Pan-American Association of Computational Interdisciplinary Sciences

Journal of Computational Interdisciplinary Sciences (2011) 2(1): 13-22
© 2011 Pan-American Association of Computational Interdisciplinary Sciences
Printed version ISSN 1983-8409 / Online version ISSN 2177-8833
http://epacis.org

Handling and publishing wireless sensor network data: a hands-on experiment

Ivo K. Koga[1], Claudia Bauzer Medeiros[1] and Omar Branquinho[2]

Manuscript received on August 10, 2010 / accepted on January 30, 2011

ABSTRACT

eScience research, in computer science, concerns the development of tools, models and techniques to help scientists from other domains to develop their own research. One problem which is common to all fields is concerned with the management of heterogeneous data, offering multiple interaction possibilities. This paper presents a proposal to help solve this problem, tailored to wireless sensor data – an important data source in eScience. This proposal is illustrated with a case study.

Keywords: WSN data management, Software Components, OSGi.

1 INTRODUCTION

Wireless Sensor Networks (WSN) [2] are a special kind of *ad hoc* network, composed of a huge amount of small nodes with low processing capacity, limited power source, high mobility and higher probability of failures than other kinds of networks due to communication, power, and/or node failures. Nodes potentially have different types and functionalities and monitor a wide scale of physical and environmental variables (e.g. temperature, humidity).

WSNs allow the acquisition of data in difficult conditions, for a wide range of spatial and temporal resolutions and scales. The sensors can be intimately connected with the observed phenomena, being kept active during a long time, being deployed everywhere – under the sea, underground or in space. Ubiquitous and pervasive, they can be also implanted in our bodies (and thus generate data for eHealth studies) or our home (for ambient applications).

This possibility of monitoring many phenomena, in various temporal and spatial scales, produces a large volume of heterogeneous data. Heterogeneity and volume of data, combined with heterogeneity in user requirements, pose many problems. The storage, retrieval and visualization of data in this kind of setting is a challenge which is associated with the first Grand Challenge in computer science defined by the Brazilian Computer Society – Management of large distributed multimedia data volumes [13].

The goal of this work is to contribute towards solving one of the many facets of this grand challenge, by proposing a practical way of storing and publishing sensor data, making possible the extraction of information by different types of users. Each kind of user profile can determine their special needs, defining what they want from the available data allowing the extraction of relevant information.

This work is related with ongoing research on the management of sensor data in eScience – in particular, for biodiversity and environmental studies. We present our proposal by means

Correspondence to: Ivo K. Koga – E-mail: koga@ic.unicamp.br
[1]Institute of Computing, State University of Campinas (UNICAMP), 13084-971 Campinas, SP, Brazil.
[2]PUC, Campinas, 13086-900 Campinas, SP, Brazil.

of a case study of management of environmental data in an application related to agriculture.

The publication of sensor data, on the Web, involves issues that go beyond the nature of the data being collected and are intimately related with problems of the Web itself – such as data heterogeneity, privacy, volume of data and user requirements. Hence, additionally, we investigate this proposal under the light of the perspective of Web Science [3].

The main scientific contributions are the following:

- framework that combines distinct technological solutions;
- interoperability to support sensor data publication;
- discussion of sensor networks in the context of Web Science;
- validation of the framework for a real case study, emphasizing extensibility and flexibility.

The rest of this paper is organized as follows. Section 2 presents a brief overview of WSN data management. Section 3 presents our approach. Section 4 discusses our case study and section 5 presents ongoing work.

2 RELATED WORK

This paper concers the handling and publication of large volumes of sensor network data. There is a wide range of open problems in this domain. This paper is concerned with the issues of flexibility in data publication on the Web and interoperability across networks. Thus, we concentrate on discussion of work on publication of data in eScience, and some Web Science issues.

2.1 Data publication – interoperability and flexibility issues

Different types of systems are being proposed and deployed to support scientists' work in many research areas handling heterogeneous data sources, including sensor data. Biodiversity systems are an example of this type of system to support the work of biologists. Examples are studies in ecology or environmental monitoring. On closed environments, sensor networks are being used in scientific studies concerning health (e.g. patient monitoring) or chemistry (experiment monitoring) – see [7].

In all these contexts, there are countless initiatives concerning WSN data management, that range from network configuration and energy management to data processing and publication [15, 8, 26, 18]. This paper is concerned with solutions that support the latter stage – i.e., once data are collected, how to provide flexible mechanisms that will forward data to be processed and published, hiding low-level details. Our choice of related work reflects this, concentrating on architectures for sensor data management and publication.

2.1.1 Architectures to support flexibility

Many solutions have been proposed to overcome the problems of heterogeneity and interoperability of sensor data management. Chu et al. [4] created an architecture called NICTA[3] Open Sensor Web Architecture (NOSA) which combines a Service Oriented Architecture and WSNs using the services specified in the Sensor Web Enablement [17] from the OpenGIS Consortium. Figure 1, reproduced from [4], shows NOSA and its components. There are 4 layers: Sensor Fabric, Application Services, Application Development and Applications. The first layer deals with sensors and their emulation/simulation, the second is composed by services that support network management, the third provides the APIs, tools and configuration and the last has the applications that use the sensor data.

In NOSA, sensor data are always processed by entities external to the sensor network. This can be an advantage in scenarios where the deployment and maintenance of the sensors are easy. These scenarios consider that sensors will only sense and send data, without any processing, which consumes more power (most expensive activity in face of power consumption) [2]. However, in specific scenarios, it could be more appropriate to use WSN preprocessing capacity before actually sending data. Another disadvantage is that applications cannot reuse code.

Pastorello Jr [19] also followed a multiple layer approach, but from another perspective. He dealt with the problem of production and management of WSN data through a framework that uses software components called Digital Content Components [21] and scientific workflows to provide management facilities and easy access to the sensor data. Unlike NOSA, this work does not consider the OpenGIS Consortium standards for WSNs. However, it has some advantages such as flexibility, letting open the possibilities for development of new components for access and management of sensors, regardless of the sensors' implementation and technology.

Global Sensor Networks (GSN) [1] is a platform developed in Java that provides an infrastructure for the integration of technologies of heterogeneous sensor networks using a set of abstractions and XML. GSN has the advantage of facilitating the WSNs

[3]Australia's Information and Communications Technology (ICT) Centre of Excellence.

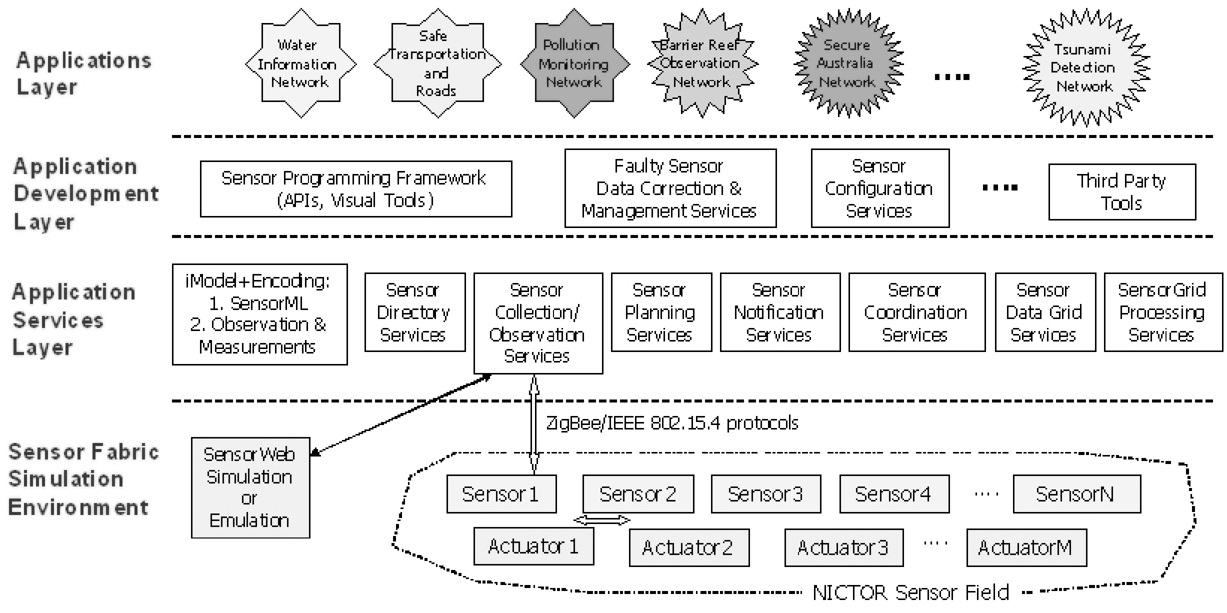

Applications Layer
Water Information Network
Safe Transportation and Roads
Pollution Monitoring Network
Barrier Reef Observation Network
Secure Australia Network
Tsunami Detection Network
Application Development Layer
Sensor Programming Framework (APIs, Visual Tools)
Faulty Sensor Data Correction & Management Services
Sensor Configuration Services
Third Party Tools
Application Services Layer
iModel+Encoding: 1. SensorML 2. Observation & Measurements
Sensor Directory Services
Sensor Collection/ Observation Services
Sensor Planning Services
Sensor Notification Services
Sensor Coordination Services
Sensor Data Grid Services
Sensor Grid Processing Services
ZigBee/IEEE 802.15.4 protocols
Sensor Fabric Simulation Environment
Sensor Web Simulation or Emulation
Sensor1
Sensor 2
Sensor3
Sensor4
SensorN
Actuator 1
Actuator2
Actuator3
ActuatorM
NICTOR Sensor Field

Figure 1 – NICTA Open Sensor Web Architecture.

deployment when it hides its implementation details. On the other hand, it also hides platform specific parameters that might render each deployment more flexible.

2.1.2 Handling heterogeneity for specific applications

While the previous section concerned generic architetural solutions, other kinds of sensor-related research propose solutions that are tailored to specific applications, or application domains. This requires, for instance, designing special purpose databases, or developing special software.

For instance, the GeoCENS project [11] focus on capturing data from local scale sensor networks, deployed and operated by individual scientists. These kinds of data are more likely to remain underutilized and eventually lost. The project deals with challenges like heterogeneity (there is much more heterogeneity in these deployments), protecting data ownership (researchers spend time and funding on data collection, so they want to protect their property), motivation (there should be incentives to motivate people to publish their data), and provide an intuitive and coherent user interface. For instance, they consider digital watermarking to protect sensor data and have created a 3D web interface to allow users to manipulate sensor data in a more intuitive way.

The Southeast Alaska MOnitoring Network for Science, Technology, Education and Research (SEAMONSTER) [6] was developed and deployed in Alaska to study glaciated watersheds. It stores all the data measured in a PostGIS database. The SensorWeb Enablement (SWE) protocols from Open Geospatial Consortium (OGC) [20] were used in SEAMONSTER in order to provide interoperability. Geoserver [16] was used to deliver dynamically generated geospatial output in their web portal. KML (Keyhole Markup Language) was adopted for interoperability. KML is an XML language focused on geographic visualization, including annotation of maps and images. The project developed a portal to provide temperature, humidity, precipitation and voltage data using openLayers and accessing Bing from Microsoft to give the node location.

The Life Under Your Feet project was developed and deployed for soil monitoring at an urban forest in Baltimore [22]. It measures and saves soil moisture and temperature *in situ*. Their key requirements for soil ecology sensor systems include fidelity, accuracy, precision, sampling frequency, fusion with external sources, experiment duration and deployment size. Their solution, employed at a micro underground scale, is now being ported to a very different environment – monitoring conditions in Brazil's rainforest.

2.1.3 Interoperability and publication

The problems faced by all these proposals analyzed in this section range from a micro perspective (a large amount of sensors in a single network) to a macro one (between WSNs and between them and the Web). Pastorello Jr proposed components and workflows to deal with the heterogeneity problem. GSN was proposed as an infrastructure to overcome some deployment problems using XML and abstractions implemented in Java. NOSA encapsulates the operations in a software layer that uses the Sensor Web Enablement standards and grid computing to provide a middleware that provides services that overrides sensors' implementation complexity.

On the application side, GeoCENS dealt with the absence of local scale sensor network data. SEAMONSTER deployed a system that provides some measurements and publishes it on a simple interface using openLayers and Bing, geared towards specific user needs. Life Under Your Feet provided soil measurements and its database considers requirements to provide quality in monitoring sensor networks for scientific data for underground soil conditions.

In these and other efforts, the idea is to provide several layers of isolation between the sensor networks and the users. Then, one can customize and develop each layer and concentrate on the solution of a few problems at a time. As will be seen in section 3, our proposal combines features from some of the reviewed papers.

2.2 Publishing sensor data on the Web – a few Web Science concerns

The discussion on Section 2.1 concerned interoperability and publication issues, for eScience needs. However, most of those projects are concerned with the Web environment, which is increasingly becoming a prime environment for eScience research.

In this context, which also touches our work, we should also look at associated issues, in Web Science.

The term Web Science was first introduced by Berners-Lee in [3]. It has since given origin to large international research efforts, including The Web Science Trust [24]. In Brazil, the theme motivated a National Institute of Science and Technology (INCT) in Web Science – the Brazilian Institute for Web Science Research [5].

Formally, research in Web Science is concerned with the Web as the primary object of interest. Thus, rather than considering the Web as a medium for collaboration, communication and socializing, it studies the Web itself. In our work, this means among others concentrating in two issues:

- the effects of data publication on the Web and its impact on the long tail of data e.g., see [11];
- the use of Web Services as a basis for interoperability.

Long tail concerns are becoming increasingly popular among eScientists. The idea is to access data collected by thousands of individual researchers, but which are difficult to find. Publication of these data on the Web makes sure they become public, but does not ensure their accessibility, nor their visibility (both of which Web Science concerns [3]).

Data publication via Web Services increases accessibility, since service interfaces must follow specific standards. However, these same standards sometimes hamper particular needs. For instance, services do not allow updates. Also, since they have been conceived to enhance interoperability, they may hide information that would be useful to an end-user – e.g., sampling frequency or data quality provided by a sensor gateway. Thus one must consider extending services, to provide more flexibility.

Visibility is even more complicated. Since, as shown in section 3, we use components to display sensor data coupled to services, additional semantics must be conceived for publication. This is subject of future work.

3 PROPOSED SOLUTION

We are concerned with interoperability and publication flexibility, as two facets of the first Grand Challenge. Our solution is based on two aspects:

- Web services – to provide interoperability between applications, WSNs, data servers and user applications;
- Components – to support reuse and loose coupling, and to allow multiple user-friendly visualizations of sensor data.

Figure 2 gives a high level view of our proposal. It has three main components (or layers): WSNs (on the left), data servers (on the right), and user applications. Data communication among components is supported by Web Services. Specific functionalities are implemented by software components. Each WSN is assumed to connect to an access point. Each access point runs a Data Load service that sends (pushes) the raw data to a central data server.

The data server (right side of the figure) implements two Web Services: a Receive service and a Publication service. The Receive service formats the data received from each Data Load service into standard tuples, and stores them in a database. The Publication service publishes basic methods that execute SQL queries on the database.

User requests are treated as follows: Distinct query parameters and visualization requests are implemented as components that invoke the Publication service. Storage and visualization

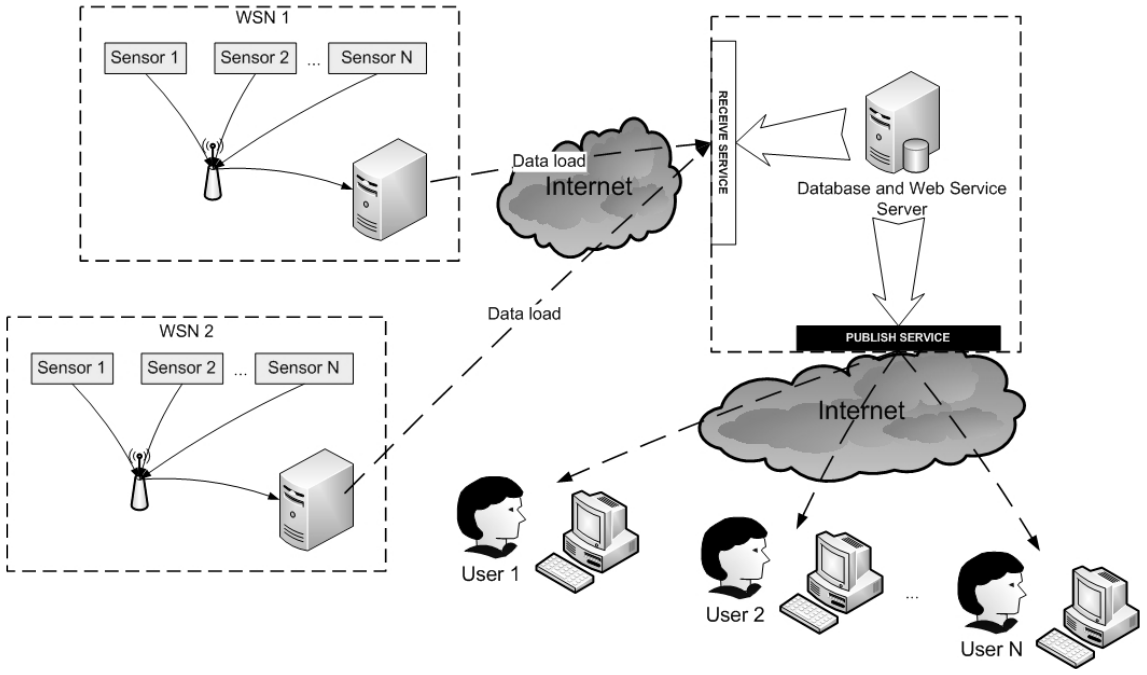


Figure 2 – Architecture of the solution.

thus follow two independent pipelines. In the "push" pipeline, Data Load pushes data into the Receive service. In the "pull" pipeline, software components request data from the Publication service. Hence, different applications can build their own components and this provide user-tailored visualizations.

This solution has the following merits. First, it takes advantage of Web Services to provide access interoperability. Second, since it is based on components, it is extensible – e.g., components can also be developed at the WSN access point side to preprocess the data before it is sent to the server – e.g., providing fusion facilities. Alternatively, as in our case, services can be developed at the server side to integrate and customize data according to distinct application requirements. Also, different servers can be installed to support, for instance, distinct needs or to integrate data from different networks.

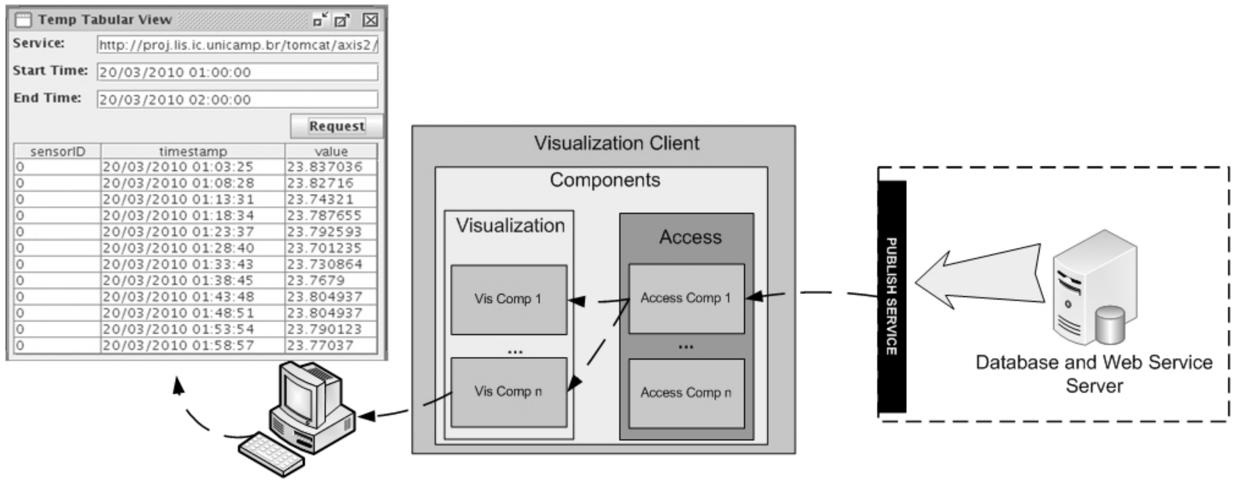
The use of software components makes possible the development of user-specific components to access and visualize WSN data. This is shown in Figure 3 where we have distinct components for accessing and visualizing data, separating this in a Model-View-Controller pattern [10]. Here, one access component can be used by many visualization components and also one visualization component can use many access components.

In order to provide a first prototype for visualization of sensor data, we used the FLAVOR framework [9]. FLAVOR was developed to support flexible design and construction of software components to visualize measurements of network traffic. We point out that such measurements can be treated as time series – and thus FLAVOR was used to visualize our sensor measurements. As new requirements to access and visualize WSN data appear, FLAVOR may need to be progressively extended.

Our solution combines aspects from NOSA and Pastorello's work (see section 2). From the latter, it adopts the philosophy of components to encapsulate functionality and increase modularity. From NOSA, it uses aspects of publication using Web Services, thereby increasing interoperability. Moreover, we treat the heterogeneity problem at the storage level, standardizing the format to store sensor data. Thus, the role of components is to provide distinct visualization formats, including simultaneous views of multiple sensors.

Our infrastructure uses OSGi [23] as the software component standard. The OSGi defines a standard and component oriented environment that provides a Java framework which supports the deployment of extensible components called bundles.

OSGi provides standard primitives that allow applications to be constructed through small, reusable and collaborative components. It manages the installation and update of bundles in a dynamic and scalable way. It also provides resources for one to take advantage of the dynamic load of code, that is, it allows the dynamic deployment of components in the environment without the need of restarting the entire application. [12]



Service: http://proj.lis.ic.unicamp.br/tomcat/axis2/

Start Time: 20/03/2010 01:00:00

End Time: 20/03/2010 02:00:00

sensorID	timestamp	value
0	20/03/2010 01:03:25	23.837036
0	20/03/2010 01:08:28	23.82716
0	20/03/2010 01:13:31	23.74321
0	20/03/2010 01:18:34	23.787655
0	20/03/2010 01:23:37	23.792593
0	20/03/2010 01:28:40	23.701235
0	20/03/2010 01:33:43	23.730864
0	20/03/2010 01:38:45	23.7679
0	20/03/2010 01:43:48	23.804937
0	20/03/2010 01:48:51	23.804937
0	20/03/2010 01:53:54	23.790123
0	20/03/2010 01:58:57	23.77037

Figure 3 – Components to request published data and visualize it.

Using OSGi, our environment can be updated at execution time (without the need of restarting). For instance we can add new functionalities at the WSN client, preprocessing the data before sending it to the data server. This new functionality can be added seamlessly by just installing a new component at the WSN client, without the need of stopping and restarting the application and without the need of interrupting the data flow from the WSN to the data server.

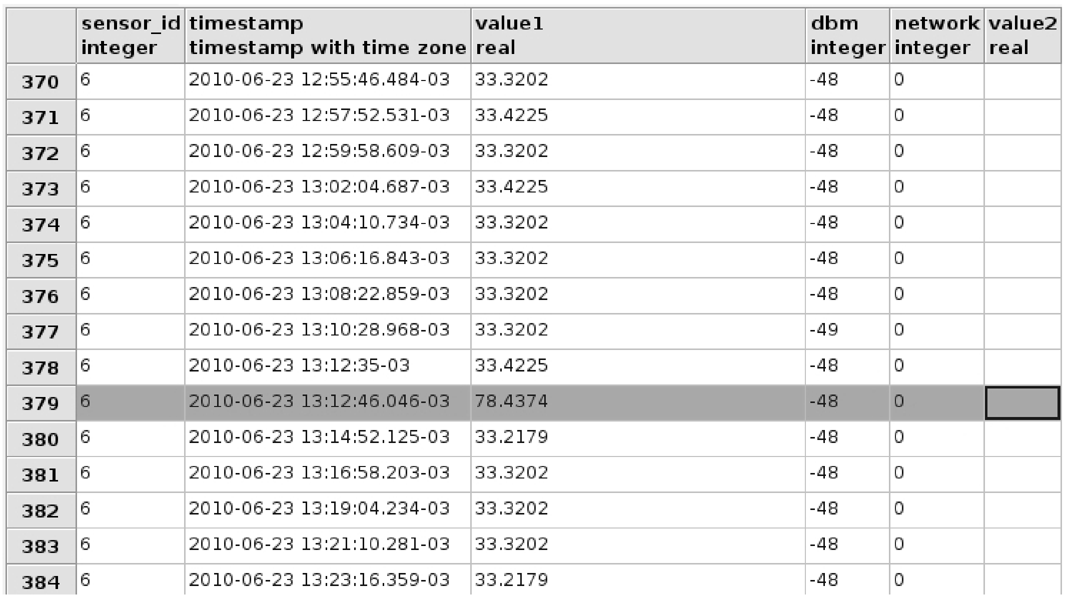
Our solution also provides flexibility for multiple kinds of queries – e.g., involving aggregation, interpolations or transformations. There are two ways of doing that: adding new methods at the Publication service or implementing new components at the user application level. The former is done by adding new methods to the service that will map to the PostgreSQL queries. The latter can be implemented as a chain of components that implement such functions on top of basic invocations of the Publication service (i.e., our solution differs from others, since instead of changing the service we add components). In such a case, a request for an aggregation over a period of time for "n" temperature measurements can be translated into an execution of a sequence of components – the first component will request from the Publication service data and the second will compute the aggregation.

4 CASE STUDY: A HANDS-ON EXPERIMENT

Our case study concerns managing data from a WSN deployed at the Faculty of Agriculture Engineering (FEAGRI) at UNICAMP. Sensor data are collected at an access point installed at FEAGRI, to be processed at the Laboratory of Information Systems (LIS), at the Institute of Computing. For this experiment, we were concerned with basically three issues – sensors heterogeneity, data publication and processing. To create a test case of heterogeneous sensors, we collect data from sensors sensing air humidity, light and temperature. Even though we had a small amount of sensors, we had to deal with 3 types of measurement, each with different frequency of data acquisition and units.

All the sensors send data to a local base station which is connected to a computer (the access point) that has a web service (the Data Load service). This service, developed by us, sends the measured sensor data to a server located at LIS. This WS server (Receive Service) at LIS then stores the data in an appropriate way in a PostgreSQL database. In our specific implementation, data are collected at about 2 minute intervals. Finally, data are published through another WS (Publication Service).

Since our concern in this first stage was in interoperability and publication flexibility, our data records are very simple and store a minimum of information. Figure 4 shows a short list of records in the raw sensor data database. This table has 6 attributes: record id, sensor id, value, timestamp (when measure was taken), and network id. The last attribute, named value2, indicates whether this sensor is capturing more than one value. This particular table is a snapshot of a humidity measurement in june 23rd, 2010. Record 379 shows an outlier at Universal Time 13:12:46 where it measured 78.4, while all other measurements for the same time period had values between 33.2 and 33.4. The publication of these data must take this into consideration – here, this was probably due to some sensor malfunction. For temperature measurements, our extension to FLAVOR consisted in creating two components: TempSensorAccess and TemperatureSensorTabularView, respectively the access and the visualization



	sensor_id integer	timestamp timestamp with time zone	value1 real	dbm integer	network integer	value2 real
370	6	2010-06-23 12:55:46.484-03	33.3202	-48	0	
371	6	2010-06-23 12:57:52.531-03	33.4225	-48	0	
372	6	2010-06-23 12:59:58.609-03	33.3202	-48	0	
373	6	2010-06-23 13:02:04.687-03	33.4225	-48	0	
374	6	2010-06-23 13:04:10.734-03	33.3202	-48	0	
375	6	2010-06-23 13:06:16.843-03	33.3202	-48	0	
376	6	2010-06-23 13:08:22.859-03	33.3202	-48	0	
377	6	2010-06-23 13:10:28.968-03	33.3202	-49	0	
378	6	2010-06-23 13:12:35-03	33.4225	-48	0	
379	6	2010-06-23 13:12:46.046-03	78.4374	-48	0	
380	6	2010-06-23 13:14:52.125-03	33.2179	-48	0	
381	6	2010-06-23 13:16:58.203-03	33.3202	-48	0	
382	6	2010-06-23 13:19:04.234-03	33.3202	-48	0	
383	6	2010-06-23 13:21:10.281-03	33.3202	-48	0	
384	6	2010-06-23 13:23:16.359-03	33.2179	-48	0	

Figure 4 – Measurement of humidity from sensor 6 on june 23rd, 2010.

components for accessing the temperature sensor data available. Figure 5 shows an example of visualization of temperature data, using a table format. An alternative means of visualizing the sensor data appears in Figure 6. This was developed using a distinct software, but using the same underlying data stored in PostgreSQL. Such flexibility in handling data is only possible because of our architectural choices.

There were several difficulties in deploying the first sensors, ranging from engineering problems to defining a storage format for data. For instance, the setting up of the communications infrastructure and the calibration of sensors took more than one year.

Web Services was one of the many solutions discussed for communication between layers. Again, service specification and implementation was time consuming. However, once the services were running and the network was deployed, the extension of access and visualization alternatives is proving to be relatively straightforward, because of the architectural solution adopted.

5 PUBLISHING DATA IN THE CLOUD

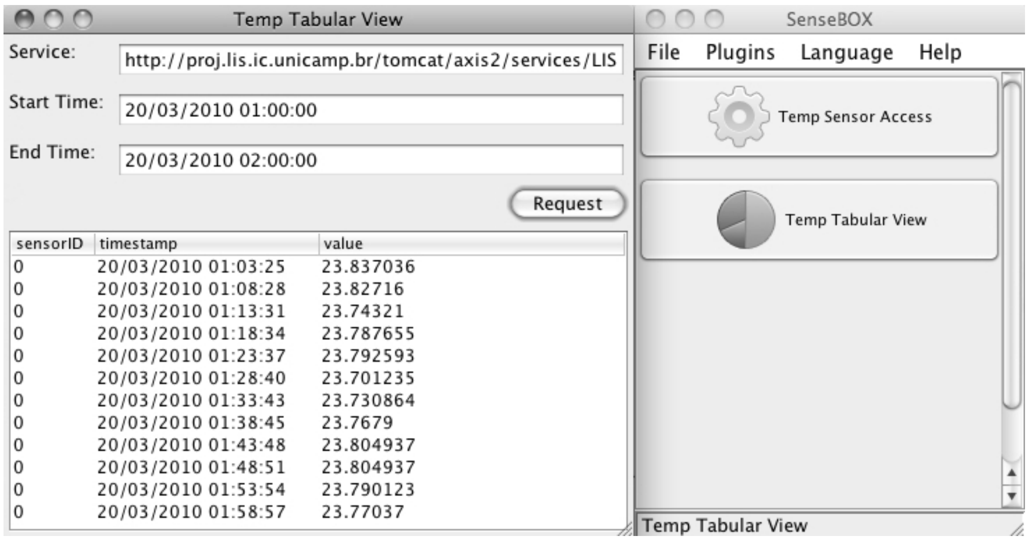
To attest the flexibility of our approach, we conducted another experiment using cloud computing [25] – in our case, Microsoft Windows Azure [14]. This was accomplished in a very straightforward way: we just had to modify our Receive service to store the WSNs data into the local dabatase and into the cloud. In more detail, this, was performed as follows:

1. the data that came from the WSNs continued to flow to the local database in the same predetermined frequency (i.e., about every 2 minutes);
2. the data that came from the WSNs into the cloud were stored according to the real collection frequency of the WSNs (i.e., one or more data points per minute), without the need to reduce sampling frequency to save storage space

The applications querying the data from the data server infrastructure continued to access the Publication service, getting the data of interest without any need of code modification. At the same time, we developed another application, to query the cloud database and display the data. Again, this just required a simple access to the cloud service to get data from there.

Figures 7 and 8 show charts of temperature and humidity, respectively, from the cloud (a) and from the local database (b). The underlying databases are different (PostGreSQL in our server, SQL Server in the cloud) and the data volume is different (one or more values per minute in the cloud, one value per 2 minutes in our local server). However, at the visualization/application end, the graphs are presented at 5 minute average intervals, thereby showing the same curves. The final visualization is nevertheless different because two distinct rendering algorithms were used, one for each application.

Extending the problem to run in another platform was simple and straightforward, thanks to our infrastructure choices. We just had to update the Receive Service we provide for the clients



Service: http://proj.lis.ic.unicamp.br/tomcat/axis2/services/LIS

Start Time: 20/03/2010 01:00:00

End Time: 20/03/2010 02:00:00

sensorID	timestamp	value
0	20/03/2010 01:03:25	23.837036
0	20/03/2010 01:08:28	23.82716
0	20/03/2010 01:13:31	23.74321
0	20/03/2010 01:18:34	23.787655
0	20/03/2010 01:23:37	23.792593
0	20/03/2010 01:28:40	23.701235
0	20/03/2010 01:33:43	23.730864
0	20/03/2010 01:38:45	23.7679
0	20/03/2010 01:43:48	23.804937
0	20/03/2010 01:48:51	23.804937
0	20/03/2010 01:53:54	23.790123
0	20/03/2010 01:58:57	23.77037

Figure 5 – Application visualizing Temperature sensor data in a data table.

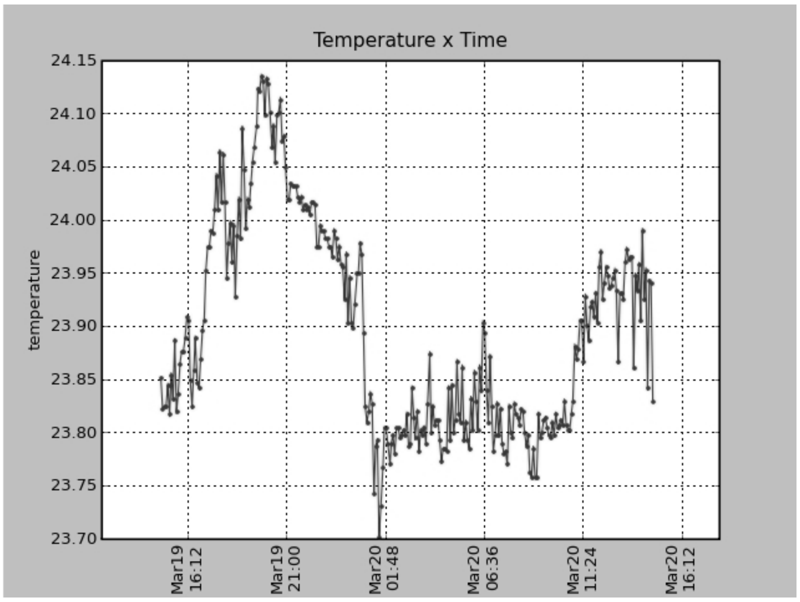


Figure 6 – Another view of some of the data presented in Figure 5.

to store data and create another application to access the cloud, for the same basic visualization. We point out that our goal was not to compare the performance of the two experiments (with and without the cloud). Rather, the idea here was to check the extensibility and flexibility of our solution.

6 CONCLUSIONS AND ONGOING WORK

This paper presented our proposal to process and visualize sensor network data. Our approach is based on combining Web Services (to provide interoperability among sensor networks, data servers and user applications) and components (to provide flexibility in data preprocessing and visualization). Since OSGi was used as the component standard, we can dynamically update an application at execution time without the need to restart the entire application. New functionalities (components) can be added or removed without breaking the data flow from the WSN to the data server or restart the user visualization application. As a consequence, the features that are not involved in the update will not be affected and the new ones can be instantaneously run.

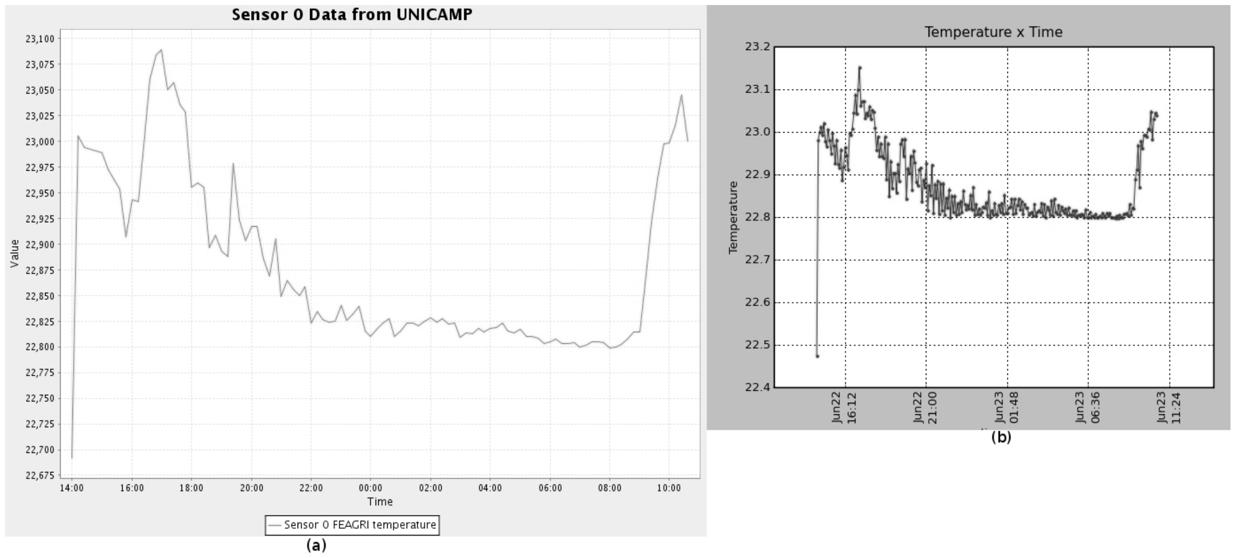


Figure 7 – Temperature data in the cloud (a), and in the local database (b).

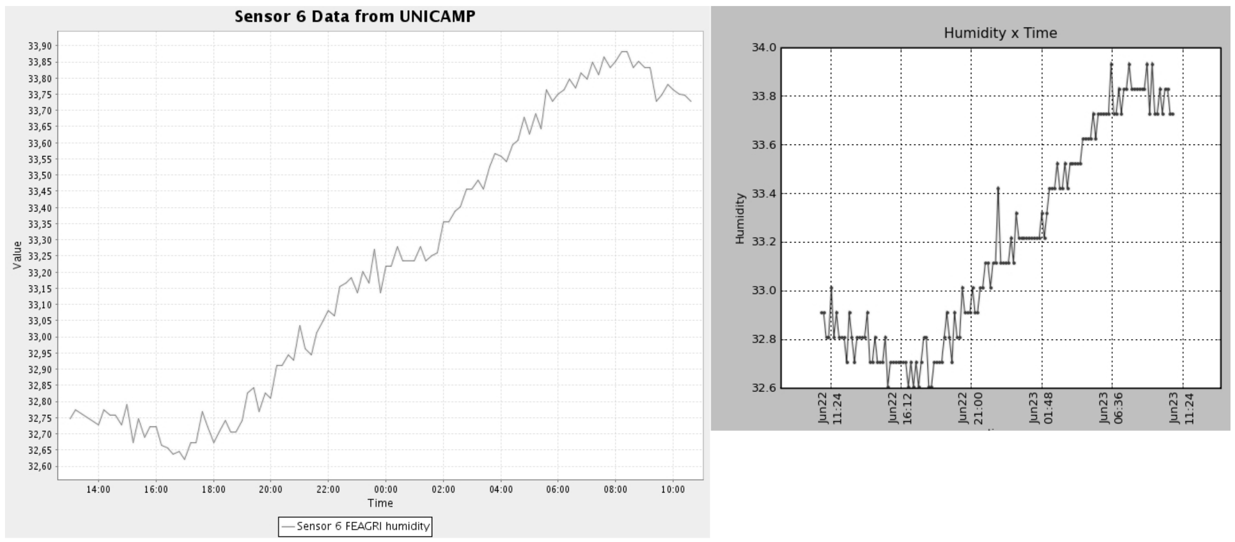


Figure 8 – Humidity data in the cloud (a), and in the local database (b).

There are many directions for continuing this work, that range from solving issues such as detecting faulty sensors to providing users with a wide range of visualization and filtering options. Another direction involves extending the sensor data database with additional information – e.g. on quality – such as adding attributes discussed by [22].

At the same time, we need to concern ourselves with the Web Science issue of visibility. It is not enough to publish data on the Web: indeed, means must be found to ensure that these data are found and correctly interpreted. One possibility is to register the Publication Services, in which the registration is enhanced with enough semantic information – e.g., with ontological annotations. This kind of solution is part of our ongoing research.

ACKNOWLEDGMENTS

This work was partially financed by FAPESP (grant number 09/52336-7), the Microsoft Research Fapesp Virtual Institute (eFarms project), CNPq (BioCORE project), INCT in Web Science (CNPq 557.128/2009-9) and CAPES, as well as a project grant from CNPq.

REFERENCES

[1] ABERER K, HAUSWIRTH M & SALEHI A. 2007. Infrastructure for data processing in large-scale interconnected sensor networks. In International Conference on Mobile Data Management (MDM), pp. 198–205.

[2] AKYILDIZ IF, SU W, SANKARASUBRAMANIAM Y & CAYIRCI E. 2002. Wireless sensor networks: a survey. Computer Networks, 38: 393–422.

[3] BERNERS-LEE T, HALL W, HENDLER JA, O'HARA K, SHADBOLT N & WEITZNER DJ. 2006. A framework for web science. Found. Trends Web Sci., 1(1): 1–130.

[4] CHU X, KOBIALKA T, DURNOTA B & BUYYA R. 2006. Open sensor web architecture: Core services. In Proceedings of the 4th International Conference on Intel ligent Sensing and Information Processing (ICISIP 2006) (Piscataway, New Jersey, USA, 2006), IEEE Press, pp. 98–103.

[5] FOR WEB SCIENCE RESEARCH, B.I. 2010. Brazilian institute for web science research. http://webscience.org.br, accessed in August 2010.

[6] HEAVNER M, FATLAND DR, HOOD EW & CONNOR CL. 2010. SEAMONSTER: A wireless Sensor Web prototype applied to studying glaciated watersheds.
http://esto.nasa.gov/conferences/estf2010/papers/
Heavner_Matt_ESTF2010.pdf, accessed in August 2010.

[7] HEY T, TANSLEY S & TOLLEY K., Eds. 2009. The Fourth Paradigm: Data-Intensive Scientific Discovery. Microsoft Corporation.

[8] HUANG C-F, TSENG Y-C & WU H-L. 2007. Distributed protocols for ensuring both coverage and connectivity of a wireless sensor network. ACM Transactions on Sensor Networks, 3: 1–24.

[9] KOGA IK, SAMPAIO L & MONTEIRO JAS. 2007. FLAVOR: A dynamic and open framework for the development of network measurement access and visualization tools. In XXV Brazilian Symposium on Computer Networks and Distributed Systems (SBRC) (Belém, PA, 2007), pp. 665–678.

[10] KRASNER GE & POPE ST. 1988. A cookbook for using the model-view controller user interface paradigm in smalltalk-80. J. Object Oriented Program., 1(3): 26–49.

[11] LIANG SH, CHEN S, HUANG C-Y, LI R-Y, CHANG DY, BADGER J & REZEL R. 2010. Capturing the long tail of sensor web. In International Workshop on Role of Volunteered Geographic Information in Advancing Science, part of GI-Science 2010 (Zurich, Switzerland, 09/2010 2010).

[12] MARPLES D & KRIENS P. 2001. The open services gateway initiative: An introductory overview. IEEE Communications Magazine, 39: 110–114 (Dec. 2001).

[13] MEDEIROS C. 2008. Grand Research Challenges in Computer Science in Brazil. IEEE Computer Society, 41(6): 59–65 (June 2008).

[14] MICROSOFT. 2010.
http://www.microsoft.com/windowsazure/. Windows azure, accessed in August 2010.

[15] NAKAMURA EF, LOUREIRO AAF & FRERY AC. 2007. Information fusion for wireless sensor networks: Methods, models, and classifications. ACM Comput. Surv., 39(3): 9.

[16] OGC. Geoserver. 2010. http://geoserver.org, accessed in August 2010.

[17] OGC. Sensor Web Enablement.
http://www.opengeospatial.org/projects/groups/sensorweb, accessed in August 2010.

[18] PANTAZIS NA, NIKOLIDAKIS SA & VERGADOS DD. 2009. Energy-efficient routing protocols in wireless sensor networks for health communication systems. In PETRA '09: Proceedings of the 2nd International Conference on PErvsive Technologies Related to Assistive Environments (New York, NY, USA, 2009), ACM, pp. 1–8.

[19] PASTORELLO JR GZ, MEDEIROS CB & SANTANCHÈ A. 2007. Providing homogeneous access for sensor data management. Technical Report IC-07-012, Institute of Computing, State University of Campinas, May.

[20] REED C, BOTTS M, DAVIDSON J & PERCIVALL G. 2007. Ogc-sensor web enablement: overview and high level architecture. In Autotestcon, 2007 IEEE (17-20 2007), pp. 372–380.

[21] SANTANCHÈ A & MEDEIROS CB. 2007. A component model and an infrastructure for the fluid web. IEEE Transactions on Knowledge and Data Engineering, 19(2): 324–341 (February 2007).

[22] SZLAVECZ K, TERZIS A, OZER S, MUSALOIU-E R, COGAN J, SMALL S, OZER S, BURNS R, GRAY J & SZALAY AS. 2006. Life Under Your Feet: An End-to-End Soil Ecology Sensor Network, Database, Web Server, and Analysis Service. In 3rd Workshop on Embedded Networked Sensors (EmNets 2006) (May 2006), pp. 51–55.

[23] TAVARES AL & VALENTE MT. 2008. A gentle introduction to osgi. SIGSOFT Softw. Eng. Notes, 33(5): 1–5.

[24] TRUST TWS. The web science trust.
http://webscience.org/home.html, accessed in August 2010.

[25] VAQUERO LM, RODERO-MERINO L, CACERES J & LINDNER M. A break in the clouds. ACM SIGCOMM Computer Communication Review, 39(1): 50 (Dec. 2008).

[26] ZHAO Q & GURUSAMY M. 2008. Lifetime maximization for connected target coverage in wireless sensor networks. IEEE/ACM Trans. Netw., 16(6): 1378–1391.