Implementing the Draft W3C Semantic Sensor Network Ontology

Daniel O'Byrne, Rob Brennan, Declan O'Sullivan
Knowledge and Data Engineering Group, Trinity College Dublin, Ireland
{dobyrne, rob.brennan, declan.osullivan}@cs.tcd.ie

Abstract

This paper examines the feasibility of using ontologies to model generic sensor networks, based on the capabilities of the current generation of ontology tools. The creation of such an ontology, the current tool's capacity to adequately maintain it, and its potential functionality, as part of a larger semantic system are addressed here. These topics were addressed by constructing a generic sensor ontology and attempting to implement it as part of a larger semantic system. The process and the conclusions drawn from it are described below.

1. Introduction

The concept of a semantic sensor web [1] has recently been gaining attention in the web-based sensor research community. This envisions the merging of standard W3C semantic web technologies such as meta-data definitions encoded as RDF (resource description framework) documents and XML-based sensor descriptions such as SensorML (to describe sensor deployments and capabilities) developed by the Open Geospatial Consortium[1] (OGC). This means adding formal semantic annotations to existing standard sensor models in order to provide semantic descriptions and enhanced access to sensor data. This is accomplished with model-references to ontology concepts that provide more expressive concept descriptions and relationships between concepts. It is hoped that the use of formal meta-data for describing the sensors' outputs, platforms, locations, and control parameters will enable a new generation of flexible sensor-based applications.

To support this activity the W3C setup the Semantic Sensor Network Incubator Group[2] in March 2009. The three main activities of this group are to develop an ontology-based framework for describing sensors, mappings between these ontologies and standard sensor models and finally specification of semantic annotation for the sensor markup language.

[1] http://www.opengeospatial.org/projects/groups/sensorweb
[2] http://www.w3.org/2005/Incubator/ssn/

In this paper a draft of the W3C sensor ontology was analyzed and specialized to apply to a semantic sensor deployment case study based on management of smart buildings. A prototype was built with current semantic web tools. We describe the system and lessons learned, especially with respect to current tool support.

In this paper we first describe related work and then the case study that forms the context of our work. We go on to describe the architecture of our prototype and the specialized version of the draft W3C sensor network ontology we developed. We then give a description of our practical experiences with semantic sensor web tools and their impact on our design and implementation and finish with some conclusions.

2. Related Work

In recent years, sensors have become both more affordable and widely available and hence have been deployed in diverse application areas. There are already a number of established international standards for describing sensors and their data that have been produced by the OGC Sensor Web Enablement (SWE) activities. Many see these standards as limiting since, although they do provide a level of syntactic interoperability, they do not provide support for explicitly encoded semantics and thus computer –based logic or reasoning. Ontology-based representations are widely viewed as the solution to this perceived weakness as they provide additional functionality.

The OGC SWE standards are based on XML and due to the continued push, both from academia and industry, towards Tim Berners-Lee's vision of the semantic web, there exists large body of research focused of translating XML based information into OWL (Web Ontology language) – based Ontologies [2][3]. The OGC SWE has already produced and adopted a number of XML-based standards and the ability to link these existing standards to a new ontology-based sensor standard is highly desirable to ensure rapid deployment and backwards compatibility.

A number of diverse sensor related ontologies have already been created, such as the Wireless Sensor Networks Ontlogy (WISNO) [4] and the Ontonym Ontologies [5]. The WISNO is a two-tier ontology framework that, while easily scalable in terms of the size of the sensor network, is very scenario specific and abstracted from the low level sensor data. The Ontonym Ontologies by comparison are much broader in scope and less abstracted from the physical sensor data, but as a result requires the ontology to handle far more data as the sensor network grows.

Another prominent instance is the OntoSensor ontology [6], built using the SUMO upper ontology [7]. This is a sensor knowledge repository design based upon SensorML (Sensor Model Language) – one of the OGC SWE XML-based standards. OntoSensor builds upon the SensorML framework to provide some advanced functionality like SPARQL querying support and it has been suggested that it may serve as a component in a more comprehensive application that includes more advanced inference mechanisms (such as description logic-based reasoning). OntoSensor, like most of the other emerging sensor based ontologies, is restricted in two key ways; the design was based mainly around the OGC SWE SensorML standard rather than sound ontology principles such as having many inter-related properties, and it lacks a certain degree of inference and data interdependency that are among the key strengths that make ontologies so desirable.

These limitations in the emerging sensor ontologies have led to a call for more generic sensor ontologies. This led the W3C to launch the Semantic Sensor Network Group to begin the formal process of producing ontologies that define the capabilities of sensors and sensor networks. After reviewing a number of the currently available sensor ontologies, the incubator group has adopted the use of the 'semantic sensor network ontology' proposed by Holger Neuhaus and Michael Compton [8] as their main baseline.

While the idea of developing and using an ontology to describe sensors seems theoretically sound and if successful should deliver advantages over competing XML based sensor schemas, there is an implicit assumption that the current development tools for ontology-based applications are suited for developing and utilizing such an ontology. This paper examines this assumption by developing a working application utilizing a generic sensor ontology based on the drafts of the W3C Semantic Sensor Network Group.

3. Case Study

In order to place realistic requirements on our semantic-sensor-based application, this work was carried out in the context of a larger program of research focused on sensor enhanced Facilities Management (FM) systems for smart buildings.

These systems rely on pervasive sensor networks collecting environmental monitoring data and storing it in a centralized data warehouse with parallels to traditional telecommunications operations, administration and maintenance systems. In addition to environmental monitoring data, location tracking systems, such as Ubisense, were deployed to track buildings maintenance personnel and to support maintenance task scheduling activities. This provided a diverse set of sensor data to which semantic meta-data was added in order to facilitate flexible context-aware applications based on sensor data correlation and re-use. In addition to the sensor data, ontologies are used to capture the building descriptions including the essential properties relevant to FM.

Traditionally such context-aware applications have been very expensive to build and test so within the project there is a dedicated smart building simulation tool set development team [9]. This simulation toolset, SimCon, had previously been built to use sensorML models so the task of the current work was to adapt that to use ontology-based models based on the W3C draft. It is hoped that this will enable new context-aware applications that consume the semantic sensor data as a dynamic data warehouse. The first use case to be supported was for a context-aware building maintenance personnel support application that queried sensor data to display a map of the current surroundings for an engineer overlaid with sensor readings for the sensors in the immediate vicinity. The location-based sensor data was generated by the SimCon toolset and mimics real world data generated by the Ubisense real time location system. This, for example, aids the engineer with building fault diagnosis tasks as environmental monitoring data can be used both to detect and isolate building faults to specific building sub-systems. Ontology-reasoning can be used to perform root cause analysis for faults derived from a stream of sensor events once facilities management domain expert knowledge has been captured in an ontology.

4. Architecture

In this section we describe the requirements for the semantic sensor web-based system, the components deployed and the dynamic behavior of the system.

4.1. Requirements

4.1.1. Sensor Data. Access to appropriate **s**ensor data is fundamental to the target application. Simulation techniques for sensor networks have matured considerably in recent years and simulations are oftenmore cost effective than deploying real sensors. Our prototype system utilizes emulated sensor data rather than actual sensor data. This sensor data was generated in a 3D environment,with embedded sensors, constructed using PUDECAS [9] and SimCon (simulated context) [10] tools. Thus the VR environment can produce realistic sensor data (including noise, loss, inaccuracies, etc.) which is stored in an XML database.

4.1.2. A Sensor ontology. To enable semantic applications to process the sensor data, a sensor and sensor data ontology was needed. It was decided to use the generic sensor ontology recently proposed to the W3C [8] as a base for the system's ontology. This system would only be used with a small number of known sensors, so it was possible to simplify the generic W3C ontology slightly in order to expedite the system's implementation. Simplifying the ontology enabled the system to be built and tested far quicker than it could be otherwise while maintaining enough detail to examine the current generation of ontology tools.

4.1.3. SPARQL. The system would also require a way for context-aware semantic applications to access the data. It was decided that SPARQL, an RDF query language and W3C recommendation, would be used for this. This would enable access to any part of the sensor ontology, in a variety of formats, once an appropriate SPARQL endpoint had been added to the system.

4.2 Component Descriptions

Figure 1 illustrates the main components of the final system. Each is described in more detail in the sub-sections below.

4.2.1. SimCon. Additional functionality needed to be added to SimCon since the original design was based around sensorML rather than an ontology-based approach. SimCon converts Sensor data collected from the VR simulation environment into custom internal implementation classes, so the sensor data needed to be extracted from these classes and used to populate the sensor ontology. Simcon is written in Java, so it was possible to make the additions using the Jena toolkit[3].

4.2.2. Jena. Jena is an open source Java framework for building Semantic Web applications developed by HP Labs. It includes, among other functionality, a RDF manipulation API, an OWL API and the ability to read and write RDF in the RDF/XML, N3 and N-Triples formats. Jena was chosen when extending Simcon due to this functionality, and the fact that it has an active development community.

4.2.3. Database. In order to properly test and evaluate our system, a persistent store was required for the ontology-based sensor data. This requirement led to the inclusion of SDB (SPARQL Data Base), a component of Jena that supports RDF storage and querying. SDB is transactional. The storage is provided by an SQL database and an SDB store can be accessed and managed via command line scripts and the Jena API.
For the SQL database, MySQL[4] was chosen. It is a relational database management system and is one of the SQL databases supported by SDB.

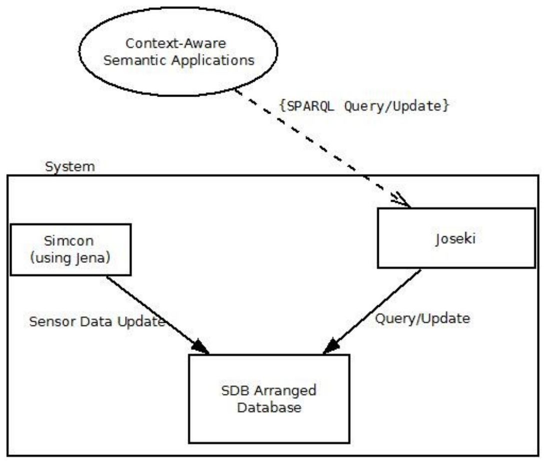

Figure 1. System Overview

4.2.4 Joseki. The System's sensor information needed to be accessible to applications in real time and for this a SPARQL endpoint was used. Joseki is a HTTP engine that supports the SPARQL Protocol and the SPARQL RDF query language. It was developed by HP Labs and can be used to give a SPARQL interface to an SDB installation. With Joseki, it was possible to access all of the sensor information in real time as it is updated by the simcon simulation. Joseki also has the

[3] http://jena.sourceforge.net/
[4] http://www.mysql.com/

ability to be connected to multiple separate datastores and supports the proposed SPARQL/Update standard.

4.3. Dynamic Behavior

Figure 2 illustrates the system's dynamic behavior: (1) simulated sensor data is generated in Simcon, (2) it is published to the database via the Jena SDB API, (3) in parallel Joseki can handle SPARQL-based queries for the data from context aware applications.

Simcon regularly generates new sensor readings (at a configurable interval, e.g. 5s) and this new data is periodically used to update the system's database. One advantage of the Jena SDB interface is that it handles all interactions with the database as transactions, thus enabling the database to handle potential scheduling and locking issues between Simcon and Joseki.

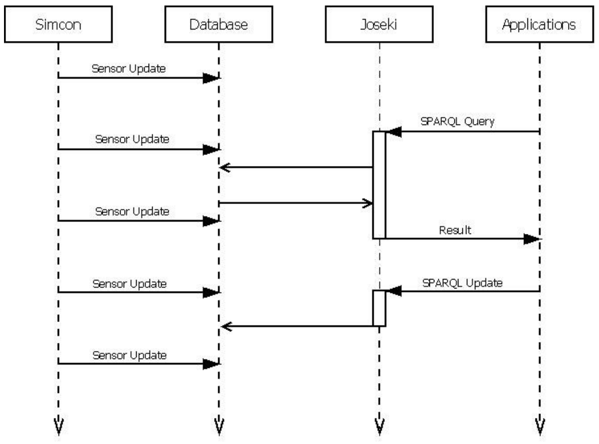

Figure 2. Dynamic Behavior.

Applications may query Joseki over a HTTP connection and Joseki has its own locking policy in place to handle multiple and simultaneous queries. Once it receives a query, Joseki in turn accesses the Database and retrieves the most current version of the sensor ontology. Joseki performs the SPARQL query over the ontology and can then return the result to the application in a variety of formats.

5. A Specialised W3C Sensor Ontology

The ontology used in the system was based on one proposed to the W3C Semantic Sensor Network Incubator Group. It is described below (see fig. 3) along with its main differences with that ontology.

5.1. Sensor Ontology Elements

Each element used in the ontology is described below:

Sensor. The Sensor class is present mostly as an abstract hub to link all the major facets of a sensor together. These facets are the Sensor Grounding (The physical details of the Sensor), The Operation Model (The sensor data and information about the sensor data) and what type of sensor it is you are dealing with.
Sensor Grounding. The SensorGrounding class contains all the information about the sensor's physical reality, such as its location and its source of power. It also contains the Sensor's Serial ID (which for the purpose of the system must be unique).
Type Of Sensor. The typeOfSensor class contains information on what type of sensor is being described. The system was tested using mostly simulated Ubisense (tag based positional tracking) and Zigbee(low rate, short range wireless) sensor data.
Operation Model. The OperationModel class, like the sensor class, is an abstract hub – in this instance, providing links to the ModelResult and ResponseModel classes.
Model Result. The ModelResult class contains information about the data the sensor retrieves. It effectively contains meta data for the sensors data. Information such as a sensor's accuracy and resolution are essential for a complex application to usefully decipher its readings.
Response Model. The Response Model contains a sensor's actual readings.

5.2. Differences with W3C Draft Ontology

In the course of our work we made several simplifying assumptions compared to the W3C draft sensor ontology and these are described here.

5.2.1. Less Advanced Datatypes. The diagrams depicting the proposed Generic Sensor Ontology for the W3C indicated the wide use of datatype properties. When implementing the System, it was decided that the considerable overhead of using such properties was unnecessary and in many cases detrimental to the systems implementation, as the current generation of ontology tools struggled to properly parse and handle them. For example, when using the OWL 'float' datatype, neither Jena nor Joseki could parse and return just a value and instead returned the entire OWL statement including both the value and its type.

5.2.2. Fewer Classes. The W3C ontology contains far more classes and properties than the one constructed for the system. This stems from the aim of the W3C ontology to be generic and thus provide support for as many sensor types as feasible. Since the type of sensors being used and the environment of the system was not only known in advance, but controllable through the use of SimCon and PUDECAS, it was decided to simplify the process of constructing and interacting with the ontology by removing the extraneous classes. The W3C process classes were also removed from the ontology as the system would not be using them. These classes describe how a sensor operates by modeling concepts such as inputs and outputs and how sensors connect to each other.

In summary our system was more specialized and less expressive than the W3C draft proposal but the essential patterns of the draft are retained in our design.

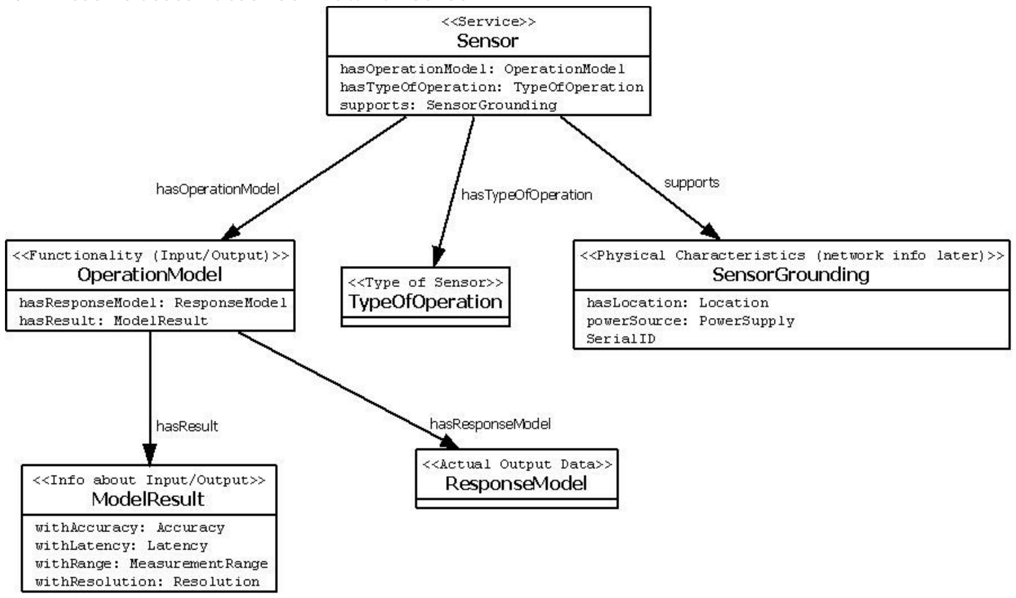

Figure 3. Generic Sensor Ontology

6. Practical Experiences with Ontology Application Development Tools

One of the first steps in constructing the system was sensor ontology design. Many tools exist to simplify the construction of ontologies and Protégé [11] was chosen as it is a free, open source ontology editor and it has a large user community. Constructing the initial ontology with Protégé was an easy task once the design was complete; however problems occurred later, when the time came to interface the ontology with SimCon.

Protégé provides advanced OWL functionality to an ontology with a mouse click, but Jena is not so user friendly. Each OWL feature added in Protégé quickly became lines of additional code when using Jena. The system needed to add new instances to the ontology for each sensor it discovered, and with the initial version of the sensor ontology, this translated into hundreds of lines of code. First the relevant classes needed to be initialized in Jena, then their attributes and properties. This painstaking process prompted the revision and simplification of the sensor ontology being used.

Another issue that arose when attempting to interface the ontology with Simcon was the seemingly unavoidable hard-coding of Class names. Jena converts the entire ontology, including instances, into a model which it can then navigate. The only effective way to navigate over an ontology model is to query it for particular triples, but this requires you to know the subject, object or property in a given triple. This effectively means that the system is completely restricted to the ontology initially chosen for it and coding performed in Jena is not reusable with another ontology or if the initial ontology changes.

Navigating the ontology to update instances was the largest delay encountered while constructing the system. With Jena, instances aren't updated so much as they are systematically deleted and replaced. First the instance must be located (using each sensor's unique Serial Id in this case), then the ontology model must be traversed from that point, deleting and reconstructing all Nodes with the updated data. This entire process must be hardcoded with advanced knowledge of the ontology's structure and, depending on the ontology's structure, can constitute a large amount of code.

It was initially hoped to use Joseki as both a SPARQL endpoint and storage solution, however it quickly became apparent that it was unsuited as a storage solution on its own. While Joseki is capable of storing an ontology file and loading it upon start up, it doesn't check the file for changes once it has initially been loaded. It also refuses to write through any updates made using SPARQL/Update, instead updating only the copy it stores in its working memory. It was this limitation which prompted the addition of a MySQL database to the System using SDB.

7. Conclusion

7.1. Sensor Ontologies. While a generic sensor ontology may be a viable way to model sensors, the implementation of this system has raised a number of important points. Such an ontology should be as close to perfect as possible when finalized, because the current tools require a high degree of ontology specific hard-coding in almost any application utilizing it.

Also worth noting is that while the layout of the ontology proposed is unusual due to its lack of inheritance, such a ‘flat’ structure seems to work well as far as implementation goes. Due to the need to add, remove and update sensor instances from such an ontology, a generic sensor ontology would seem to have little capability to be reasoned.

7.2. Updating. The main problem with using an ontology based solution for sensor networks is updating. Throughout the construction of this system, it has become obvious that the current generation of ontology tools is neither geared towards, nor capable of handling, ontology updates. There seems to be an assumption on behalf of the semantic web community that the knowledge in ontologies will remain static and not need to be changed. Examples of this are not hard to find; SPARQL is seen as the main way to interact with OWL based ontologies, yet the SPARQL/Update recommendation has not yet been ratified by the W3C. Out of Jena’s 40 pages of documentation, Instances make up less than half a page and updating them isn’t mentioned.

As Jena is an important tool for ontology manipulation, its method for updating instance data is especially worrying. Each time the system needs to update a sensor, it reads in the entire ontology, including all instances, from the database. Jena then wraps it all into an ‘ontModel’ container so that it can search through it. The instance to be updated must then be found using some unique identifier and then the program must traverse from that identifier to the node it wishes to change. This entire process is very resource intensive, especially as the number of instances grows.

The sensor networks envisioned for the future are seen as being comprised of thousands of sensors and this is one of the reasons a generic sensor ontology and context-aware semantic applications are so appealing. However, it is this scale which could also make such networks impossible with the current tools. Sensors return a frequent and regular stream of data and the current ontology updating methods would most likely be incapable of keeping up with it.

8. Acknowledgement

This work is supported by the Irish Government under the “Network Embedded Systems” project (NEMBES) in the Higher Education Authority's Programme for Research in Third Level Institutions.

9. References

[1 A. Sheth, C. Henson, S. S. Sahoo, "Semantic Sensor Web," *IEEE Internet Computing, vol. 12, no. 4, pp. 78-83, July/Aug. 2008*

[2] H. Bohring, and S. Auer, “Mapping XML to OWL ontologies”, *Leipziger Informatik-Tage vol. 72,* GI, 2005, pp. 147-156.

[3] T. Rodrigues, P. Rosa, and J. Cardoso, “Mapping XML to existing OWL ontologies”, *Internatioal conference WWW/Internet 2006*, pp. 72-77.

[4] Y. Hu, Z. Wu, and M. Guo, “Ontology driven adaptive data processing in wireless sensor networks”, *Proc. 2nd international conf. on Scalable information systems*, 2007.

[5] G. Stevenson, S. Knox, S. Dobson, and P. Nixon, “Ontonym: a collection of upper ontologies for developing pervasive systems”, *Proc. 1st Workshop on Context, Information and Ontologies*, ACM New York, 2009.

[6] D.J. Russomanno, C. Kothari, and O. Thomas, “Building a sensor ontology: A practical approach leveraging ISO and OGC models”, *The 2005 International Conference on Artificial Intelligence*, 2005.

[7] I. Niles and A. Pease, “Towards a standard upper ontology”, *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, ACM New York, 2001, pp. 2-9.

[8] H. Neuhaus, M. Compton, “The Semantic Sensor Network Ontology: A Generic Language to Describe Sensor Assets”, *http://www.w3.org/2005/Incubator/ssn/wiki/images/2/2e/SemanticSensorNetworkOntology.pdf*, last visited on 27th September 2009.

[9] E. O'Neill, D. Lewis, and O. Conlan, “A Simulationbased Approach to Highly Iterative Prototyping of Ubiquitous Computing System”, *SIMUTools 2009: Proceedings of 2nd International Conference on Simulation Tools and Techniques*, Rome, Italy, 2-6 March, 2009.

[10] K. McGlinn, E. O’Neill, D. Lewis, “Modelling of Context and Context Aware Services for Simulator Based Evaluation”, *MUCS 2007 - 4th International Workshop on Managing Ubiquitous Communications and Services*, Ireland.

[11] H. Knublauch, R.W. Fergerson, N.F. Noy and M.A. Musen, “The Protege Owl Plugin: a open development

environment for semantic web applications", *Lecture Notes in Computer Science*, Springer, 2004, pp. 229-243