SENSOR DATA STREAM PROTOCOL (SDSP)

Version: 0.3 (August 14, 2012)

Edited by: Edosoft Factory, S.L (Edosoft)

Project Data

Acronym: ISN

Name: Interoperable Sensor Networks

ITEA number: 09034

Consortium:

- Vrije Universiteit Brussel
- Freemind
- MTP
- Edosoft
- MAIS

Document data:

Doc name: Sensor Data Stream Protocol (SDSP)

Doc version: 0.3

Doc type:

Version	Date	Remarks
0.1	July 16, 2012	First Draft
0.2	July 27, 2012	Added System Architecture
0.3	August 14, 2012	Preliminary version

Contents

Figures

1. Purpose

This document contains the State-of-the-art, design and specification of Sensor Data Stream Protocol (SDSP) that will be developed in the scope of the Interoperable Sensor Networks (ISN) project. This document integrates the deliverable *State-of-the-art: communication, technology, interoperability,* which addressed the state-of-the-art study on communication protocols done in the first part of the project.

2. Introduction

Traditional sensor network that are predominately centralized in organization and also the adhoc networks need a method to circulate their results and republish integrated data for future analysis. In addition, these contributors of data will need a process to link to other data sources that are related to their own. Finally, a system needs to be crated so that interested parties can find the data that is being shared. In this manner, to support the good data practice of sharing and also enable the additional services needed for discovery and storage of sensing data, it is introduced the Sensor Data Stream Protocol (SDSP). SDSP is a XML based language for sensor data that enables the good data practices outlined above and also promotes the additional services needed for aggregation, reference, and discovery.

3. State of the art

With the aim to build an open system for sensor data exchange and sharing, there are many related projects that SDSP can use as reference. First, there are many languages that have been used in describing various aspects of a sensor network, but most lack all the required features that SDSP is based off. Also, there is a technology for the World Wide Web (WWW) that tries to provide a universal medium for information exchange by putting documents with computer-processable (Semantic Web). Finally, there exists middleware architecture for sensor networks that attempt to support interoperability. The problem with these middleware architectures is that often times tend to ignore the data exchange portion.

3.1. Sensor Schema Languages

3.1.1. SensorML

SensorML is a standard which has been developed by the Open Geospatial Consortium (OpenGIS) [2]. SensorML intends to describe a sensor system in great detail. With the aim to detail the geospatial properties of sensor, SensorML uses GML[3] which is a standard also from OpenGIS. For example, it includes the relative positions of sensors, detailed description of the sensors themselves, physical entities that the sensor can measure, accuracy the sensor can achieve, the manufacturer, etc. Additionally, SensorML includes basic functionalities to describe processes which handle and transform sensor data. They intend to develop interoperability interfaces and meta-data encodings that enable real time integration of heterogeneous sensor webs.

3.1.2. TinyML

Despite looking like a good language, SensorML has some major shortcomings. It provides a sensor-centric approach for sensor coordination and description and it is pretty heavy weight and very complex. TinyML [4] addresses these shortcomings and implements a lightweight version but still following the basic ideas of SensorML. On the other hand, TinyML is tailored to embedded sensor networks. The fundamental elements of TinyML are the sensor, the platform, and the sensor field. A sensor describes a specific sensor and its properties. A platform represents a physical system with an energy source, a processor and a radio device. Additionally, a platform has a collection of sensors. The sensors measure basic physical phenomena, like humidity, temperature, air pressure, etc. At the larger scale, a sensor field is a collection of platforms and represents a sensor network.

Another important advantage is the capability to define virtual devices. TinyML can define virtual devices, which can be generated in an ad-hoc fashion and group together different platforms or sensors. This allows the ability to easily query a subset of platforms and sensors by taking advantage of virtual devices. Each element in TinyML can have a query or set a flag to indicate a response / actuation of that component. This is a very simple mechanism to interact with a sensor network.

The advantages of TinyML are that it gives a universal interface to interact with a sensor network and it is very lightweight. Unfortunately, TinyML does not have any notion of mobility, i.e., for a sensor network where nodes move around, and there is no rigorous implementation available.

3.2. Newsfeeds

3.2.1. Really Simple Syndication (RSS) and Atom

They are XML formats for messages and other information that are updated in a fairly frequent fashion. Often times the documents that are written in RSS and Atom formats are referred to as newsfeeds or feeds. These syndication languages are normally used to describe new entries in web blogs, publishing of metadata regarding change in the content of an information source, and aggregating of information from various sources. Semantics for these syndication languages involve the use of a set of metadata that describe the overall feed, individual entries or items that have summary information regarding a certain topic, and finally a link to the actual content which is often located on the web [5], [6]. In the main, RSS and Atom are the best ways of exchanging information regarding a certain source based on a specific topic in Internet. But the system works well for syndication of information that leans toward news or blog type format schemes. Several researchers have generalized RSS to sensor data. Simple Sensor Syndication (SSS) places sensor data over RSS and has shown how users can act in response to these feeds [7].

3.2.2. Numerical Really Simple Syndication (NRSS)

Numeric Really Simple Syndication (NRSS) is based on the same principles of RSS and Atom but extends the necessary ideas for syndicating numerical data over the web. It is designed to describe such things as the price of stock quotes, the temperature of a pond, etc.[8] For providing a context to the actual data that is being transferred, NRSS includes a description field. In addition, the items in NRSS have metadata tags along with other components including units and time period of operation and it supports representation of data in format such as comma separated values (CSV) and Data Access Protocol (DAP). Despite NRSS has many of the elements desired for SDSP, does not provide a mechanism to get detailed information about a sensing system deployment in a standard fashion. Furthermore, it still uses the same semantics as RSS and thus has fields that relate to newsfeeds rather than generic sensing data.

3.3. Semantic Web

The Semantic Web solves the shortcoming of HTML by using descriptive technologies such as Resource Description Framework (RDF), Web Ontology Language (OWL), and XML to provide context for the information. RDF is a simple data model for referring to objects and is often expressed using XML. RDF provides concepts such as resources, classes, and properties to describe a particular data set through a schema language [9]. Often times, data can be organized in an object oriented manner where resources are grouped as a class. These classes can have distinct attributes associated with them. In addition, resources that are similar can inherit or become sub-classes of existing resources that have been already described. Overall, the Semantic Web seems to be an interesting language to represent data but the exact data model for a specific data set still needs to be defined. For instance, there could be several different representations for sensor data using RDF definitions. Thus, SDSP uses some of the concepts articulated by the Semantic Web but does not employ RDF directly.

3.4. Middleware Architectures

3.4.1. Atlantis Framework

The Atlantis Framework [10] is based on TinyML but addresses several of its shortcomings. The basic elements are the same: fields, platforms, and sensors. Additionally, the Atlantis Framework adds data handling abstractions, and a query field for more detailed queries. It makes further improvements by defining a field task object which can handle asynchronous data retrieval. For this purpose, it adds an additional data broker which handles the tasks, and specific broker behaviors to describe how to handle the task itself. The Atlantis Framework adds data filters and event subscription possibilities. On the downside, there is not a standard way to manage the sensor systems since a registry does not exist. But overall this is an evolution in making sensor systems more interoperable although many sensor data problems are neglected

3.4.2. Tiny Application Sensor Kit

The Tiny Application Sensor Kit (TASK) [11] was designed for use by end-users with minimal sensor network knowledge. TASK uses TinyDB[12] as a back-end running on nodes and is thus tailored towards sensor networks running on Mica2 mote type systems. Additionally, it can handle only one deployment at a time and needs to be reconfigured to be used for a different deployment, i.e., it can not handle multiple deployments simultaneously.

4. SDSP Schema Languaje

4.1. Purpose and Goals

The Sensor Data Stream Protocol (SDSP) schema language is focusing in promoting the sharing, storage, and exchange of sensor data between different components. The schema language aims to address such issues as describing the context of the actual sensor system deployment, providing a mechanism to actually exchange data, and finally provide a mechanism to enable discovery and ranking of sensor data sources.

4.2. Componentes

There are essentially three distinct components in the SDSP schema language: context, sensor and connection. First, there is the context section where information regarding the actual sensor system deployment is detailed. Attributes such as the project associated with the deployment, the sensing area of the system, description of the sensor system and the platform that is being used are included.

The second section details the actual sensors involved in the deployment and the actual data values. More metadata information is provided for each sensor as part of this section. Finally, there is the connection section used to link to other sensor data sources that are related to the one being described. Many of the elements have an additional link element associated with them. This was included so that some of the XML tags can be linked instead of being embedded. This additional link element was used whenever the specific tag will likely not change very often. Thus, a link to a location that contains this information would be useful and thus prevent needless transferring of static information every time data is transferred.

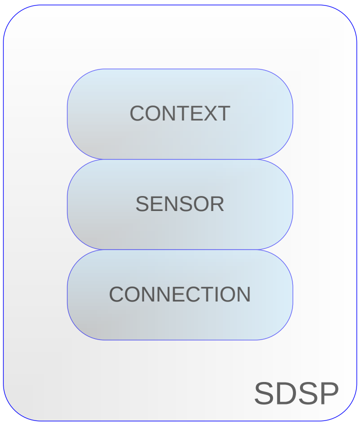

Figure 1: Main components of SDSP.

4.3. Sensor System Context

The sensor system context provides the needed metadata in order to get such details as the general sensing area, purpose, platform associated with the system, etc. The various tags are represented in Figure 2.

The project tag is a string element that gives a specific name to the overall sensor system. One can think of this as an administrative name given to the deployment. For instance, a potential project name could be "ISN Las Canteras Beach, Las Palmas de G.C Temperature System" to describe a sensor network that is used to monitor the temperature change in the beach area in Las Palmas de G.C.

The description element is a string that is a detailed annotation regarding the actual deployment. Information such as the specific purpose of the deployment, the individuals or group involved in maintaining the installation, and the duration that the sensing system will be used are potential details that could be included here.

The tag element, which is a string that is separated by commas, here is used for providing informal information regarding a certain project. While the description element serves as a through annotation, the tag is much more informal and should contain only short phrases or keywords. The tag element is very useful in helping users search for data since there is additional context associated with the project. For instance, if a deployment of sensor nodes occurred in Las Canteras Beach in Las Palmas de G.C in regards to measuring water pollution in the sea (such as levels of certain substances as Arsenic and Lead), then the project name might be “Las Canteras Beach Las Palmas de G.C Water Pollution” but the tag field can be used to provide additional details such as “Water Quality”, “Arsenic”, “Lead”, etc.

Another element in the context section is the location. The location is used to give a reference for the home location associated with a sensor system. It is considered the coverage area for the particular sensor system. The location is described either by a point or a polygon element. The point element contains a simple string of the format latitude, longitude, altitude and is derived from the GML standard [3]. The polygon is composed of a multi line string, where each line represents a corner of the polygon and is also derived from the GML standard. The format for each line is the same as the point element. Additionally, the polygon has to be closed, i.e., the first and last point has to coincide.

The platform element is used to represent the physical entity that is connected to a network and also the sensors themselves. Platforms can range from such things as motes to cluster heads that connect to a network of embedded sensors. For instance, "TelosB mote".

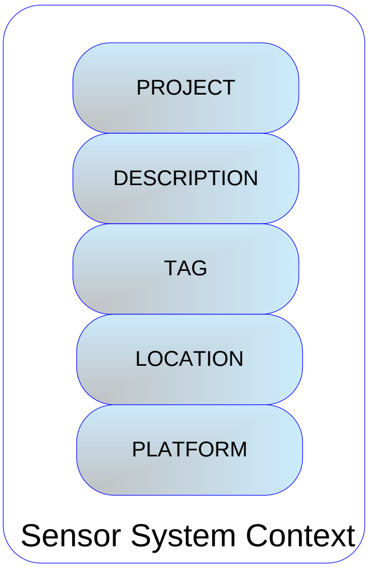

Figure 2: Context information for the SDSP

```
<context>
	<metadata>
		<project>ISN Las Canteras Beach, Las Palmas G.C Water Pollution
		</project>
		<description>Edosoft Factory, S.L</description>
		<tag>Water Quality, Arsenic, Lead</tag>
		<location>28.137034, -15.438173, 0.0</location>
		<platform>
			<description>Telos B</description>
			<type>mote</type>
		</platform>
	</metadata>
</context>
```

Figure 3: Detail of the context element of the SDSP

4.4. Sensor Data

The next major component of the SDSP schema is the sensor data section. Basically, the data is organized by the actual sensors that provide the measurement outputs. Thus each sensor has its individual section with specific context information and the data values associated with that sensor. In each sensor entry, these sensors would describe their context using the metadata element. Specific location information can be given at this point for each sensor.

The platform that the sensor actually exists on is detailed. Information about the platform including an identifier, description, and type can be provided. For instance, if a temperature sensor is attached to an embedded device such as a “Telos Mote” and it has a virtual identification number of ISN006 then these values can be expressed in the platform section of the sensor context. Furthermore, type information for the sensor is conveyed here as well. Example sensor types include: “VG400”, "DS5" or “Hidra Probe II Soil Moilsture and Salinity”. The description section contains information regarding details of the sensor like the sample rate at which it is gathering information or any special changes that have been made to the sensor itself.

The location tags in the sensor data section are similar to the element given during the sensor system context. But now they are used to describe the exact position of a sensor in a deployment. This tag is optional because sometimes it does not make sense to have a GPS based location and instead the relative location can be exemplified in the description element.

An important note is that a sensor does not necessarily have to be a real component. In fact, the schema encourages the use of virtual sensors that essentially aggregate sensor readings from several sensors or concatenate information in some fashion. With these virtual sensors, the type needs to have the keyword “virtual” listed in it. Furthermore, the description needs to contain the type of processing that is being taken place. Finally, a link can be provided to the actual transform that occurs. The transform can be a more detailed document outlining the processing that occurs or a reference to an Extensible Style Sheet Transform (XSLT) document.

In addition to context, there is a sample section which contains a set of values that are obtained from a typical sensor. A sample contains three distinct elements: unit, time, and value. The time is basically the timestamp associated with the sensor reading. This is given in the form of “CCYY-MM-DDThh:mm:ss” where “CC” represents the century, “YY” the year, “MM” the month and “DD” the day. The letter “T” is the separator between the date and time, and “hh”, “mm”, “ss” represent hour, minute and second respectively. In addition, the particular date/time combination can be annotated with the difference between the specific timezone that is being represented and the Coordinated Universal Time (UTC) by specifying a “Z” immediately after and the sign, + or -, and the difference from UTC represented as hours and minutes of the form hh:mm. The unit describes how the value is being expressed in terms of measurement. Examples of units include meters per second, Celsius, or volts. Finally, the value contains the output of the sensor itself based on its measurement of a certain phenomenon.

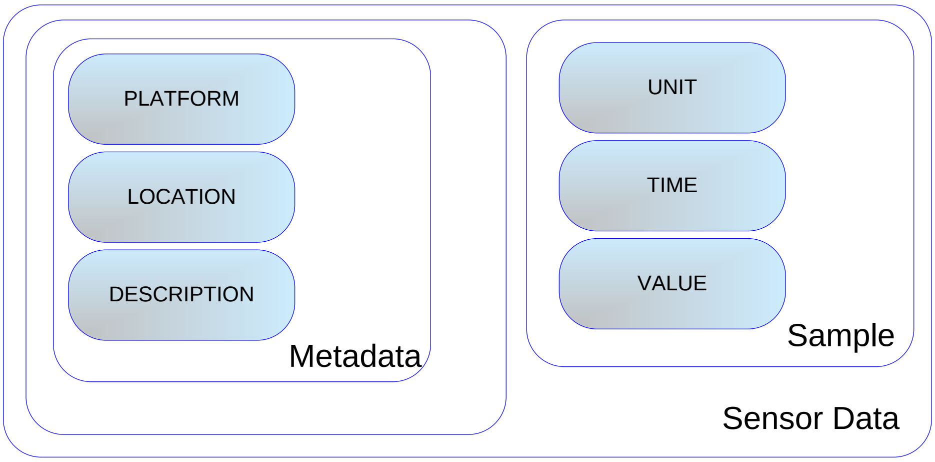

Figure 4: Sensor information for the SDSP

```
<sensor>
	<context>
		<metadata>
			<platform>
				<id>ISN006</id>
				<description>Telos B</description>
				<type>DS5</type>
			</platform>
			<location>28.137034,-15.438173,0.0</location>
			<description>Located in Las Canteras Beach near Peña La Vieja
			</description>
		</metadata>
	</context>
	<sample>
		<unit>mg/l(Ph)</unit>
		<time>2012-07-21T08:30:10</time>
		<value>7.8</value>
	</sample>
	....
</sensor>
....
```

Figure 5: Detail of the sensor element of the SDSP

4.5. Connection

The connection section contains links to other sensor stores that are related to the current system. This can be a connection to other sensor stores outputting in SDSP or a URI that represents some other form of data, such as sensor data that is outputted as a webpage. Figure 7 visually represents the various parts of the connection section. Basically, the resource component has four elements: name, link, description, and tag.

The name of the resource represents an alias given to the sensor store that is being linked.

The description contains information related to how this external resource is related to the one that is being described.

The tag element simply contains keywords, separated by commas that can be used to represent the external resource in an informal fashion.

Finally, the link element is a universal resource identifier to the external data source. The reason for the connection section comes from the fact that these SDSP stores can be adhoc or centralized. In both cases, discovery of other related elements becomes a major problem without having some type of guide to help jump start the process. To enable this type of discovery of other sensor stores the ability to link to resources that are related or even geographically located in a similar area is provided. Also, one can imagine that sensor stores that provide more reliable data will get more in-bound links. Thus, search engine services can use the link structure to rank sensor stores when returning results for a certain query.

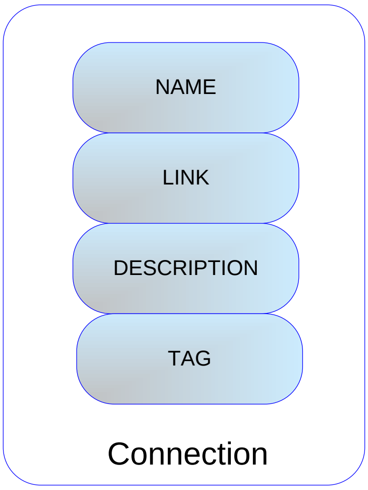

Figure 6: Connection information for other SDSP sources and related sensor systems

```
<connection>
      <resource name=Canary Observatory of Sea Water>
            <link>http://www.aguasmarinascanarias.es</link>
            <description>
                  Department of Oceanic Sciences
            </description>
            <tag>
                  DOS, Canary sea water
            </tag>
      </resource>
</connection>
```

Figure 7: Detail of the connection element of the SDSP

5. System Architecture

SDSP fits into an architecture that enables sharing, searching, and storage of sensor data, thus, we introduce an architecture that enables these types of services. Figure 8 shows the various components involved in the SDSP architecture.

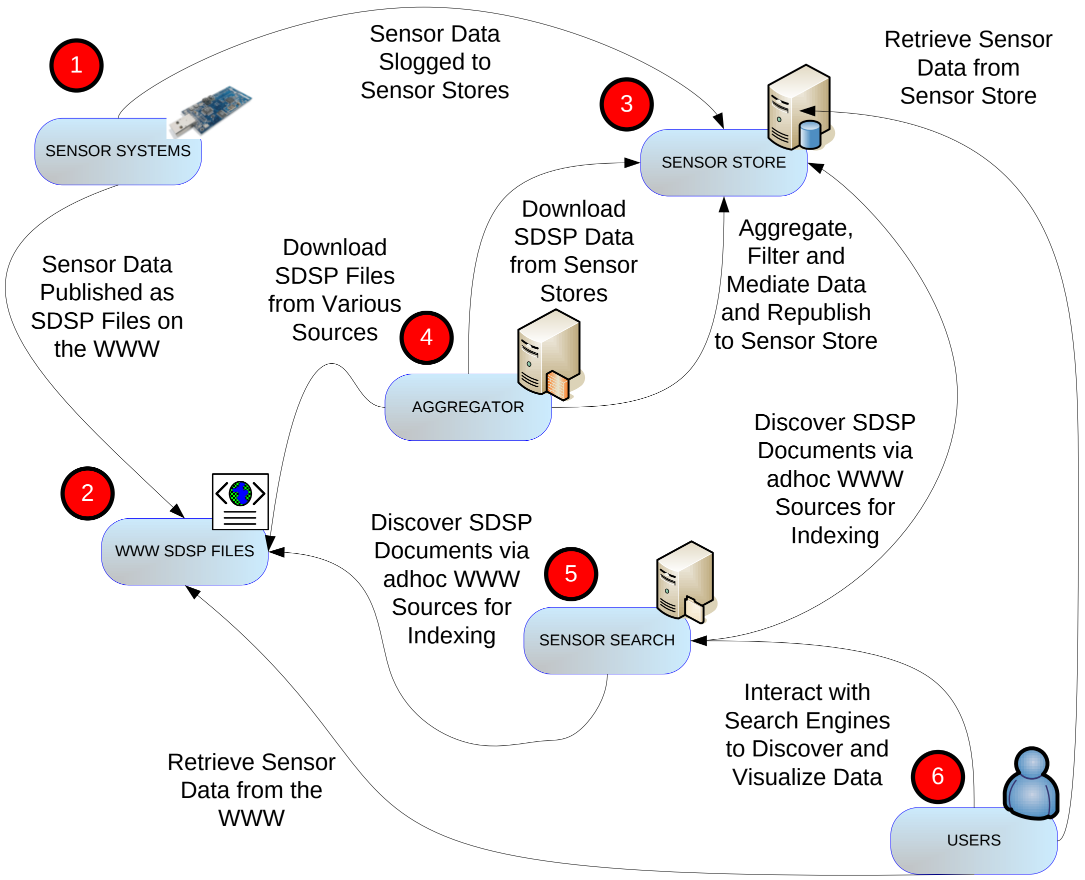

Figure 8: SDSP System Architecture

In principle, it would seem that the architecture is quite complex, but in reality, the functions of each entity in the system are fairly straight forward in terms of operations and the interactions are clearly definable.

5.1. Sensor Systems

Sensor systems are the actual data sources of the network. They can be resource-constrained devices like embedded devices or Motes which can measure such things as temperature, humidity, light intensity, etc. Often times the constrained devices are grouped together by a Gateway node and connected to Internet Protocol (IP) based network. These sensing systems publish data in SDSP format and make information available either in an adhoc fashion by publishing on independent web sites or send the data to specified sensor stores. By enabling the two forms of sensor data publication, we make sharing information easier and customizable to the preferences of the sensor system operator. If a user has access to services where information can be put up, then the sensing system can simply publish the SDSP XML file to the WWW similar to publishing a web site. On the other hand, if the user does not want to worry about storing the information for a long term basis, then the information can go to sensor stores, which is similar to the paradigm used by current users of such sites as FlickR and YouTube.

Finally, another important note about sensor systems is that they are highly heterogeneous in terms of both the software and hardware they run. Thus, tools need to be created that can run on various different platforms that publish data using SDSP format.

5.2. WWW SDSP Files

Sensor systems can publish or slog data in an adhoc manner without the use of centralized storage sources. In this case, SDSP data will be stored on web servers. This is similar to the current scheme of individual users publishing their own web pages. Thus, these SDSP files can exist on various different domains. Furthermore, the actual SDSP data can be simply XML files or data that is embedded in an existing webpage. Overall, by allowing this type of publication method, we enable users that have storage capabilities to user their own resources. Furthermore, since they are not using a separate storage source, they have full control of how the information is stored, published, and shown to the outside community. The only downside regarding this approach is that these sources of SDSP data might be less reliable in terms of uptime. Furthermore, backup of the data might not take place so if the WWW server that hosts the files goes down for some reason, then the files might not be cached in any way.

5.3. Sensor Store

Sensor stores are designed to provide a repository for sensor data. These sensor stores can range from being large and generally well connected and managed as well as small and run by a group of individuals. The basic difference between a sensor store and an adhoc storage mechanism is that the sensor store will have certain guarantees that it will be based on. For instance, the sensor store will actually store or save the data that is slogged to it. In certain situations the data provided by sensor systems might not be complete, in other words have gaps in terms of information. We can imagine that certain sensor systems would like to have the missing information filled through some type of approximation based on data. This can be accomplished at the sensor store by using other related information around the source sensor system and statistical estimation functions. Note that when this process takes place, the estimated data should be flagged by the sensor store. Also, the sensor store would resolve any external links that are used in the SDSP file and validate that the external information is available. Furthermore, the sensor store will have some guarantees in terms of availability of data, both in terms of uptime and the backup of information. Finally, the sensor store will provide a known named link that can be used as a pointer for other users of the data.

Another challenge that sensor stores possess is that they need to support a wide variety of disclosure models. For instance, say a sensor system provides surveillance images, along with additional context information, for a certain area in Parque del Retiro in Madrid. This information is only needed for people in the Madrid community, specifically the security personal. The administrator of the sensor system might not want users outside of the Madrid community to be able to access the data in any way. Furthermore, even in the Madrid community, certain groups might get more access than others. The security group of Madrid might get access to view all the data and even have permission to edit the data itself, but people that are actually in the area might only get access to a summary of the information. Thus, there exists an open research challenge to not only come up with the language for disclosure but also to implement the rules in an efficient, secure manner.

Sensor stores are considered an intermediary between data and sensor search engines. Thus, the sensor stores will need a specific interface for search engines to be able to get information regarding the data that is being stored. Meta data information will be very important for search engines, so this information will need to be relayed to search engines in an efficient manner. Furthermore, the meta data information will need to be as current as possible. Thus, if the sensor system logs an SDSP entry that has a link to the context or meta data information, then this data needs to be updated both initially and checked on a regular basis for changes. The sensor stores need to able to provide the sensor search engines with snapshots of the actual data that is being stored. This might include summaries of the behaviour in terms of noise exhibited, general trend, and even statistics on how often the information is updated or used.

Users must be able to access the information on sensor stores in a defined manner. One can imagine having browsable pages for each sensing system that allow users to get a snapshot of the data that is being produced and also having a programmable interface that end users can use to access the actual data. Security protocols must be in place to ensure that only users that should be able to access the data are actually able to.

An example of a sensor store is SensorBase (SensorBase.org)[13].

5.4. Aggregator

The basic idea of an aggregator is to combine various data sources and provide an alternative view of the data. This aggregation process can be used to provide a richer context in some way or just make the data more accessible for a certain group by simplifying it. There will be services that essentially download various SDSP data, either by finding them using search engines or directly contacting sensor stores and adhoc sources, and then perform some computation on the data and finally publish the information back to a sensor store. The aggregated information will be expressed as a virtual sensor and the actual computation performed will be detailed in the aggregation element of the SDSP.

To further illustrate the purpose and need for an aggregator, we will go through a simple example. Say there are two data sources, one that provides information about noise levels in a certain area and another that provides traffic data. The traffic data is provided by a government source and is slogged to a sensor store while the noise level information is provided by a group of individuals and is made available through a specific URL on a web site. If someone had the intuition that noise levels are directly related to traffic in a certain neighborhood, they can show this by employing an aggregation service. The service would essentially combine the two different SDSP data sources and provide an aggregated data source that shows both in a SDSP file with the appropriate time stamps. In this case, the aggregator simply did a concatenation of information. More complex manifestations of the aggregator might do some simple computation and only show event notifications of when traffic and noise levels are high at a certain location. As one can see from the above example, the actual computation would take place on the aggregator service platform. Thus, the aggregator would need to have an interface where users can provide computation tasks and will probably be closely tied to a sensor data search engine to find sources of data.

5.5. Sensor Search

Sensor search engines purpose is to index and search sensor data. We can imagine two types of sensor search engines: meta data search and signal search. Meta data search engines will enable users to find appropriate SDSP data sources based on context information such as location, types of sensors used, or platforms employed by the sensing systems. On the other hand, signal search will actually act upon the data in some manner. For instance, in the case of meta data search, a sample query might be to find all the temperature sensors in a five mile radius of the zip code 35011. The result from this type of query would be a list of SDSP data source URIs both that are adhoc and those that are at sensor stores. In the case of signal search, a query could be finding the least noisy spot in Las Palmas de G.C. Then, the search engine would have to actually answer this query by analyzing or at least getting a summary of the data represented by noise level sensors from the Las Palmas de G.C region and return the location that results in the least amount of noise pollution.

Research challenges exist in finding a mechanism to actually discover various sensor streams automatically. The connection section in SDSP can aid in this process along with interfaces to sensor stores. Furthermore, the SDSP sources will need to be indexed efficiently for searching purposes, and the meta data information provided by the sources has to be up to date. The search engine might have to give answers to queries even if certain data is missing or unreliable by using additional resources that exist. Search engines will need to provide a representation of the data in some visual fashion. Simple pages of text or data will not suffice in conveying the purpose or meaning of the actual data. Elements such as charts, graphs, and maps will need to be employed. Finally, the actual query language will need to be defined in a way that makes sense for the data that is provided. If there is a location based search mechanism then providing an interface that users can click on a map might be better than having the user specify the location context using keywords. The same kind of visual search idea can be applied to specifying trends for data as well. Thus, the query user interface will need to be explored as part of the sensor search engine research plan.

5.6. Users

Users or consumers of the data will interact with the system mainly by using search engines. Basically, they will issue queries to the search engines, and the search engines will either return a snapshot of the actual data to answer

the query or links to the SDSP sources for the user to interact with. Much like normal web surfers, users of SDSP data sources can find adhoc sources by simply surfing the web. They can take a look at the SDSP data and the context surrounding it if they want. Also, they can issue queries to sensor stores regarding the data. Thus, interaction works in three folds: web surfing for adhoc sources, querying sensor stores, and using sensor search engines.

6. References

[1] http://www.itea2.org/

[2] Mike Botts, editor. OpenGIS Sensor Model Language (SensorML), Implementation Specification.

[3] Simon Cox, Paul Daisey, Ron Lake, Clemens Portele, and Arliss Whiteside, editors. OpenGIS Geography Markup Language (GML) Implementation Specification.

[4] N. Ota and W.T.C. Kramer. TinyML: Meta-data for Wireless Networks.

[5] Dave Winer. RSS 2.0 Specification. http://blogs.law.harvard.edu/tech/rss, 2002.

[6] Wikipedia. Atom Syndication Schema. http://wikipedia.com/atom, 2006.

[7] M. Colagrosso, W. Simmons, and M. Graham. Demo abstract:Simple sensor syndication. In SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems, 2006.

[8] Agilent Labratories. Numerical RSS: A protocol for Syndicating Numeric Data. http://nrss.org, 2005.

[9] S. Decker, S. Melnik, F. van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The Semantic Web: the roles of XML and RDF. Internet Computing, IEEE, 4(5):63–73, 2000.

[10] K. Lee. IEEE 1451: A standard in support of smart transducer networking. Instrumentation and Measurement Technology Conference, 2000. IMTC 2000. Proceedings of the 17th IEEE, 2, 2000.

[11] P. Buonadonna, D. Gay, J.M. Hellerstein, W. Hong, and S. Madden. Task: Sensor network in a box. Proceedings of the Second IEEE European Workshop on Wireless Sensor Networks and Applications, 2005.

[12] Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tinydb: an acquisitional query processing system for sensor networks. ACM Trans. Database Syst., 30(1):122–173, 2005.

[13] D. Estrin, R. Guy, K. Chang, M. Hansen and E. Graham. Sensorbase. http://www.sensorbase.org, 2006.