Extending Semantic Sensor Networks with QueryML

Keyi Zhang and Alan Marchiori
Department of Computer Science
Bucknell University
Lewisburg, PA
Email: {kz005, amm042}@bucknell.edu

***Abstract*—As sensors become more affordable and versatile, more and more sensors are deployed in different environments to help people observe their surroundings. However, due to their various physical structures, it is very challenging to have a universal schema to identify, search, and query sensors and sensors' data. Fortunately, there are two main approaches to address some of these problems, namely Semantic Sensor Network (SSN) from W3C and Sensor Web Enablement (SWE) from the Open Geospatial Consortium (OSG). Both utilize XML to extend sensors' metadata and let machines understand the semantic meaning of a sensor. However, even though they provide a universal way to describe and deliver high-level sensor information, neither enable the querying of historical data. In this paper, we briefly examine the current semantic sensor web developments, SNN and SWE along with their advantages and challenges. Then we present our extensions to enable querying historical data within the semantic sensor domain that we call QueryML. QueryML can be used to extend the capabilities of either SSN or SWE to support querying historical data.**

I. Introduction

An ever increasing number of networked sensors are used in various application areas, such as environmental monitoring, military, health care, and building performance monitoring. However, sensor networks have their own problems that have not been solved entirely. Different sensor networks have different application scales, manufacturers, communication protocols, and application architectures. As a result, sensor data can normally only be processed and consumed by their own custom applications. This makes it tedious to develop applications able to synthesize data from different sensor networks.

Fortunately, there are two semantic approaches that begin to address this problem, namely, Semantic Sensor Network (SSN) from the World Wide Web Consortium (W3C) and Sensor Web Enablement (SWE) from the Open Geospatial Consortium (OGC). Although both of them adopt XML as their main markup language, their philosophies are very different. SSN borrows ideas from Linked Data [1] and focuses on linking sensor networks using semantic descriptions of sensor data. On the other hand, SWE uses syntactic interoperability to discover sensor and transform sensor observations and provides more well-defined specifications for sensors [2].

Both frameworks offer the ability to query current sensor data values. However neither approach can query historical data. The philosophy behind this is that historical data are mostly aggregated and then are stored in a database; therefore the system should focus only on providing current data [3]. However, in many cases, the need to query history data rises frequently. For instance, environmental scientists need a wide range of historical data to understand the current weather trends. Nevertheless, there are several difficulties existing for solving this problem. Because different sensor networks may have vastly different system architectures, a generic data query specification should be developed.

The new specification should also give sensors semantic meaning and able to be integrated into either of SNN or SWE easily. In this paper we propose a new XML based language which gives sensors semantic query metadata, which gives applications instructions about how to query the history data. The new XML schema is designed upon SensorML, which is commonly used in SWE, thus it can be used directly in SWE sensor networks.

The structure of the paper is as follows: we will summarize the SNN and SWE along with SensorML, in Section II and Section III. In Section IV we will present the new XML based schema, QueryML, that allows applications to query data according to the preconfigured metadata stored in the sensor or sensor mapping server.

II. Semantic Sensor Network

As envisioned by the inventor of World Wide Web, Sir Tim Berners-Lee, the Internet needs a universal data navigation framework that allows different information resources to be linked together and thus be searched easily [4]. SSN borrows this idea and expands it into sensor networks [5]. It takes any level of details that relate to sensor, stimulus, and observations into a Web Ontology Language (OWL). Any application that wants to query the sensor or sensor observation data can use the Resource Description Framework (RDF) query language to query the linked data.

A. SNN Ontology

The SNN ontology is based on the central Stimulus-Sensor-Observation pattern, which consists of four different perspectives: sensor, data, system, and feature and property. Any change in the environment can be treated as stimulus, and any deice that transforms the incoming stimulus into digital form will be considered as the sensor. An observation is made when the data is interpreted.

The notion of a stimulus in SNN has broader meaning than the traditional one. Any change in the physical world that can be detected by a sensor will be treated as a proxy of property, i.e. stimuli. Thus, it cannot be used to describe abstract information such as sets and regions [5].

Sensors in the SNN ontology are not limited to physical devices; human beings can also be treated as sensors for some observable stimuli. A sensor must follow a certain method to measure the physical properties and describe how they observe.

Observations are social objects, i.e. it acts as the nexus between incoming stimuli, the sensor, and the output of the sensor [6]. In other words, observations are the interpretation of incoming stimulus transformed by sensors as it performs additional analysis and produce results based on the stimulus.

B. *OWL for Describing Sensors*

OWL is designed by W3C and is widely used by applications which adopt SSN as a sensing framework. It is based on RDF and is highly exchangeable with RDF. There are three subsets of OWL: OWL Lite, OWL DL, and OWL Full. Each of them give users different functionality and application areas.

An RDF statement has three components: a subject, a verb, and an object. The subject must be a resource in Uniform Resource Identifier (URI) form that can be uniquely identified. The verb is a predicate tells the relationship between the subject and object and it is also identified by a URI. An object is the target of the statement. It can be anything to represent the data, such as a floating point number, or a URI identifier linked to another RDF statement.

```
<rdf:RDF xmlns:rdf="http://www.w3.org
    /1999/02/22-rdf-syntax-ns#" xmlns:p="http
    ://www.example.org/person#">
  <rdf:Description rdf:about="http://www.
      example.org">
    <p:name>Bob</p:name>
    <p:age>25</p:age>
    <p:ID>1234</p:ID>
  </rdf:Description>
</rdf:RDF>
```

Fig. 1. RDF Example to Describe a Person

In Figure 1 we use RDF to define a person (p). The person's URI is given by the XML namespace, and each predicate are also defined as a URI, e.g. http://www.example.org/person/name. The object usually is the value of the XML element. For instance, Bob is the object in the name statement, and the entire statement can be translated into English as *the person's name is Bob*. Following the same rules, we can conclude that Bob's age is 25 and his ID is 1234.

OWL adds many schemata to describe the sensor and sensor observation semantically within a standard framework. It provides modular definitions such as *Object*, *Method*, and *InformationObject*. Although OWL itself cannot describe time and location, this information can be included via OWL imports [5]. More details can be found from the W3C [1]

Several query languages are available to query sensors and sensor data, such as SPARQL[2] and OWL-QL[3]. However, these are incomplete for our use as discussed below.

[1]http://www.w3.org/2005/Incubator/ssn/ssnx/ssn
[2]http://www.w3.org/TR/sparql11-query/
[3]http://ksl.stanford.edu/projects/owl-ql/

C. *Challenges yet to Solve*

Because different sensor networks are deployed in different domains, with highly varied configurations, most sensor network applications are domain-specific. In addition, because linked data allows user-defined verbs (predicates) in the statement, it is challenging to perform data fusion and aggregation across different sensor networks [7].

Although SPARQL is the widely used linked data query language that can be applied to semantic sensor network, it is verbose to apply it to OWL based SNN because its original design is not for OWL. Although some well adopted query languages have been developed to help rapid application development, the performance is still limited [8]. Therefore a new language designed to query data, like QueryML, should be developed.

Most importantly, providing a semantic streaming data query method to query sensor data is the still main task for SNN. Although several approaches have been developed as mentioned in [7], methods to ensure data quality and perform heterogeneous data stream fusion is still an open question.

III. Sensor Web Enablement (SWE)

SWE is developed by OGC to enable the Web-based discovery, exchange, and processing of sensor observations, as well as the tasking of sensor systems [9]. SWE also prevents technology lock in from any specific application and sensor manufacture technology, making the system easier to expand in the future.

A. *Observations and Measurements (O&M)*

O&M is the core service in SWE. It gives specifications to many aspects of the system. First, it defines a model describing sensor observations as an act of observing a certain phenomenon. Second, it defines the observation types to those sensors whose observations represent spatial relationships. In this way, the semantic web concepts are introduced to this service as it can link different sensor observations together, though only in a limited way [10].

B. *SensorML*

The OpenGIS Sensor Model Language (SensorML) provides an information model and encoding to describe sensors for discovery as well as sensor observations. Each observation is exposed as a process in SensorML, which can be divided into two different types: physical process and non-physical process, such as merely mathematical operations [11].

A process in SensorML is either a physical or computational operation that may receive inputs, and based on configurable parameters and methodology, generate outputs [11]. Thus, SensorML uses processes to model sensors and detectors which convert observable phenomenon into digital data.

The intention behind sensor discovery is to support plug and play operation when adding a new sensor to the system. Assume that a sensor has a SensorML description, when plugged into the sensor network, the Sensor Observation Service (SOS) discovers the sensor and requests the SensorML

```
<sml:outputs>
  <sml:OutputList>
    <sml:output name="temp">
      <swe:Quantity definition="http://sweet.jpl
          .nasa.gov/2.2/quanTemperature.owl#
          Temperature">
        <swe:label>Air Temperature</swe:label>
        <swe:uom code="Cel"/>
      </swe:Quantity>
    </sml:output>
  </sml:OutputList>
</sml:outputs>
```

Fig. 2. SensorML Example (From http://www.sensorml.com/sensorML-2.0/examples/)

description. The registration service is then informed about the new sensor and the sensor is made discoverable by other nodes.

In the Figure 2, SensorML defines an air temperature sensor with Celsius as its measurement unit. An online resolvable dictionary is assigned to the *Quantity* so that applications can get very specific information about what is being measured. This ensures interoperability between various sensor communities.

C. More Descriptive, Less Semantic

The entire SWE is a systematic solution for sensor description, discovery, and obtaining measurements. Currently SWE, covers many types of sensor systems as weather stations, air pollution, and satellite imaging systems.

However, as mentioned in the SNN final report [6], SWE does not provide semantic interoperability and does not provide a basis for reasoning that can ease development of advanced applications. Because all of the relationships between sensors are predefined, without an ontology, the amount of logical information provided by SWE is limited [12].

IV. QUERYML: ENABLING HISTORICAL DATA QUERY

It is undeniable that streaming real time sensor data is important for monitoring the environment and events. However, historical data can offer additional insights that real time data cannot provide. For instance, in terms of weather monitoring, it is helpful to query historical data to analyze specific patterns to improve predictions in real-time. Some researches have already made attempts to solve this problem, such as [13]. Nevertheless, none of them have combined semantic sensor concepts with historical data query abilities. Here we present our design, QueryML, to gracefully handle the querying challenge within the semantic sensor domain.

A. Structure

QueryML is a semantic description language based on XML. It can be easily transformed into SNN or SWE. It is designed for sensor networks where historical sensor data can be acquired through a RESTful web interface. Each QueryML description for a single sensor node has three components: data source web interface, input data transformation, and output data extraction. All of the descriptions can be treated as supplemental information to SNN or SWE; therefore they can be part of the SNN ontology or the SensorML statements. By integrating into either the SNN or SensorML, we can extend the core semantic features of either framework with cross platform historical query capabilities.

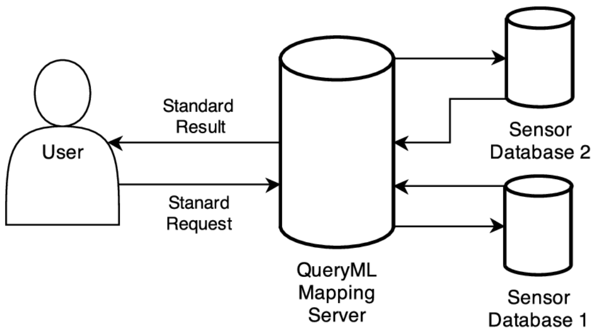

Fig. 3. Mapping Server between the Applications and the Data Sources

1) Data Source Web Interface: A data source URI is the main web interface providing the sensor data. A RESTful based interface hides the low level implementation details such as how the query is handled by the underlying database (e.g., SQL). This RESTful interfaces provides a layer of abstraction above the low-level database that are supported by most modern systems.

2) Input Transformation: Different sensor networks accept different query request formats. In order to hide the query details from user applications, an input data transformation is required. Necessary information for the query, such as the timespan and sensor identifier, is defined by the sensor in it's QueryML description. An application can then construct a valid query using the QueryML description to format the specific request details.

3) Output Data Extraction: As with the database inputs, the outputs are also highly varied in existing sensor systems. Thus, a standard transformation is necessary to operate on the outputs to normalize the results from different sensor networks. We use the QueryML output description to define the extraction of time series data from the raw XML or JSON results provided by the RESTful interface.

B. Query Data Source

As shown in the Figure 3, in order to hide the difference between different sensor networks, the QueryML middleware layer performs the necessary data transformations. The input transformation is applied to the standard query to produce a specific sensor database request. The sensor database's result is then extracted using the output data specification and provided to the user using a standard representation.

Figure 4, defines a QueryML description whose data is stored on an OpenTSDB server. Each QueryML statement defines a query parameter added into the query string. For instance, *<qml:MetricName>* and *<qml:Metric>* together define the metric name and value in the query string. When requested to give historical data from two timestamps, 1412121600 and 1412208000, the system will generate a query string using the

```
<qml:Description>
  <qml:Iuput>
    <qml:url>http://host/api/query</qml:url>
    <qml:MetricName>m</qml:MetricName>
    <qml:Metric>sensor1.temp</qml:Metric>
    <qml:QueryMethod>GET</qml:QueryMethod>
    <qml:TimeSpanName>start-end</qml:
        TimeSpanName>
  ...
  </qml:Iuput>
  <qml:Output>
    <qml:OutputFormat>JSON</qml:OutputFormat>
    <qml:OutputMetricName>[]/metric</qml:
        OutputMetricName>
    <qml:OutputTimestampName property="Key"
        Type="Array">[]/dps</qml:
        OutputTimestampName>
    <qml:OutputValueName property="Value" Type=
        "Array">[]/dps</qml:OutputValueName>
  </qml:Output>
</qml:Description>
```

Fig. 4. Sample QueryML Description

QueryML description, http://host/api/query?m=sensor1.temp& start=1412121600&end=1412208000, which is recognized by the OpenTSDB server.

When data is returned, the system will extract the query data and send it back to the user. Data extraction is operated by accessing certain elements defined by the QueryML. Because the current implementation is based on regular expressions, we define several patterns to assit the data extraction. For example, in OpenTSDB, timestamps are nested inside the *dps* element, which is also nested inside an array, the system will recognize *[]*, iterate through the *dps* element in the list, and return the timestamp and value as key-value pairs.

The system is highly abstracted to applications and is ready to recognize different sensor networks. For instance, different networks may support different timestamp formats, such as ISO or Unix epoch. The system will convert the timestamps automatically based on QueryML description and output the converted timestamps to the application.

V. Conclusion and Future Work

In this paper we examined two major approaches using semantic sensor web concepts to describe sensors and sensor observations so that standard applications can understand the sensors in a semantic way. However, neither of them provide a way to query historical sensor data. Thus we propose QueryML to define the structure of a query in the same way SensorML defines the structure of a sensor, so that applications can query historical data from different sensor networks in a general way. It is a work in progress and we are still improving the schema so that it will support more database web interfaces, such as restSQL (http://restsql.org/). A working example can be found at https://github.com/Kuree/QueryML.

References

[1] P. Barnaghi, M. Presser, and K. Moessner, "Publishing linked sensor data," in *CEUR Workshop Proceedings: Proceedings of the 3rd International Workshop on Semantic Sensor Networks (SSN), Organised in conjunction with the International Semantic Web Conference*, vol. 668, 2010.

[2] K. Walter and E. Nash, "Coupling wireless sensor networks and the sensor observation servicebridging the interoperability gap," in *Proceedings of 12th AGILE International Conference on Geographic Information Science*, 2009.

[3] O. Corcho, J.-P. Calbimonte, H. Jeung, and K. Aberer, "Enabling query technologies for the semantic sensor web," *International Journal of Semantic Web Information Systems*, vol. 8, no. 1, pp. 43–63, Jan. 2012. [Online]. Available: http://dx.doi.org/10.4018/jswis.2012010103

[4] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far," *International Journal on Semantic Web and Information Systems*, vol. 5, no. 3, pp. 1–22, 2009.

[5] M. Compton, P. Barnaghi, L. Bermudez, R. GarcíA-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog *et al.*, "The SSN ontology of the W3C semantic sensor network incubator group," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 17, pp. 25–32, 2012.

[6] L. Lefort, C. Henson, K. Taylor, P. Barnaghi, M. Compton, O. Corcho, R. Garcia-Castro, J. Graybeal, A. Herzog, K. Janowicz *et al.*, "Semantic sensor network xg final report," *W3C Incubator Group Report*, vol. 28, 2011.

[7] O. Corcho and R. García-Castro, "Five challenges for the semantic sensor web," *Semantic Web*, vol. 1, no. 1, pp. 121–125, 2010.

[8] R. Fikes, P. Hayes, and I. Horrocks, "Owl-qla language for deductive query answering on the semantic web," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 2, no. 1, pp. 19–29, 2004.

[9] M. Botts, G. Percivall, C. Reed, and J. Davidson, "Ogc® sensor web enablement: Overview and high level architecture," in *GeoSensor networks*. Springer, 2008, pp. 175–190.

[10] A. Bröring, K. Janowicz, C. Stasch, and W. Kuhn, "Semantic challenges for sensor plug and play," in *Web and Wireless Geographical Information Systems*, ser. Lecture Notes in Computer Science, J. Carswell, A. Fotheringham, and G. McArdle, Eds. Springer Berlin Heidelberg, 2009, vol. 5886, pp. 72–86. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10601-9_6

[11] M. Botts and A. Robin, "OpenGIS sensor model language (SensorML) implementation specification," *OpenGIS Implementation Specification OGC*, pp. 07–000, 2007.

[12] D. J. Russomanno, C. R. Kothari, and O. A. Thomas, "Building a sensor ontology: A practical approach leveraging iso and ogc models." in *The 2005 International Conference on Artificial Intelligence*, 2005.

[13] J. Ledlie, C. Ng, and D. A. Holland, "Provenance-aware sensor data storage," in *Proceedings of the 21st International Conference on Data Engineering (ICDE)*, 2005.