Design and Performance Evaluation of a System for Storing and Visualizing Data from a Sensor Network

Wassim Drira, Éric Renault, and Djamal Zeghlache

Abstract—This paper presents architecture design and performance evaluation of a back end system used to store, index, manage and visualize water quality measurements collected by sensor networks deployed in rivers, lakes and coastal regions. The embedded communications systems in each sensor node enable ad hoc network operation to relay the measurements to the back end system in charge of managing, processing and providing data to end users and large communities through web interfaces. The focus is on the back end system architecture description and on the evaluation of its performance.

Index Terms—WSN, environment monitoring, data management, data collection.

I. INTRODUCTION

The soil, the air, the water have dramatically changed since the beginning of the industrial revolution at the end of the 19th century and some signals let environmental specialists understand that the climate may have evolved to some critical states. As a result, for some years, the environment and its quality have become a major concern for more and more people around the globe. People want to be informed about the quality of the environment they are living in. Politics, media, organizations in charge of the management of cities and/or natural areas are willing to have tools that provide accurate information in an almost real-time manner in order to make the good decisions for the future, react in the most appropriate way in case of problems and report the best information to the public.

In this scope, this paper describe a system architecture that has been designed and implemented for gathering, intercepting, storing, transforming, indexing and visualizing the measurements collected from sensor networks. This system has been used in real world experiments and was integrated in the Mobesens project [1].

The remaining of this paper provides in section II an overview of the related work, followed by a description of our proposed system architecture in Section III. Section IV describes the Mobesens project where our system was successfully integrated and tested. A real world experimentation is the subject of Section V. Section VI encloses the performance results that been measured during some real experimentations. We wrap this paper with a conclusion in Section VII.

II. RELATED WORK

Wireless Sensor Networks (WSN) are widely used, in different domains and different places, from border control to satellites. During the past years, the research community on sensor networks has mainly focused on the establishment of efficient and scalable networks by enhancing routing [2], addressing [3] and aggregation [4] mechanisms in WSN. However, issues like gathering, intercepting, storing, transforming and indexing of measurements collected from sensor networks have not been well treated, causing the lack of a standard middleware. Two approaches are used in designing such an application:

- **Data collection view** where the application considers the sensor network as a distributed database. The two best-known solutions for this approach are Cougar [5] and tinyDB [6]. This approach is not suited for resource constrained networks as it generates an important communication overhead.

- **Application specific view** where the application receives collected data from the sensor network through a gateway to store and visualize it. Its benefits are numerous like the ability for long term data storage and indexing, high processing power [7], etc. This approach was selected in many environment monitoring works [8]-[14]. We choose it also to develop our middleware system.

Environment monitoring has been the concern of many research works. The GlacsWeb project [12] monitors the behavior of ice caps and glaciers for understanding the earth’s climate. Collected data from probes (inserted in the glacier) are relayed through Base Stations to a Sensor Network Server where it becomes accessible using web services. In [15], authors proposed an air pollution monitoring system involving a context model for understanding the air pollution status on the remote place. The Integrated Sensor Web Grid Cyberimplementation [14] has two goals. The first one is to demonstrate the integration features of heterogeneous environmental wireless sensor networks. The second one is to present a sensor web grid Cyberimplementation as a server hiding the system heterogeneity.

There are also some other projects which deal with monitoring water quality. In [9], authors proposed a Remote Water Quality Monitoring System based on WSN. In their system, sensors send measurements data via a coordinator to a control center where they are stored in an SQL database and shared with other users. Jin et al. [13] proposed a novel architecture which collects data, store them in a database and
display them on a real-time manner using a C# .net software. However, this obliges users to install this software on their computers.

Han et al. [16] proposed an architecture using AJAX in Environment Monitoring System based on WSN to enable real-time monitoring. In order to provide a more efficient system, we use AJAX Push technology [17] also known as Reverse AJAX, which gives the ability for the web server to send only updated measurements to the users as they become available.

In this article we present a new system architecture to store, index and visualize sensor data gathered from WSN and it has been tested on the context of monitoring water quality and network status using RSSI indicator in a real-time manner. The storage is done on a cloud storage file system and the indexation on a no-SQL database in order to make the system more flexible and able to be extended easily. In fact, during the project we use frequently new sensors. Thanks to our system, we do not need to design new database tables or to adjust our storage solution. All we need is a driver to transform raw data to XML.

In addition, using the results of other research works [18], [19] can help other WSNs to be compatible with our system architecture without using a driver.

Moreover, we use web services and XML to exchange data measurements. This makes the system easily integrated into other projects like Sensor Web Grid Cyberimplementation [14] and the SWE (Sensor Web Enablement) [20] project which aims at enabling interoperability between traditionally disparate community sensor networks.

In this article we present a new system architecture to store, index and visualize sensor data gathered from WSN and it has been tested on the context of monitoring water quality and network status using RSSI indicator in a real-time manner. The storage is done on a cloud storage file system and the indexation on a no-SQL database in order to make the system more flexible and able to be extended easily. In fact, during the project we use frequently new sensors. Thanks to our system, we do not need to design new database tables or to adjust our storage solution. All we need is a driver to transform raw data to XML.

In addition, using the results of other research works [18], [19] can help other WSNs to be compatible with our system architecture without using a driver.

Moreover, we use web services and XML to exchange data measurements. This makes the system easily integrated into other projects like Sensor Web Grid Cyberimplementation [14] and the SWE (Sensor Web Enablement) [20] project which aims at enabling interoperability between traditionally disparate community sensor networks.
The XML version of data is indexed in the indexation space.
The XML version of data is encapsulated in a message and sent to the ActiveMQ eventing server.
The gateway is returned a unique ID.

In order to guarantee the interoperability between the gateway, the grid and third-party applications, a HTTP Rest interface has been developed. It allows the gateway to send data frames to the grid and third-party applications to interact with stored and indexed data in the grid.

This interface is generic enough to adapt to new sensors and nodes that may join the network. For example, a node identified by NodeID can send a frame to the grid by encapsulating it into a PUT request to http://{ServerAddress}/{SensorType}/NodeID where SensorType is a unique identifier associated with each sensor type and a unique identifier (UID) is returned as a response to the node. This UID can later be used to retrieve the same data from the grid using a GET request: http://{ServerAddress}/UID.

Data stored in the storage space are persistent. This means that unless they are intentionally removed, data remains in the system regardless the application that stored them is still alive or not. As a result, in order to navigate through data stored and indexed in the system without necessarily knowing their identification, a dedicated interface has been made available. This interface receives a query from a user or an application using the XQuery language and returns the list of UID's that matches the given set of criteria.

C. Real-Time Display at the End-User Side

Apart from getting a specific piece of data using its UID or searching for a set of data matching a set of criteria, the system architecture we developed allows to display the most up-to-date data in a real-time manner. This has been made possible with the introduction of an eventing system based on the combination of a JMS Server [24] and ICEFaces [25]. This solution allows data to be visualized on the graphical user interface without refresh notification required by the user or client. Moreover, the time needed to update the graphical user interface is very low compared to the frequency of the measurements. This is a key feature since even if data are not updated in a real-time manner in the graphical user interface; the data update is performed quickly enough so that information provided to the end-user is always up-to-date.

Also note that all clients are synchronized and updates happen concurrently towards all clients. Changes in sensed or stored data trigger automatically updates in all the clients according to their subscriptions and expressed interests. Then, when a client induces a change on a piece of data, it is also immediately reflected to the entire user community.

Fig. 3 presents the architectural details for the implementation of the real-time display. The two main elements are the JMS Server which aims at dealing with any events in the system and ICEFaces that implements an Ajax PUSH link to any registered clients. At initialization, the JMS Server initiates a persistent communication with ICEFaces. Then, any time a piece of data is received by the system, the XML version of the data is sent to the JMS Server that checks whether the data matches any events. For any matching events, the data is forwarded to ICEFaces that also forwards the data to the registered clients.

The connection between the system and the clients for the real-time display is performed using the publish/subscribe paradigm, i.e. it is the responsibility of the client to subscribe to the system to receive up-to-date data. In order to do so, the client needs to subscribe to a topic in the ActiveMQ eventing server. After the subscription, the client receives a message every time a new frame matching the selected sensor type is received by the system.

IV. THE MOBESENS PROJECT

The Mobesens project [1], which stands for Mobility for Long Term Quality Water Monitoring, aims at developing sensors to measure different physico-chemical parameters of water so as to appreciate its quality in the environment, and set up both a communication network and a grid infrastructure to allow data to be first transmitted from sensors to the core network and second to be stored, processed and displayed in a convenient and easy-to-use graphical user interface. Note that, even though the project is limited to measure and report water quality parameters, the presented work may be adapted to any other kind of environments, like the soil, the air, etc.

In this project, we deployed many wireless sensor networks in different areas: Lake Leman, Brest, Thau Lagoon and Ebro River. Each network is composed of CSEM sensor nodes [26] protocols. Each node is connected to one/many physico-chemical sensor via an RS485 interface. Nodes can be mobile or static. In order to allow data to be collected and transferred for processing, a gateway with broader communication capabilities is placed close to the wireless sensor network. This gateway may either be static or mobile, located on the sea shore or in/on the water. Then, data are transferred to the indexing, storing and eventing system both for persistent storage and to allow more complex processing. Data stored in the system are available through the Internet. A visualization interface is provided to help end-users to display data for monitoring, analysis, etc. We choose to develop a web-based visualization interface as it gives basic mechanisms to allow various research groups to work together. The system architecture described in section III is integrated with the other elements of this project.

V. REAL WORLD EXPERIMENTS

The architecture presented in this paper has effectively
been developed and is functional. It is possible to see some partial results at this location: http://www.mobesens-vs.eu/.

The system is located in Évry, close to Paris, France. The storage space and the indexation space are running on a cluster composed of 32 nodes each including an 8-core Intel Xeon E5540 cluster with 24 GB RAM. The visualization interface has been developed using a Tomcat web application server. After the entire system has been developed, some real experiments have been conducted on the Lake Leman, Switzerland and on Brest, France. The third experiment was conducted on the Ebro River, Spain. The configuration included in the latter was composed of 20 buoys, each one equipped with sensor nodes and ISFET (Ion-Sensitive Field-Effect Transistor) sensors to measure temperature and pH, a kayak equipped with a multi-parameter sensor to realize profiles at different depths, and a sensor node to forward data to the Gateway which is responsible on collecting measured data and sending them to the GRID in Évry through the 3G network.

During and after deployment, it has been possible to validate automatically received data (physico-chemical measurements) and visualize them in a real-time manner using any web browser in smartphone, tablet or laptop. In addition, using the real-time interface, it was possible to monitor the network, sensor measurements and calibration under a boat in the sea. This has allowed us to detect and resolve problems quickly during and after deployment.

VI. PERFORMANCE RESULTS

Various experiments have been performed in order to check the performance of this work and especially the ability for the system to provide data in a real-time manner.

In the first test, we stressed our system in order to study how it would behave. The Gateway is emulated using a laptop which regularly sends GPS position frames of different nodes to the system. In the same laptop, a browser is opened for the visualization interface to display sending and receiving data times. We used the same laptop to be able to measure the difference between sending and receiving data times using the same clock. For each experiment, each node sends one hundred messages, e.g. for the experiment with eight nodes, 800 messages have been sent to the system. Then, the mean and the standard deviation of the end-to-end delay between sending and receiving frames are computed. Results are presented in Fig. 4. It shows that the end-to-end delay is low (less than 500 ms can be considered as real-time for water monitoring as water physico-chemical parameters are changing slowly in the natural environment) and stable.

In the second test, message size varies. The gateway sends one hundred message of each size to the system. Fig. 5 shows the mean and the standard deviation of the delay between sending the message and receiving the UID, e.g. ID_mapping in Fig. 2, for each message size in a logarithmic scale. The default payload length in TinyOS is 29 bytes, so putting a node message in the system takes less than 100 ms which is reasonable compared to the communication delay between the gateway and nodes in the WSN.

In the third test, the gateway launches some threads at the same time, to send a 40 bytes message to the system. Threads emulate nodes that are attempting to send a message at the same instant to the system. This experiment is performed one hundred times for each number of threads. Fig. 6 displays two curves: the blue one shows the mean and the standard deviation of the delay between sending the message and receiving the identifier in each thread, and the red one shows the mean and the standard deviation of the execution in the system, e.g. functions represented in Fig. 2. The results show that for example if 64 nodes start communication with the system in the same instant, each communication will end after a mean of 450 ms and the execution of storing, indexing and transforming functions in the system takes 320 ms. So after a mean of 450 ms all communications are ended which highlight the system performance.


VII. CONCLUSION

This paper presented a system architecture we developed to store, index and visualize any data sent by sensor nodes in a real-time manner. Our system has been used in the scope of the MOBESENS project and tested on real world experiments. It has shown its usefulness to remotely monitor water quality using WSN. The performance measurements show that the end-to-end delay, i.e. the elapsed time between the time the data are issued by a sensor and the time they are effectively displayed in the browser of the end-user, satisfies the real-time requirement.

REFERENCES


Vassim Dirra received the Ph.D. degree in computer science from the University of Pierre and Marie Curie (Paris 6) and the Institut Mines-Télécom/ Télécom SudParis, in 2012. His research interests are focused on wireless sensor network security and data management.

Éric Renault has been an associate professor at Institut Mines-Télécom, Telecom SudParis, Évry, France, since late 2001. His research interests include high-performance computing and messaging, RFID systems and security. In 1995, he received a MSc in Computer Engineering and a MSc in Computer Science, and a PhD in Parallel Computing in 2000 all from the University of Versailles—Saint-Quentin-en-Yvelines, France, where he also served as an Assistant Professor. In 2001, he was a Research Associate at Dartmouth College, NH.

Djmaou Zeghlache graduated from SMU in Dallas, Texas in 1987 with a Ph.D. in Electrical Engineering and joined the same year Cleveland State University as an assistant professor. In 1990 and 1991 he worked with the NASA Lewis Research Center on mobile satellite terminals, systems and applications. In 1992 he joined the Networks and Services Department at Telecom SudParis of Institut Telecom where he currently acts as Professor and Head of the Wireless Networks and Multimedia Services Department. Professor Zeghlache is also acting Dean of Research of Telecom SudParis. He co-authored around one hundred publications in ranked international conferences and journals and was an editor for IEEE Transactions on Wireless. His interests and research activities span a broad spectrum related to fixed and wireless networks and services. The current focus is on network architectures, protocols and interfaces to ensure smooth evolution towards loosely coupled future Internet, cloud networking and cloud architectures.