Development of a web-based interface for remote monitoring long-distance power transmission overhead lines

Sibukele Gumbo, Hyppolite N. Muyingi
Department of Computer Science, University of Fort Hare
Email: sgumbo@ufh.ac.za, hmuyingi@ufh.ac.za

Abstract— **The system described in this paper aims at acquiring, storing and displaying data collected from a wireless sensor network. Various open source tools have been used in the development and implementation of the system for remote monitoring and reporting temporal changes of a long distance power transmission line.**

***Keywords*— application software, Enterprise Java Beans, metadata, open source, PostgreSQL, query, TinyOS, wireless sensor networks.**

I. Introduction

This paper describes the development and implementation of a web-based user interface for gathering and integrating sensor network data to enable real time visualization, control and interpretation of the condition of a structure. Through active monitoring, early detection and instant alerting, failures in systems such as long distance power transmission overhead lines, whose cables and devices are most located remotely, can be reported quickly to re-establish services. To monitor power line systems, the wireless nodes are typically placed in a straight line along a long and narrow area, with a base station at the end where further processing and control occurs, thereby creating a many-to-one communication model [6] [7].

Currently, power line communication (PLC) and aerial optical fiber cable are being used for monitoring of transmission systems. The limitations include the cost and complexity of wiring for communication of the fibre cables, limited reliability of power line-based telecommunication medium and low scalability of these fixed data acquisition systems. The Power Electric Utility Industry is trying to adopt emerging technology in the management of their systems. It becomes necessary to investigate the applicability of a new generation of wireless sensor nodes in measuring quantities in power lines for remote monitoring and distributed control.

II. The Wireless Sensor Data Acquisition System

A wireless data acquisition system consists of a computer, a base station and wireless sensor node platforms. For our work, Mica sensor nodes were chosen for conducting experiments as they operate on an open source platform thus programs can be developed and deployed to suit the application. Mica nodes also have very low power consumption, but other methods of powering the sensors may need to be implemented e.g. energy harvesting when applied in applications where it is not feasible to change the batteries such as power line monitoring [1].

A. *Hardware platform architecture for a Mica sensor node*

A typical wireless sensor node has features shown in Figure. 1. All these features are present in the Mica nodes which are/ used to perform experiments. These nodes have sensor boards that may measure voltage, moisture, temperature values, electric field or magnetic field, parameters that may need to be monitored in overhead power line systems [6][1].

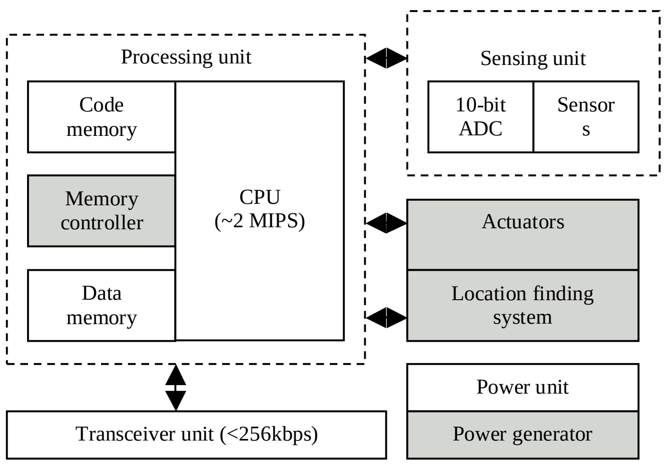


Figure 1. Hardware architecture for a sensor node adapted from [5]

B. The wireless sensor network protocol stack

The Institute of Electrical and Electronics Engineers (IEEE) has defined a standard for low-power, low-data-rate wireless data transmission called 802.15.4 [2]. No unified protocol stack for wireless sensor network exists for this standard, but wireless sensor nodes can be built to have a collection of known protocol functions which resemble Figure 2.

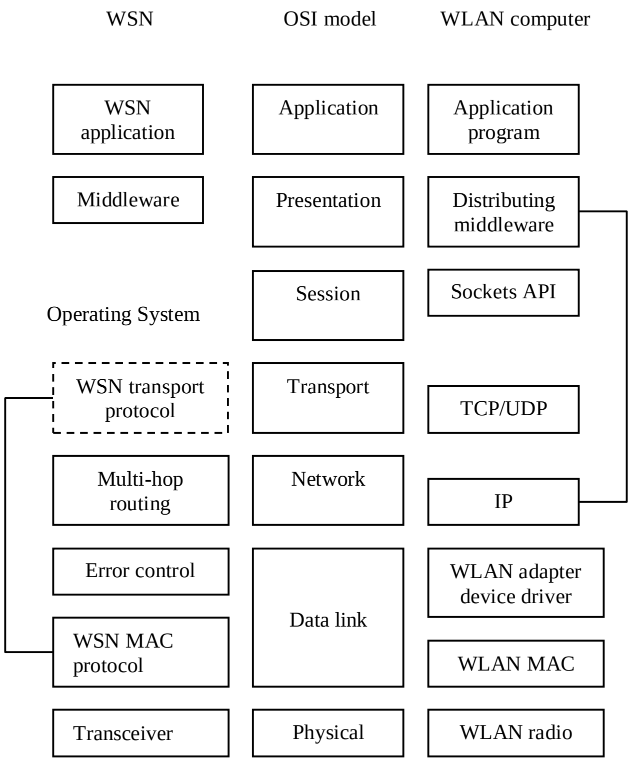


Figure 2. WSN, OSI model and WLAN protocol stacks from [5].

C. Wireless sensor network operating system and software

The Mica node wireless sensor platform, TinyOS, is an open source platform developed at the University of California-Berkeley and commercialized by Crossbow [1]. TinyOS includes development tools, network protocols, sensor drivers, data acquisition tools, and source code and utility programs. This operating system is optimized for low powered processors with RAM in the order of a few kilobytes (currently 256 bytes and 4kB of ROM) [4]. The small amount of on board RAM does not allow large quantities of sensor data to be stored. As a result, only data interrogation algorithms are embedded in the sensor node processing unit for limited data processing. Data collected by the node is therefore propagated to the sink node using message passing and stored in the database as shown in Figure 3. The nodes should be able to package and deliver the data in the correct format which is required by the database. The data collected is extremely useful for evaluating long term sensor network deployments.

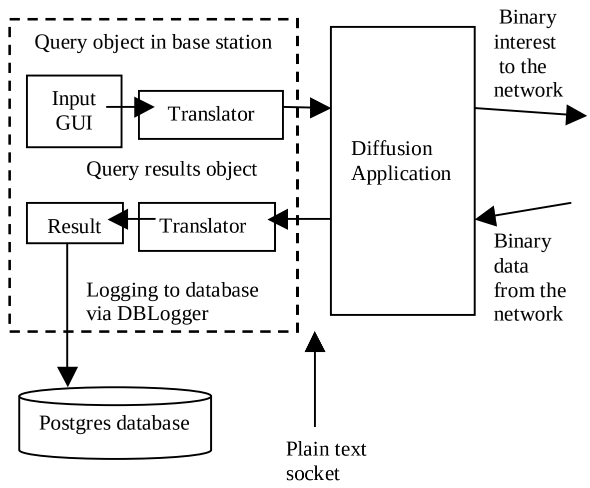


Figure 3. Reference model of the architecture of the base station software

TinyDB is an open source overlay application for extracting information from a network of TinyOS sensors. A network topology manager exists in this application that manages the connectivity among motes by maintaining the node routing tables and tracking neighbours using the radio network. The placement of nodes in power line monitoring is deterministic hence data is routed through predetermined paths. To use TinyDB, no configuration or programming is required on the motes, only its TinyOS components should be manually installed on each mote in the network. TinyDB provides a facility to develop and deploy standalone data-driven applications in a sensor network using SQL queries and a Java Application Programming Interface (API) [9] [10]. The developed applications contain queries to extract interesting features or event information from the database.

A network of TinyDB motes are resource-constrained devices, hence an application customized for remote sensing was built to allow the system to remain in sleep mode until the external condition warrants re-awakening in order to lower power consumption [3]. Data is collected from the sensor network after the execution of this single application on the client interface of all nodes simultaneously. This data is put into a database for future mining and control capabilities. DBLogger is able to create tables, whose primary key is the motes unique identification number and hold results when instantiated with a query [4] [10].

III. WIRELESS SENSOR NETWORK APPLICATION

The wireless sensor network application was developed to allow the data files that have been stored in the database to undergo more advanced post processing. This software provides interfaces for configuring and monitoring the wireless sensor network, its parameters, node characteristics and locations and other relevant system information such as checking threshold limit states and power line damage. The application software for data access to the wireless sensor network data consists of a web-based interface, a metadata schema and a data transaction, access and storage system as shown in Figure. 4. This interface is a multi-tier Java 2 Enterprise Edition (J2EE) architecture. It consists of four layers: presentation layer, transaction layer, access layer and storage layer, which collectively serve the web interface with live data. Open source platforms have been used in the development of the software because they require no special licensing to use for research or commercial purposes, they adhere to standards, and are easy to use in the deployment of any system. The system is built using Java; hence it can run on any operating system that supports the Java runtime environment.

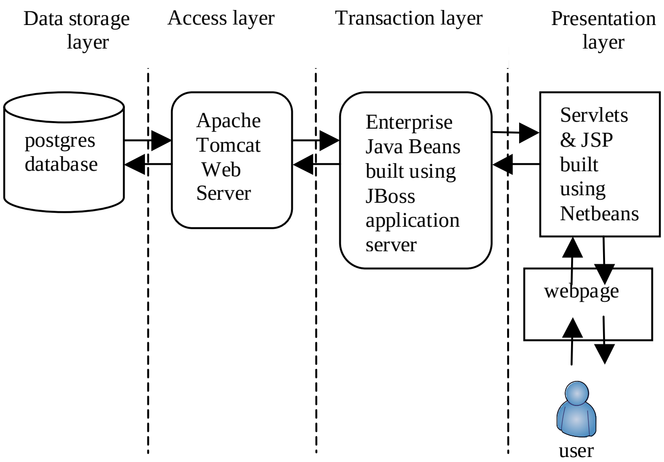


Figure 4. Reference model for wireless sensor network application

A. *Presentation layer*

The presentation layer consists of a static webpage and dynamic data requested by the user, from the database. It provides a visual interface to present data from the sensor network to enhance data comprehension. This facility is developed using Java Server Pages (JSP) and servlets, which are built and tested using Netbeans IDE. The use of JSPs makes the system more flexible and easily customizable to suit the specific needs of the user while keeping the dynamic data intact. Access to the webpage and its information is controlled by a password authentication system. The basic pages that have been developed are summarized in Table 1.

Table. 1 Basic pages

File name	**Description**
Alerts	Triggered alerts
sensorDetails	Individual sensor details
allSensors	List of all sensors with status and locations
History	Generated history graphs
Login	User authentication

The web-based interface was chosen as a tool to deliver information because web browsers are familiar to computer users and no additional software must be installed in the user's computer, since web browsers are a standard application in most systems.

Simplicity is required in this interface for efficient retrieval and visualization of data relevant to assessing structural states. Thus, a set of simple alert icons were incorporated alongside the data retrieved from the database into the webpage to depict the status of the sensors on the field. In order to achieve this, various threshold limits were set for all the sensors readings as an indicator of the degree of damage to the structures being monitored. This allows information to be read quickly by simply looking at an icon and recognizing a color. Two types of sensor alerts are used. One type deals with the threshold values on the individual sensors where sensor states (off, critical, normal or warning) are represented by a gray, red, green and yellow colored icon alerts respectively. The other type of alert depicts the status of the power line structure as a whole. This status can be computed by analysis classes in Java that perform calculations on the recorded data sets using the threshold values to determine the status of the sensors.

Interest mostly lies in the most recent measurement from each sensor in the network. As a result, any undelivered measurements are ignored. Mica sensors transmit measurements with the state of the internal counter (timestamp) to the base station to select the most recent data entry from each distinct sensor node.

Time history graphs, which provide a tool to visualize changes in the power line structures over time, are displayed on the web pages. These can be built and deployed using JFreeChart, a Java class library, because of its ability to run in a servlet environment, hence allowing it to be plugged into the web based interface. A customized Java applet showing the historical graphs is also an option for the display of historical data and will be as a future extension of the work described in this paper.

B. *Transaction layer*

The transaction layer provides the object-level interface, or index, to retrieve the data and metadata from data storage to support the webpage application. This is achieved by the use of Enterprise Java Beans within the open source JBoss Application Server. Metadata is all the information about the monitoring system that includes the sensors and the type of data that they collect, e.g. the sensor types, sensor alert types and location of sensors. Each of these data types are then grouped into classes which are then represented by Java beans. Each bean contains the logic for retrieving its data from the database. As a result, some abstraction between the webpage and the database exist, making it possible to make remote method calls for object-level interaction between the webpage and a particular class.

When a user clicks on a web page through a web browser, a corresponding Java object in the servlet is used to process the user's request. The servlet then calls the Enterprise Java Beans to locate the corresponding information from the database and return in to the user through a processed JSP. Metadata information that has been identified for this application is shown in Table. 2.

Table 2. Metadata classes

Class	Examples	Description
Sensor	Temperature sensor, Voltage sensor	Describes all sensor types used and their characteristics including each of their threshold values that trigger alerts
Data	Temperature , Magnetic field, Electric field	Data collected from the overhead power line
Location	Sensor node coordinates	Describes all sensor spatial information
Alert	Critical, Warning, Normal.	Different limits that indicate change in the power line status

C. Data access layer

The data access layer provides a container that controls access to data in the repository and manages the retrieval of files over the network. Apache Tomcat is the servlet container that is used for the servlets, Java Server Pages and Java beans technologies. It manages and invokes servlets when they are requested by a Web browser or by another servlet. The physical later portion of this layer implements the access to the database.

D. Data storage system

The storage layer maintains the actual data files and the underlying database that provides the metadata information to provide live data to the user interface. The database provides a method of organizing data collected from the sensor network into one place, using a common format. This database has a double "interface" because it interacts with the sensor nodes and the visualization interface. The properties of the wireless link connections between the sensor nodes and the effects of interference and packet loss on the quality of the collected data can be evaluated by the amount of data collected in the database. Any undelivered measurements are ignored since we do not need all messages and interest lies only with the most recent measurements. The data acquisition system linked to the database should be able to manage the raw data files and place them in an appropriate location. PostgreSQL is an open source, object-relational database management system, which supports the logging of TinyDB queries [4] [8]. The schema of the sensor data in the database is as follows:

result time : timestamp	epoch : int	nodeid : int	temp : int	voltage : int

In order for a client process to access a database, it needs to connect to a running "postmaster", which is a multiuser database server which allows many connections to the database. PostgreSQL is implemented using a 'process per user' client server model, where one client is exactly connected to one server process ("postgres") via postmaster as show in Figure 5. A C application programmer's interface, libpq, allows the client programs to pass queries to the PostgreSQL backend server and to receive the results of these queries as shown in Figure 5. Several independent implementations of libpq exist, such as the Java JDBC driver which is used for all connects to the database in this project [4] [8].

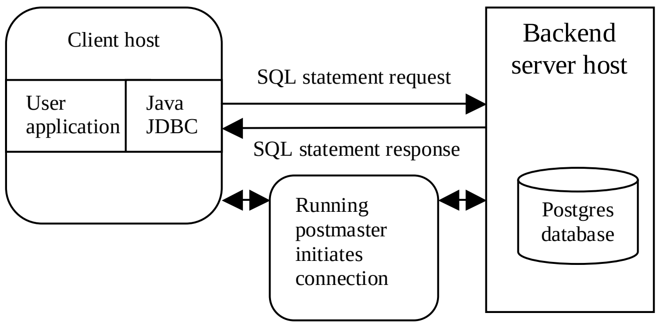


Figure 5. Communication between the client host and the database

IV. EXPERIMENTS AND RESULTS

The web based application takes advantage of the values from temperature sensor boards that are to be used to measure the variable power line quantity. As latency is a concern in wireless networks, six Mica2 nodes were used to examine the latency of data delivery in a linear aligned sensor network as shown in Figure 6, since the project objectives include building a spatially scalable, real-time power line monitoring system. Sensor nodes are resource constrained devices; thus the after analysis of the results from the experiment, nodes in the project where configured to sleep and adaptively wake up to listen to any traffic in the network thereby saving energy.

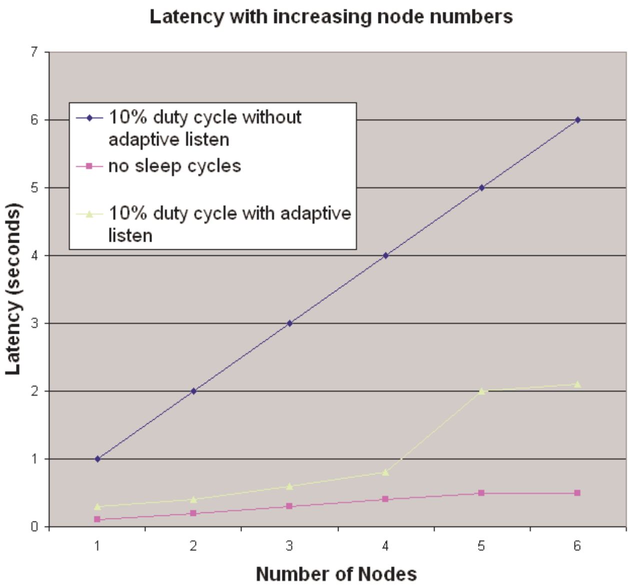
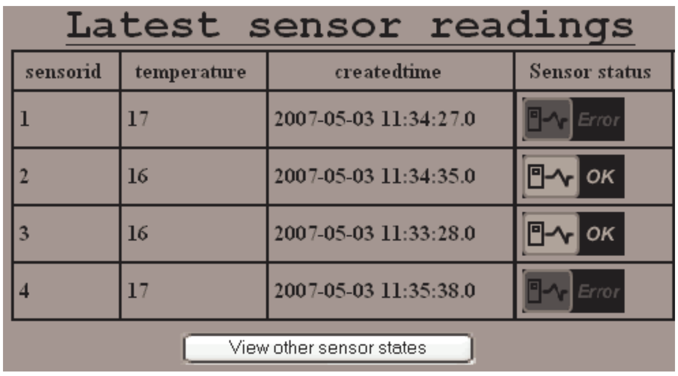


Figure 6. Latency in a linear aligned sensor network

Java classes within the servlets are used to select data values from the database. Depending on the thresholds set by an administrative user, different temperature ranges are depicted using different icons on the web interface as shown in Figure 7, simplifying data analysis.



Latest sensor readings

sensorid	temperature	createdtime	Sensor status
1	17	2007-05-03 11:34:27.0	Error
2	16	2007-05-03 11:34:35.0	OK
3	16	2007-05-03 11:33:28.0	OK
4	17	2007-05-03 11:35:38.0	Error

View other sensor states

Figure 7. Selection of the latest sensor readings by a user

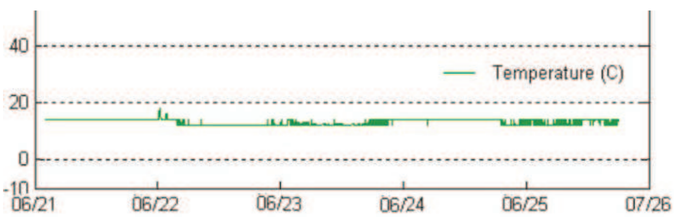


Figure 8. Historical sensor network data graph

Sensor node data is also represented with a line graph drawn using values stored in the database using JFreeChart libraries. Figure 8 shows the average temperature values of a linear aligned network with nodes separated by a distance of 100 metres to test network connectivity, hence web application feasibility to display wireless sensor node values.

V. CONCLUSION

This paper describes a system we have built to acquire, store and display data collected from a large number of linear sensors, such as ones to be found on overhead power lines. This system is a multi-tier Java 2 Enterprise Edition (J2EE) architecture built on open source platforms; hence it can run on any operating system that supports the Java runtime environment.

ACKNOWLEDGEMENTS

I would like to thank Alfredo Terzoli for his help in writing this paper. I would also like to thank the University of Fort Hare Telkom Centre of Excellence in developmental e-commerce, co-sponsored by Telkom SA, Thrip, Tellabs, Saab Grintek, Amatole Communication Services, Mars Technologies and Drisa for their generosity in funding for my Masters studies. I finally would also like to thank the WESTCOR project for initiating the project topic my research work.

REFERENCES

[1] Crossbow website. www.xbow.com

[2] J. A. Gutierrez, M. Naeve, E. Callaway, V. Mitter, B Heile, “IEEE 802.15.4: a developing standard for low-power low-cost wireless personal area networks,” IEEE Network, Volume 15, Issue 5, Sept-Oct 2001, pages 12-19.

[3] J.L. Hill, “System architecture for Wireless Sensor Networks,” Ph.D dissertation, Dept. of Computer Science, University of California, Berkeley, 2003.

[4] M. Bramwell, “Implementing a Mica2 MOTE Sensor Network,” in July 2006.[online].Available: http://hamster.foxhollow.ca/TinyOS/Guide/Mica2-Install-Guide.pdf

[5] M. Kuorilehto, M. Hannikainen, T.D. Hamalainen, “A survey of Application Distribution in Wireless Sensor Networks”, EURASIP Journal on Wireless Communications and Networking 2005:5, 774-788

[6] M. Nordman, “Architecture for Wireless Sensors in Distributed Management of Electrical Distribution Systems,” Ph.D dissertation, Dept. Elect. Eng., Helsinki University of Technology, Finland, 2004.

[7] P. Cheng, C. Chuah, X. Liu, “Energy-aware Node Placement in Wireless Sensor Networks”. Available: http://networks.cs.ucdavis.edu/publications/2004_liu_2004-12-09_10_13_46.pdf.

[8] PostgreSQL website. www.postgresql.org

[9] S. Madden, J. Hellerstein, W. Hong “TinyDB:In-Network Query Processing in TinyOS,” in September 2003.[online].Available: http://telegraph.cs.berkeley.edu/tinydb/tinydb.pdf

[10] TinyDB website. http://telegraph.cs.berkeley.edu/tinydb/

AUTHORS

Sibukele GUMBO is currently a Masters student at the University of Fort Hare. Her research interests include wireless networks and Java-based software development

Hyppolite MUYINGI, PhD. (EEEng) VUB, Belgium is a Professor in the Department of Computer Science at the University of Fort Hare. His main areas of academic interest are ICT for development, networks, and communications technologies for power utility industry. Professor Muyingi is the Head of Fort Hare Centre of Excellence.