3/2001/XMLSchema#dateTime ?2
]
```

And then template has the identifier urn:uuid:ad7g38 and urn:uuid:5e76af, we can represent the packets as:

```
[urn:uuid:ad7g38 EE#tempSensor234]
[urn:uuid:5e76af
"23.5"
"2007-11-22T12:00:00-17:00:00"]
```

Robots and swarm server can have predefined templates for each packet they can send. Usage of short packets can either be decided by the engineer at design time or the robots can use complete packets by default and agree the short format during communication. The negotiation of EN packets is not in the scope of this paper.

#### 4 THE CONTEXT INFORMATION ONTOLOGY MODEL

Context information gathered from environment can increase a robot swarm’s capabilities by enabling them to adapt to the conditions of environment. By context, we refer to any information that can be used to characterize the situation of an entity (Dey and Abowd 2000).

An ontology-based approach lets us describe contexts semantically, and share common understanding of context among devices, users, and services. Here, ontology refers to the formal, explicit description of concepts, which are often conceived as a set of entities, relations, instances, functions, and axioms. Our context information ontology model facilitates context reasoning, context sharing and reuse. For example, the model can be extended

with information about other devices, or it can be merged with another ontology to describe a larger domain. We will introduce ontology-reasoning and general-purpose rule-based reasoning in the next section.



Figure 3: Main catalogue of Context Information Ontology Model.

In our scenario, we assume a smart house which is equipped with networked robots and devices, such as cleaning robots, RFID tags, temperature sensors, and air conditioners. As shown in Figure 3, context information ontology models the basic concepts for our scenario, like device, environment, person, time and location; describes properties and relationship between these concepts. The context information ontology can be extended to different environment easily depending on other application scenarios.

#### 5 INFERENCE MECHANISM

As discussed above, inference engine will be triggered when the information from robots is added into context information ontology. New knowledge will be produced by reasoner and distributed to devices in EN format. If the EntityID of the deduced statement is one device, EN composer will add one header packet before other packets and deliver them to that device. If its EntityID is not device, the swarm server will distribute it to robots have subscribed this information.

Currently, our inference mechanism supports OWL Lite inference and rule-based inference. OWL Lite inference supports constructs of describing classes, properties, and also relationship between classes and properties. The following example illustrates that a subClassOf relationship among

devices is transitive.

```
(CleaningRobot rdfs:subClassOf Robot)
^ (Robot rdfs:subClassOf Device) ->
(CleaningRobot rdfs:subClassOf Device)
```

Rule-based inference could provide forward chaining, backward chaining and hybrid models to execute rules over RDF graph. Here are example rules to show how robots' data is reasoned and how reasoner deduces an informed action. In these rules, a cleaning robot tests whether the light of room TS354 is on, and reads previous cleaning time from RFID tag. If the light is on, it may mean that someone is working in the office, cleaning robot will go away. If previous cleaning time is not longer than 24 hours, room is no need to be cleaned. In case that the light is off and previous cleaning time is longer than 24 hours, cleaning robot will clean this room.

```
(roomTS354, lightIntensity, on) ->
(cleaningRobot521, goAway, roomTS354)

(rfidTS354, previousCleaningTime, ?PT)
^ lessThan(currentTime-PT, 24hours) ->
(cleaningRobot521, goAway, roomTS354)

(roomTS354, lightIntensity, off) ^
(rfidTS354, previousCleaningTime, ?PT)
^ greaterThan(currentTime-PT, 24hours)
-> (cleaningRobot521, clean, roomTS354)
```

## 6 SIMULATOR

We develop a simulator to simulate the EN-based communication between devices and provide the semantic support from the swarm server. We create four types of devices in the simulator: cleaning robot, temperature sensor, a wireless device designed for reporting the patient's well-being to nurses (Riekkilä et al. 2007) and a location sensor. In the indoor robot system scenario we described in this paper, we pay attention to cleaning robot and temperature sensor in the simulator.

In the simulator, the user can specify messages for any device, assemble short packets or complete packets, and send them to the swarm server. The swarm server transforms packets into RDF format, adds them into context information ontology model, and reasons on data from the robots. The deduced result will be delivered to the specific device or all devices according to the entity identification of the deduced packet. We add one entity in front of each message to describe the message itself: the type, the

sender, receiver, and a sequence number. This entity is included in template. In a real implementation, such an entity will not be needed if the same information is carried by the lower layer protocol.

Figure 4 shows situations in which the cleaning robot has sent a short packet to the swarm server and got responded action. In the cleaning robot window, user can specify the details of the message: the receiver's ID, message type, and the data content which shows the room situation that the cleaning robot is located in, including previous cleaning time and whether the room's light is on. Cleaning robot can assemble complete or short message, and will get corresponding message format from the swarm server. The swarm server responds the robots' message automatically, and sends the deduced message according to the rules specified in previous section.

The first screenshot in Figure 4 shows a short packet including the previous cleaning time and light on/off as following:

```
[urn:uuid:9c38ee "900"
"2008-05-26T05:00:00" "false"]
```

In this message, the identifier urn:uuid:9c38ee determines the template, 900 is the sequence number, and the last two values are the previous cleaning time of the room and the light off boolean tag. The swarm server deduces that the cleaning robot should clean this room TS354 and informs the cleaning robot to clean it by the following short packet:

```
[urn:uuid:bd61dee5-e9b8-422e-948d-
e7799caae04b "1001" EE#TS354 "pending"]
```

In this message, 1001 is sequence number, and the following value shows the room need to be cleaned and the action status is pending.

In the simulator, the swarm server and the robots can send 17 different packets in total: the server can send requests to the robots, while the robots can reply to the server's requests, and send asynchronous inform to the server. We calculate the length of these 17 packets, and find the result is quite promising. Short EN packet format contains on the average only 12.68% of the characters that a complete package contains and 9.67% of the characters that the corresponding RDF document contains.

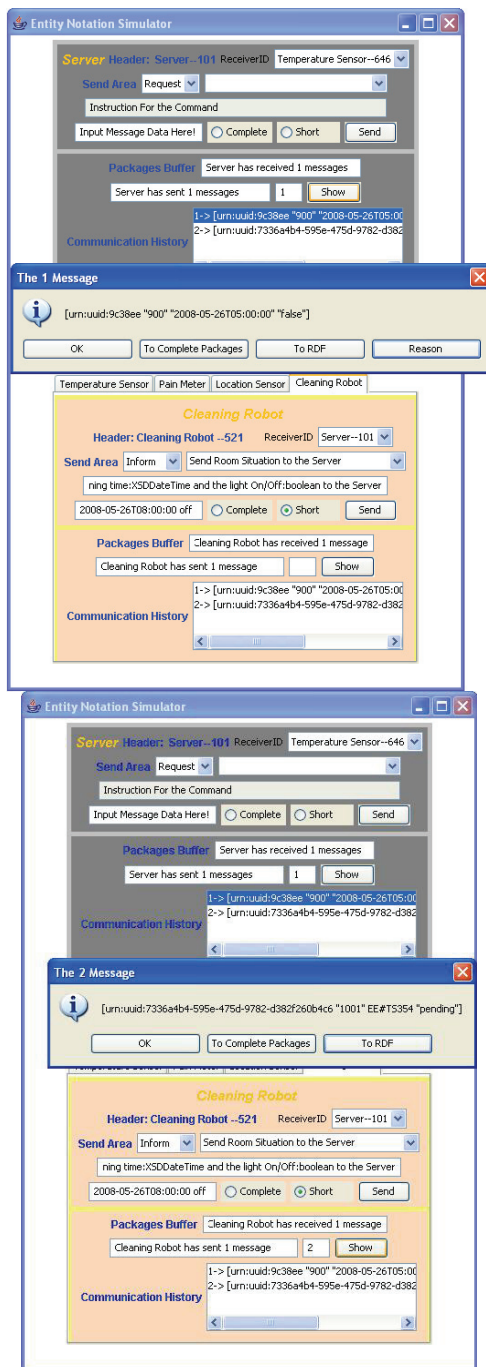


Figure 4: The Simulator Simulates the Cleaning Robot Scenario.

## 7 RELATED WORK

The idea of using Semantic Web technology to facilitate swarm intelligence has been reported by Boley (2007), but no resource-constrained robots

have been integrated into the Semantic Web-based system before. We propose EN as a lightweight data representation for robots to support Semantic Web technology. We have compared EN in (Riekkki, Su and Haverinen 2008), with XML compression technologies, e.g. Gzip (Gzip.org 2003), XMLPPM (Cheney 2000), XMLZip (XMLSolutions 1999), and some markup or data serialization languages, like JSON (json.org 1999), and YAML (yaml.org 2008). The result shows that the EN can almost reach the best compression ratio among XML compression technologies, while requiring minimal amount of computation for composing EN packets. Another alternative is binary XML formats, e.g. EXI (W3C.org 2008), WAP Binary XML Content Format (W3C.org 1999), and Infonet (W3C.org 2004). They are designed to provide a compact representation of XML for communication. They mainly use token-based compression schema in which each element, attribute, and so forth is encoded as a small integer value. Compared with them, EN could reach better compression rate and require less message sterilization processing power. Besides, EN has the advantage over the XML compression and binary XML solutions is that the EN packets are more human readable.

Many systems also adopted the solutions of expressing context information as ontology and using Description Logic (DL) based rules. Chen et al. (2004) proposed OWL to represent context information, and they utilized Java Expert System Shell (Jess) (Friedman-Hill 2008) to realize rule-based reasoning. Want et al. (2004) also modelled context by OWL in semantic Space project. They designed a two-layer context model for expressing context information, and reasoning was implemented using Jena. Korpipää et al. (2003) also presented a work on developing a lightweight ontology for mobile device context-awareness. But their context structure is represented using RDF, and this ontology contains only concepts, properties, and concept taxonomies, but no constraints. Our system has the advantage that the system can produce real-time deduced packets, and inform the robot swarm to perform them.

## 8 CONCLUSIONS

This paper presented an approach for semantic support of resource constrained robots in intelligent applications. We described our EN-based communicative architecture, which can complete the loop between robots and the swarm server by EN packets. We proposed EN formats, context

information ontology model, and a reasoner based on context information. We presented experiments and some promising results using a simulator.

The EN-based communicative architecture can be utilized for very resource-limited robots. It enables semantic functionality for the swarm of robots. The EN allows very simple and short packets to be sent by the robots; on the other hand these packets can be transformed into advanced knowledge representations in an unambiguous fashion. For example, when the robot does not even contain any OS but all functionality is programmed directly using C, it is straightforward to use UUID values as constants that are used to identify received packets and placed at the beginning of the sent packets.

The swarm server can transform EN packets from the robots into RDF format, use the information to deduce new tasks relevant to context information, and adapt the operation of the swarm to perform given actions efficiently. The context information ontology model and inference mechanism of swarm server provide semantic support for the swarm robots.

The simulation result shows that the length of short packets is less than 10% of the corresponding RDF document's length. These results are compared with other lightweight representation. Furthermore, when short EN packets are used, only composer and decomposer are needed for pre-process. A resource-constrained device does not need to run any complex algorithm to process the packet.

The future work includes building the first prototype where real robots use our EN-based communication. We will consider the uncertainty and dynamics in a real robot swarm system. EN can also be used in robot-to-robot communication in order to minimize bandwidth and computational overhead. Finally, we will develop the lightweight communication framework further based on this work, for example, to make more complex data structure be transferred to EN possible.

## ACKNOWLEDGEMENTS

This work was funded by Infotech Oulu. The author would like to thank the participants of ROBOSWARM project.

## REFERENCES

Boley, H & Chang, E 2007. Digital Ecosystems:

- Principles and Semantics, in *Proceeding of IEEE International Conference on Digital Ecosystems and Technologies*. IEEE Computer Society Press, Cairns, pp. 398-403.
- Chen, H, Finin, T, Anupam, J, Kagal, L, Perich F & Dipanjan, C 2004, 'Intelligent Agents meet the Semantic Web in Smart Spaces', *Internet Computing*, vol. 8, no. 6, pp. 1545-1651.
- Cheney, J 2000. Compressing XML with Multiplexed Hierarchical PPM Models, in *Proceedings of the IEEE Data Compression Conference*. IEEE Computer Society Press, Snowbird, pp.163-172.
- Dey, A & Abowd, G 2000. Towards a Better Understanding of Context and Context-Awareness, in *Workshop the what, who, where, when, and how of context-awareness at CHI 2000*. ACM Press.
- Friedman-Hill, E 2008, Jess, the rule engine for the javaTM platform, viewed by 24 April 2009, <<http://herzberg.ca.sandia.gov/jess/index.shtml>>.
- Gzip.org 2003, *The gzip home page*, viewed 24 April 2009, <<http://www.gzip.org/>>.
- Holland, O & Melhuish, C 1999. 'Stimergy, Self-Organization, and Sorting in Collective Robots', *Artificial Life*, vol.5, no. 2, pp. 375-393.
- Json.org 1999, *Introducing JSON*, viewed 24 April 2009, <<http://www.json.org/>>.
- Korpiää, P, Mantyjärvi, J, Kela, J, Keranen, H & Malm, EJ 2003. 'Managing Context Information in Mobile Devices', *Pervasive Computing*. vol.2, no. 3, pp. 42-51.
- Riekkä, J, Alakärppä, I, Koukkula, R, Angeria, J, Brockman, M & Saloranta, T 2007, *Wireless Pain Monitoring, in The 2nd International Symposium on Medical Information and Communication Technology*. University of OULU Press.
- Riekkä, J, Su, X & Haverinen, J 2008. Connecting Resource-Constrained Robots to Knowledge-Based System, in *Proceeding of International Conference on Modelling, Identification and Control*. ACTA Press.
- Studer, R, Benjamins, VR & Fensel, D 1998. 'Knowledge engineering: principles and methods', *Data Knowledge Engineering*, vol.25, no.1-2, pp. 161-198.
- Wang, X, Dong, JS, Chin, CY, Hettiarachchi, SR & Zhang, D 2004, 'Semantic Space: An Infrastructure for Smart Spaces', *Pervasive Computing*, vol.3, no. 3, pp. 32-39.
- W3C.org 2004, *RDF Primer*, viewed 24 April 2009, <<http://www.w3.org/TR/REC-rdf-syntax/>>.
- W3C.org 2008, *Efficient XML Interchange (EXI) Format 1.0*, viewed 24 April 2009, <http://www.w3.org/TR/2008/WD-exi-20080919/>.
- W3C.org 1999, *WAP Binary XML Content Format*, viewed 24 April 2009, <<http://www.w3.org/TR/wbxml/>>.
- W3C.org 2004, *XML Information Set (Second Edition)*, viewed by 24 April 2009, <<http://www.w3.org/TR/xml-info/>>.
- XMLSolutions 1999, *XMLZip - XML Solutions*, viewed 07 June 2008, <<http://www.xmls.com/>>.
- Yaml.org 2008, *YAML Ain't Markup Language Version 1.2*, viewed 24 April 2009, <<http://yaml.org/spec/1.2/>>.