Wireless sensor networks for environmental data collection and events detection

A. A. Poyda[1], M. N. Zhizhin[2], and A. V. Andreev[2]

Received 23 November 2012; accepted 18 December 2012; published 25 December 2012.

The article presents the technology for collecting low-frequency and event detecting in high-frequency environmental data streams using wireless sensor networks, a developed set of sensors for different sampling rates, detectors for high-frequency events, and software to convert, store and visualize the data. *KEYWORDS: Wireless sensor networks; event detecting; environmental monitoring.*

Citation: Poyda, A. A., M. N. Zhizhin, and A. V. Andreev (2012), Wireless sensor networks for environmental data collection and events detection, *Russ. J. Earth. Sci., 12,* ES6002, doi:10.2205/2012ES000525.

1. Introduction

There are two opposite approaches in the technology for collecting environmental data. The first can be realized by autonomous recorders of physical quantities (loggers): devices are installed at a required point to scan parameters at a required frequency, keeping them in a local storage. In order to obtain data from such device, a direct contact has to be established with it. This is inconvenient, if the device is installed in a remote place (for example, trees, wells etc.), and can be time consuming, if several dozen devices are installed at the studied area. The second approach is to use a fixed wire or wireless data collection network, capable of transmitting large amounts of data in real time. The disadvantage of this approach is high energy consumption, the need for laying cables or radio paths, regular maintenance of units, and the vulnerability of the entire network in the event of failure of communication channels. All this leads to a substantial increase in the cost and time of deployment.

This paper presents a compromise technology, which is a rapidly deployable wireless network consisting of a set of almost autonomous recorders (wireless sensor network, WSN). The nodes of the network do not require maintenance and replacement of power supply for months, even years, but at the same time collecting data from all nodes can be performed remotely from different places and at any time. The basis of this technology is weather-sealed intellectual sensors (motes) with a built-in microprocessor, connected by a digital radio channel with a decentralized peer to peer (P2P) communication protocol (IEEE 802.15.4 ZigBee), tolerant to faults of individual transponders and equipped with software, optimized for data collection and event detection directly on the network nodes.

Such network can be used for rapid deployment (for example, deploying sensors by UAV) in remote places, followed by collecting data by using radio without having to crawl to all sites, for example, on a volcano, in a geothermal system, in a forest on fire or for environmental monitoring, in a mine etc.

Various geophysical parameters have different characteristic sampling rates. For example, data on deformations, magnetic field or air temperature change relatively slowly, so they can be scanned with periods of days-hours-minutes, while seismic tremors and infrasound are observed at frequencies of few hundreds of Hertz. The technology of wireless sensor networks can be used for data collection and detection of events in long-period (interval day–minute), and in high-frequency (dozens of kHz) time series. The article presents a set of intelligent sensors to observe basic geophysical parameters at different frequencies. For high-frequency observations all data transmission in real-time is limited by a network bandwidth. Therefore, for high-frequency observations we proposed a set of detectors to help you analyze data directly at the network nodes, with the ability to preserve and transmit to data collection center only a summary of events.

2. Architecture of Intelligent Sensor

We used the TelosB platform produced by the American company Crossbow as a prototype intelligent sensor (Figure 1). The 16-bit Texas Instruments MSP430 microprocessor with 8 MHz clock frequency and 0.7 mA power consumption in idle mode and 200 mA in active mode, 2.4 MHz radio link, 512 KB operative memory, USB port for PC connection and a basic set of sensors of voltage supply, temperature, and relative humidity, and illumination in the vis-

[1]Research Center "Kurchatov Institute", Moscow, Russia
[2]Federal State Institution Geophysical Center of the Russian Academy of Sciences, Moscow, Russia

http://elpub.wdcb.ru/journals/rjes/doi/2012ES000525.html

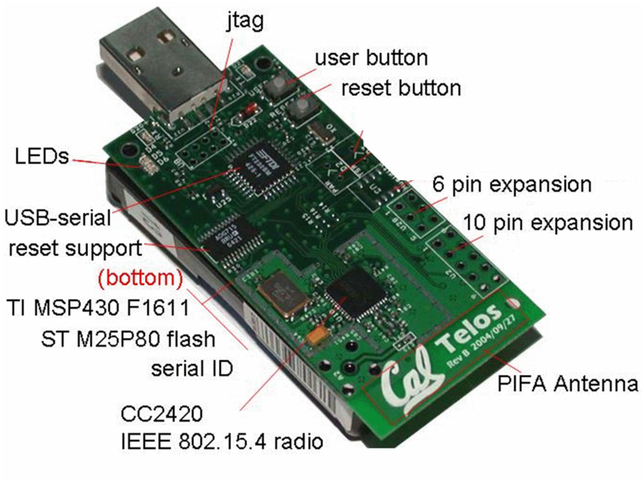


Figure 1. Main components of TelosB wireless sensor network module.

ible and near infrared spectrum are installed on the sensor's motherboard. The microprocessor has a built-in 12-bit ADC and a few digital inputs for external sensors. More detailed technical parameters of the node are given in the manufacturer's specification (Technical specification Crossbow, "TelosB datasheet" component code: 6020-0094-01. Rev B). The battery life of the sensor on two AA batteries depends on the duration and power of the radio and is a few weeks, on average. However, in the case of a sensor network to collect low-frequency observations with buffered data at the nodes, it can provide relief operation, preserving battery power for longer periods of time.

TelosB platform modules use network protocol Zigbee [*Farahani*, 2008] which provides energy-efficient multi-hop radio communication of short messages between WSN nodes with dynamic routing, choosing methods of message delivery and taking into account the performance of intermediate nodes. Distance between nodes is highly dependent on the conditions of their placement (underground, in trees, in a room), but the average distance is about 10 m. Using an external antenna, the communication range can be increased up to 50 m. Therefore, monitoring of extended objects requires either a large number of nodes that transmit data "along a chain", or intermediate Wi-Fi bridges (the so-called mesh networks, http://www.ni.com/white-paper/11211/en#toc2).

The UC Berkeley scientists have developed a special operating system TinyOC [*Hill et al.*, 2000] and the nesC extension of the C language [*Gay*, 2003; *Levis and Gay*, 2009] for multithreaded programming of data processing tasks in real-time by energy-efficient microprocessors. It allows to compile multitask applications on Windows and Linux platforms for TinyOS and place them in 8 K memory of the microprocessor to handle 512 KB data buffer. These tasks include the digitization of a number of inputs from the sensors (for example, temperature, light, etc.), cyclic buffering and data smoothing, a simple data analysis, formation of communication and transmission of data over a network to a centralized data center.

3. External Sensors

The basic set of sensors placed on the TelosB motherboard was expanded by an external thermocouple, infrasonic microphone and accelerometer.

A thermocouple with measuring threshold up to 400°C is connected to the TelosB platform based on a specialized chip ASIC MAX6675 (Figure 2). External ADC is connected to the TelosB motherboard by five channels: grounding channel, channel for transmitting command to initiate the process of obtaining the next value, clock control channel to control the speed of data transmission, digital channel for data acquisition, power supply channel. A software module was written in the nesC language to

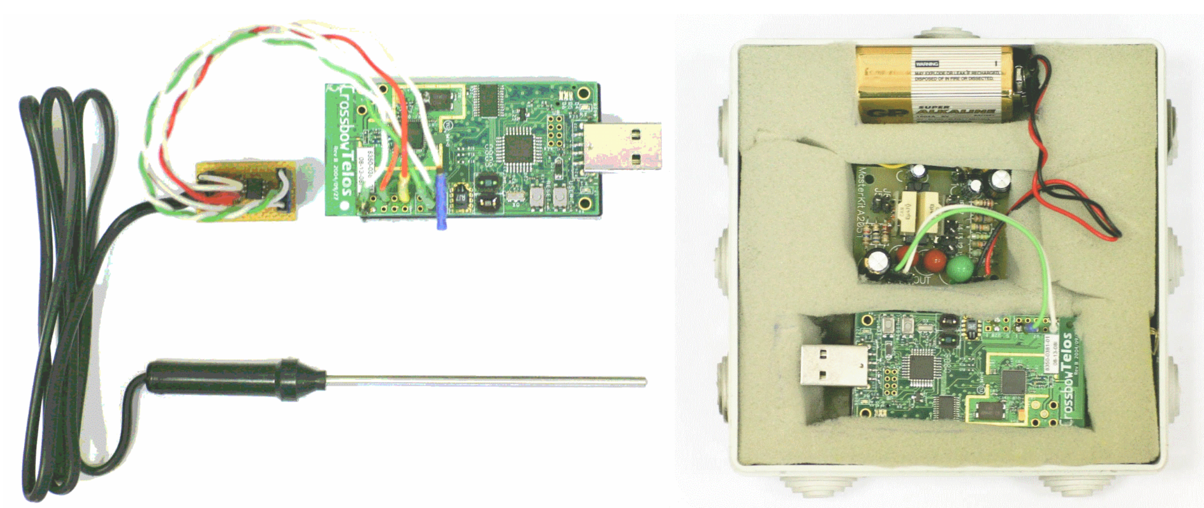


Figure 2. TelosB platform with attached thermocouple and infrasound microphone.

work with the thermocouple. Circuit design and module code in the nesC language for connecting thermocouples to the TelosB platform using an IC MAX6675 is shown here: http://sourceforge.net/projects/wsndetectors/files/sensors.doc/download.

Infrasound microphone was built on the basis of an analog operational amplifier and RC-low-pass filter with a threshold of 30 Hz. The sensor is connected directly to the TelosB platform by two channels: grounding and data transmission channels (Figure 2). The sensor generates an analog signal, so an ADC built into the TelosB platform, was used. The sensor requires an additional power supply of 9 V, as TelosB provides only 6 V. All parts of the system, including microphones, focused on the three spatial directions, are packaged in a plastic container. A driver in the nesC language was written to work with the sensor.

A board with MMA7260 accelerometer is connected directly to the TelosB platform by four channels: one grounding channel, the other three are data transmitting channels (one for each spatial axis X, Y, Z). The sensor sends data coded by an analog signal, so the TelosB ADC and a special program module in nesC language are used to work with the accelerometer.

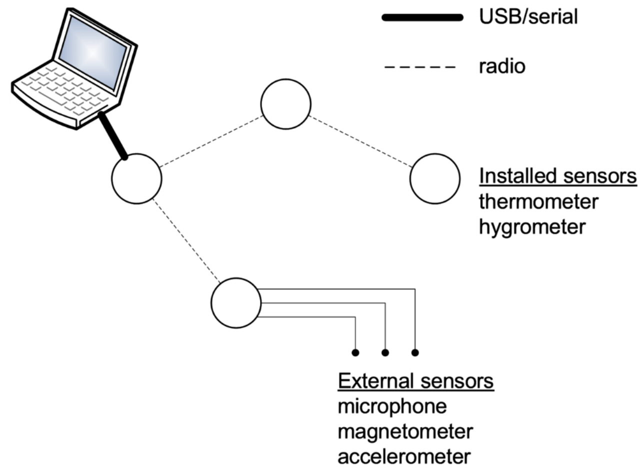


Figure 3. Scheme of wireless sensor modules network.

4. Data Collection and Transmission in Wireless Sensor Networks

To ensure the proper opeartion of the sensor network, a special program must be installed in each node, to control the platform and all of its software and hardware components. In theory, such program can be unique for each node, but because of the possibility of using hundreds of nodes in a single network, same programs are usually being used for the majority of nodes. In the simplest case, we can distinguish two types of nodes with different software: the work units for the collection of data and the base station, connecting the network nodes to the host computer.

The operating system for the sensor chip is called TinyOC and provides a package of basic software components written in nesC, including the drivers and utilities for hardware elements of the platform (radio, USB memory, ADC and DAC, integrated set of sensors, light emitting diodes, etc.) and modules for the organization of software infrastructure (task queue, message queue, network communication protocol of data radio transmission, etc.). Using the supplied components, the user can append his superstructures of higher level for data processing and management. Software environment for the development of programs for intelligent sensors on Linux platform allows you to edit the source code in nesC, compile the program, download the binary code in the microprocessor memory, debug it, and interact with other wireless sensor network nodes.

After uploading the program into the sensor memory and switching on the power, the microprocessor automatically resets and the network node begins to function. In our case, the node starts to poll sensors with subsequent data processing. One dedicated node is a base station. Its functional purpose is to send control commands to other nodes in the network, control general settings (for example, the local time of nodes), transfer data from a network to a connected computer. When transferring data, all nodes send measurements to a base station, which through a USB-port transmits them to the management program running under the Windows or Linux OS of the PC computer.

Use of the basic components of the TinyOC system allows us to organize data collection as using built-in sensors (thermometer, hygrometer), and with the help of external analog sensors (eg, microphone, magnetometer, etc.) connected to the input of an analog-to-digital converter of wireless sensor module. Using standard protocols LE (Link Estimation) and CTP (Collection Tree Protocol), we can organize data transfer between the sensor modules in a chain, forming a multi-hop way. Moreover, if a module is out of range of the base station, it can transmit data through a chain of other modules. The approximate scheme of the network is shown in Figure 3.

A sensor network works in the following way. Wireless sensor network nodes are installed in the studied area, determine their "neighbors" by radio and start to collect data. Depending on the task, the nodes can store data in memory without processing, or analyze data to allocate the required conditions (events). The optimal solution depends on the frequency of obtaining measurements.

In the case of low-frequency data (1 second or less), all observations can be temporarily stored in local memory on a node. Several megabytes of memory are enough to store data for a long enough time. At the request of the operator or the timer the control unit initiates the downloading of data.

In the case of high-frequency observations the size local memory of the sensor is not enough to store, and the capacity of the radio channel is not sufficient to transfer the total data volume: in our tests the maximum rate of a reliable transmission using ZigBee protocol is about 1500 measurements per second. In this case we use the Event Listeners:

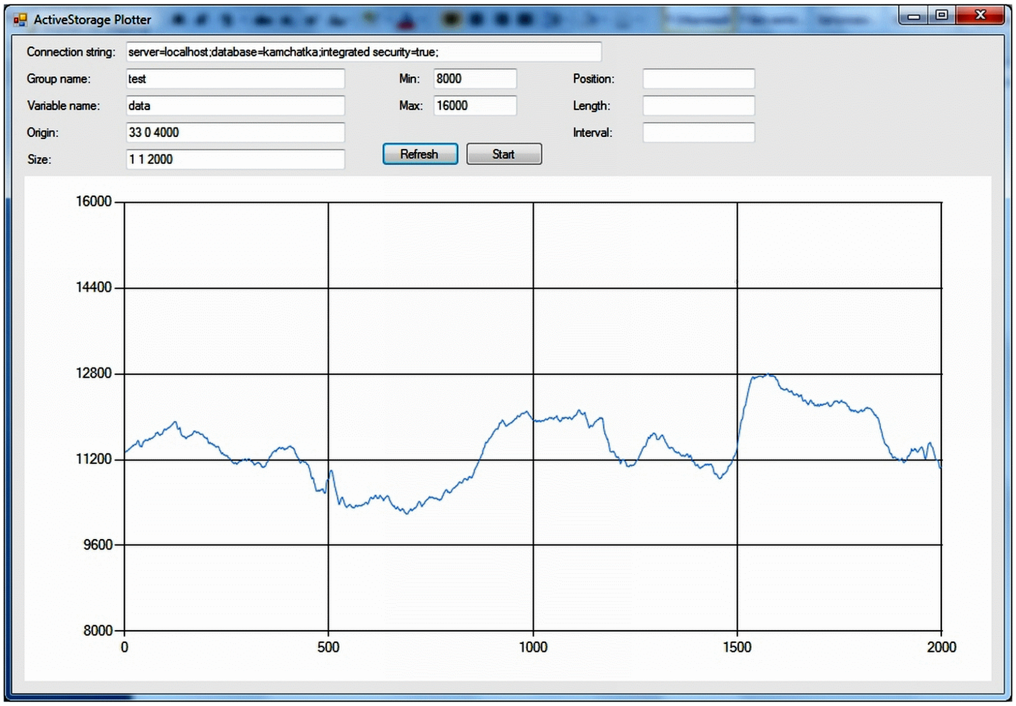


Figure 4. Screenshot of applications for WSN data visualization.

each node processes its own data, and when an event is detected, it retains data only in a small range, or refuses to save data, passing only a summary through the base station. An example of a detector of events is given in Section 5. The low data throughoutput of wireless sensor networks is the inevitable choice: either to lose a part of data, or to connect more efficient equipment (for example, Wi-Fi transmitters, power supplies, etc.), which will increase the size and power consumption of the network nodes and reduce its autonomy.

An important component of time series is observation time value. To determine the time of measurement, one can use local clocks built into each node. To put the data obtained from different nodes in a proper order, the nodes have to be synchronized in time, which can be made with one of the basic components of TinyOS, implementing the FTSP protocol (Flooding Time Synchronization Protocol) [*Maroti et al.*, 2004] FTSP is a common time synchronization protocol for wireless networks, which provides high accuracy of synchronization by using special tags on two levels of the protocol stack of radio transmission. FTSP also takes into account a drift of clock signals of sensor modules and introduces the appropriate correction using linear regression. An alternative option is to use a GPS-receiver on the selected nodes. The advantage of this approach is related to the ability to synchronize the time not only between the nodes, but also to the global world time.

After downloading data to a computer through a base station, they should be placed in a storage (they can be stored remotely if you have several computers, or they may change) with the possibility of visualization. The simplest version of the data model to store parameters is time series time–value.

5. Program Software for Data Collection and Visualization

For collection of low-frequency data for sensor network nodes based on the Koala project [*Musaloiu-E. et al.*, 2008] a control program was developed, for collection and storage of data into the local memory of a data node of built-in and external (connected both by digital and analog channels) sensors, downloading data from the nodes in a required moment through one node, connected on a USB-port (base station) along the dynamically constructed multi-hop routes, to determine the global time of all measurements and to save battery power (up to 1–2 years) by work optimization of radio transmitting components.

We have developed scripts to store data, check their integrity, transfer them from internal representation of the sensors into the standard units of measurement based on the calibration information, save data in CSV format for MS Excel or download data to a local or remote storage, which could be: SQL-databases, ActiveStorage [*Zhizhin et al.*, 2011] database, Earthworm [*Childs and Komeç*, 2003] distributed system for seismic data.

The ActiveStorage scientific database was developed at the Geophysical Center RAS to store time series, satellite images, results of numerical simulations, as well as any other information that can be presented in the form of multi-dimensional numerical arrays. The storage main features are:

- Flexible architecture allows you to keep heterogeneous data in a single storage system.

- Effective indexing of large amounts of data (hundreds of terabytes).
- Basic data processing directly on the storage nodes (arithmetic operations, statistical analysis, linear convolution).
- Integrated metadata, data description cannot be separated from the actual data.
- Automatic distribution of data (as well as parallel processing) in multiple nodes and in a cloud.
- It can be used in the Grid infrastructure through OGSA-DAI services [*Dobrzelecki et al.*, 2010; *Jackson et al.*, 2011].

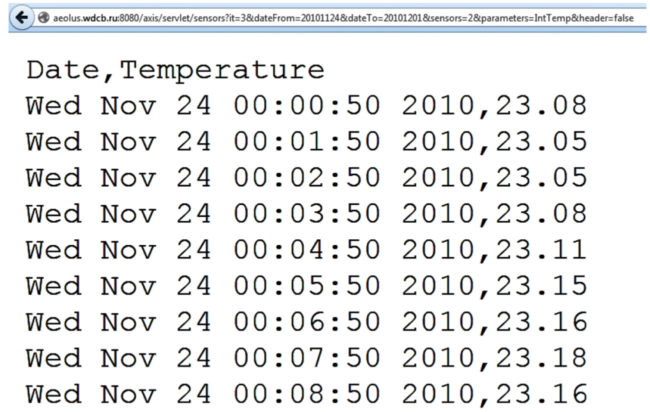
To visualize data, collected in real time (without intermediate saving to the nodes' local memory), a Windows-application was developed that allows you to read one-dimensional arrays (time series) of ActiveStorage and display them in a graph. The application enables tracking data updates in the storage online, reading a tail of specified length of the array in regular time intervals (for example, every second). The screenshot of applications for data visualization is shown in Figure 4.

To access the measurements stored in the database we have developed a web-service that allows you to query to the database for a required set of parameters for a certain time interval obtained in a given set of nodes, and returns the results in CSV format (Figure 5).

The same web-service provides graphical representation of time series for a given parameter, obtained from a given node in a given range of dates, selection and scaling of required portions of graphs, coloring and highlighting of values obtained at a given time point (Figure 6).

To visualize time series with reference to electronic maps a web application was developed based on a tile server. A map, obtained through the GoogleMaps (GoogleMaps project. Project site: http://maps.google.com/) application programming interface, was used as a cartographic layer. For visualization of the sensor location on the map we use the "pushpin" element of the GoogleMaps interface, which is portrayed as a marker on the map at the geographical location of the data source. A graphical window is tied up to the marker, which displays a graph of the time series by the Protovis package [*Bostock and Heer*, 2009] (Figure 7).

Tile server is a service to retrieve map fragments (tiles). A tile is given by level of detail (Z) and coordinates (X, Y). The tile size is fixed at 256×256 pixels. At the 0 level of detail the whole map is covered by one tile with $0, 0$ coordinates. At each level of detail, the number of tiles increases by four times. Thus the map can display raster data showing the spatial distribution of the values of geophysical field at a given time.


aeolus.wdcb.ru:8080/axis/servlet/sensors?it=3&dateFrom=20101124&dateTo=20101201&sensors=2&parameters=IntTemp&header=false

```
Date,Temperature
Wed Nov 24 00:00:50 2010,23.08
Wed Nov 24 00:01:50 2010,23.05
Wed Nov 24 00:02:50 2010,23.05
Wed Nov 24 00:03:50 2010,23.08
Wed Nov 24 00:04:50 2010,23.11
Wed Nov 24 00:05:50 2010,23.15
Wed Nov 24 00:06:50 2010,23.16
Wed Nov 24 00:07:50 2010,23.18
Wed Nov 24 00:08:50 2010,23.16
```

Figure 5. Result of work of data acquisition web-service.

6. Examples of Use

6.1. Detection of High-Frequency Events Based on STA/LTA Algorithm

At the present time the STA/LTA (Short Time Averaging/Long Time Averaging) detector is widely used. The principle of its operation (Figure 8). is to analyze the ratio of signal strength, averaged over a short time interval (STA), to the power averaged over a long interval (LTA). The decision on the presence of the signal in the noise is made if STA/LTA ratio exceeds the threshold. This threshold is usually in the range of 4–5 and depends on the properties of noise in a particular situation. It should be noted that the statistics of STA/LTA ratio is optimal for detecting signals in Gaussian noise.

After turning on, each node in the network starts collecting data from the sensor and buffering them in a circular array. Size of the circular array is chosen to be equal to the sum of the sizes of long and short windows. Buffering process continues as long as the array is completely filled. After that the sum of values for the LTA window and the sum of values for the STA window are calculated, we find the STA/LTA ratio. At the next iteration, which starts when a new measurement from the sensor is received, the sum of values for LTA and STA is not recalculated. The first value is deducted from the sum, then the window is shifted by one count forward and the last value in the window is added to the sum. This reduces the load on the processor node and increases the speed of computation. The windows move through the array in a circular mode.

When the specified threshold of the STA/LTA ratio is exceeded, a message is sent to the base station, containing the number of the sensor, which has recorded the occurring event, and the value of the STA/LTA ratio. The data are stored in the internal memory of the WSN node. Next message will be sent when the STA/LTA ratio is below the given threshold.

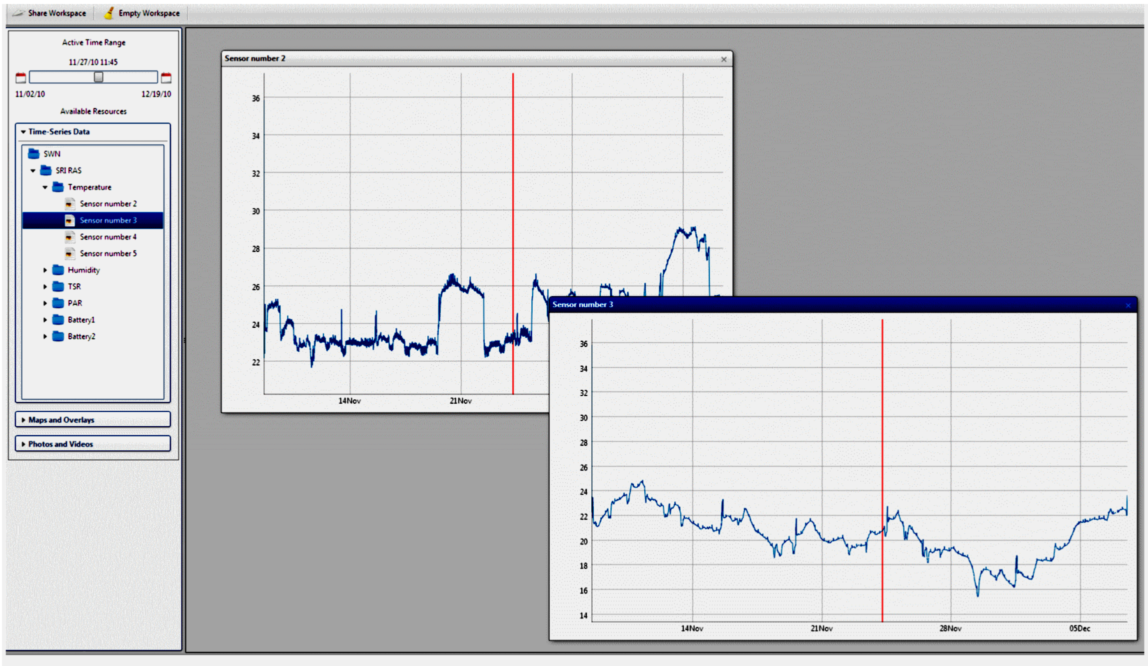


Figure 6. Web-application for graphical representation of time series.

6.2. Environmental Monitoring in Supercomputing Center

The experiment was carried out in the computing center with a total area of 1000 m^2, which hosts several dozen telecommunication cabinets of servers, providing an "aggressive" environment in the form of electromagnetic interference and physical barriers. Established for the experiment wireless sensor network consisted of four nodes, equipped with sensors for temperature, humidity, photosynthetic ac-

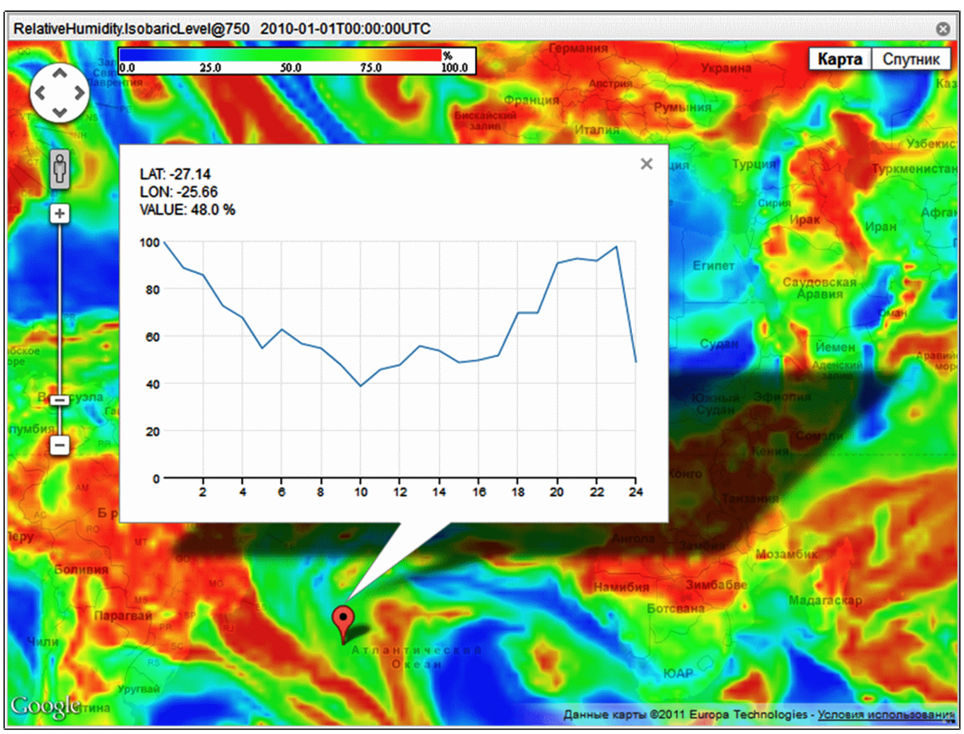


Figure 7. Mapping of time series graph.

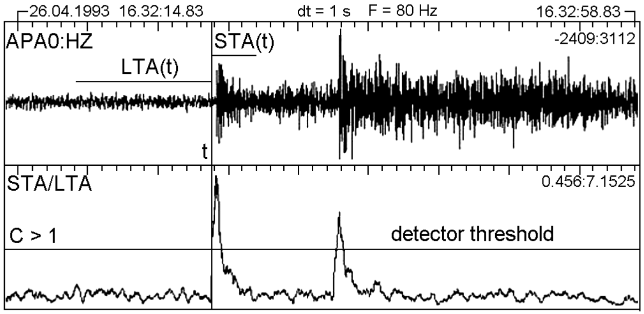


Figure 8. Scheme of LTA/STA detector.

tive radiation, light, which were placed at a distance of 10–50 meters from each other. The purpose of the experiment was a non-stop data collection of physical parameters at 1 minute frequency with an irregular data uploading to a central repository.

The work was done according to the following scenario (Figure 9):

1. The data were downloaded irregularly, at intervals of 2 to 7 days with a base station connected to the netbook running OC Windows OC Windows to host the virtual machine running TinyOS 2.1. At downloading data, the base station's location varied from close vicinity to the nearest node to a distance of 10–15 meters from it with obstacles in the form of walls and radio interference with computing equipment.

2. The monitoring results were stored in specific raw form to a local file on a virtual machine.

3. Raw data were normalized into the standard measurement units and converted into digital format.

4. Data in digital format were uploaded into a remote database MySQL, from which they can be retrieved by the system of visualization.

The experiment has established:

- duration: 34 days;

- drop in battery power from 3.2 to 3 volts on average (it is worth noting the hostile environment, periodic interruption of channels and need to re-download data lost during transfer);

- network polling rate: it depends on the remote location of nodes and the number of intermediate links in the route of data downloading, but the average time taken to download data received daily from one node (6 sensors with measurement period per minute), was equal to minute;

- network activating rate before downloading data: by default, to activate all the nodes, the base station got one and a half minute, but it actually took about 10–20 seconds.

The monitoring experiment has shown positive results in terms of saving battery life and reliability of data storage (not a single measurement was lost). Another positive result was support for communication at a distance of 10–20 meters over obstacles (walls) and in conditions of constant noise from the equipment. During more than a month of the experiment, not a single node needed technical maintenance.

7. Conclusions

There is a number of advantages and disadvantages of wireless sensor networks. The benefits include:

- Simple installation and maintenance. The network is easy to install, it doesn't require any additional communication. Downloading of data does not require

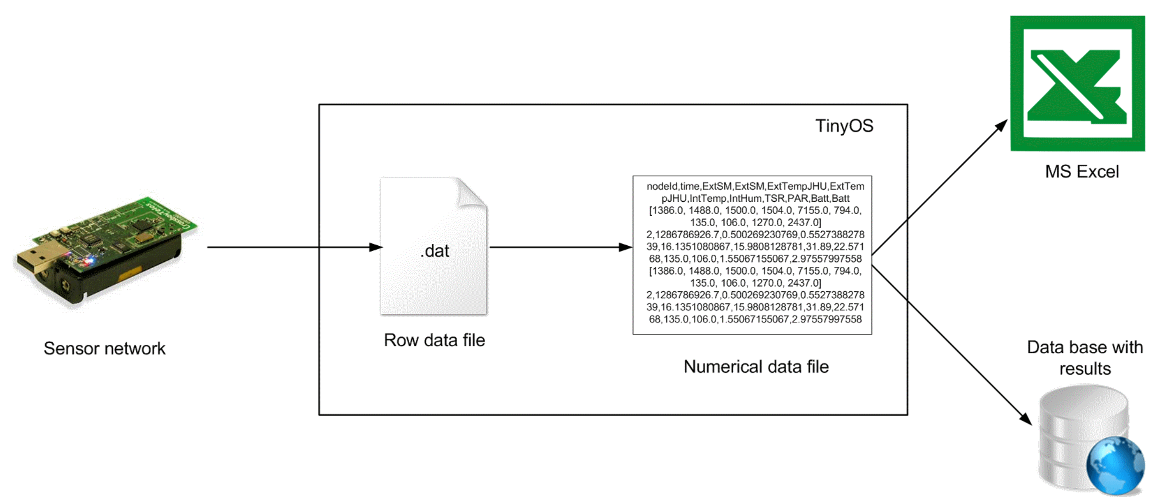


Figure 9. Scenario of data processing.

direct contact with each node, which is important for inaccessible locations.

- Mobility. Network configuration can change rapidly without additional effort. Software update for each sensor can be loaded on the same radio channels as the data collection.

- Scalability. The network may include hundreds or thousands of nodes. New nodes can be added dynamically without having to reconfigure the network.

- Separate "dense" network islands, in which data is exchanged via ZigBee, can be combined into more extensive mesh networks using Wi-Fi gateways.

The main shortcomings are:

- Limited power consumption. This is both an advantage (long battery life), and a disadvantage of wireless sensor network nodes, leading to restriction of processor power, memory and radio exchange rate.

- Errors of radio transmission. A hostile environment (interference, bad weather conditions) can cause an increase in errors of radio transmission.

Thus, the decision to use a wireless sensor network depends on the geophysical problem that has to be solved.

Acknowledgments. The results were obtained with the financial support of the RFBR grant # 10-07-00682-a.

References

Bostock, M., and J. Heer (2009), Protovis: a graphical toolkit for visualization, *IEEE Transactions on Visualization and Computer Graphics, 15*, 6, 1121–1128.

Childs, D., and A. Komeç (2003), The Kandilli Observatory Real-Time Automated Seismic Data Processing System, *ORFEUS Newsletter* (http://www.orfeus-eu.org).

http://www.orfeus-eu.org

Dobrzelecki, B., A. Krause, A. Hume, A. Grant, M. Antonioletti, T. Alemu, M. Atkinson, M. Jackson, and E. Theocharopoulos (2010), Integrating distributed data sources with OGSA-DAI DQP and Views, *Phil. Trans. R. Soc. A, 368*, 1926, 4133–4145, doi:10.1098/rsta.2010.0166.

Farahani, Shahin (2008), *ZigBee Wireless Networks and Transceivers*, Newnes.

Gay, David, Phil Levis, Rob von Behren, Matt Welsh, Eric Brewer, and David Culler (2003), The nesC Language: A Holistic Approach to Networked Embedded Systems, *Proceedings of Programming Language Design and Implementation (PLDI)*.

Hill, J., R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister (2000), System Architecture Directions for Networked Sensors, *Proceedings of Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*.

Jackson, M., M. Antonioletti, B. Dobrzelecki, N. Chue Hong (2011), *Distributed Data Management With OGSA-DAI. Grid and Cloud Database Management (eds. S. Fiore and G. Aloisio)*, Springer-Verlag, 63–86, doi:10.1007/978-3-642-20045-8.

Levis, Philip, and David Gay (2009), *TinyOS Programming*, Cambridge University Press.

Maroti, M., B. Kusy, G. Simon, and A. Ledeczi (2004), The flooding time synchronization protocol, *Proc. 2nd ACM conference on Embedded networked sensor systems (SenSys)*, November 2004.

Musaloiu-E., Razvan, Chieh-Jan Liang, Andreas Terzis (2008), Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks, *Proceedings of IPSN.*

Zhizhin, Mikhail N., Dmitry Medvedev, Dmitry Mishin, Alexey Poyda, Alexander Novikov (2011), *Transparent Data Cube for Spatiotemporal Data Mining and Visualization. Grid and Cloud Database Management*, Springer, 307–330, doi:10.1007/978-3-642-20045-8.

A. V. Andreev and M. N. Zhizhin, Federal State Institution Geophysical Center of the Russian Academy of Sciences, Moscow, Russia. (a.andreev@gcras.ru; m.zhizhin@gcras.ru)

A. A. Poyda, Research Center "Kurchatov Institute", Moscow, Russia. (poyda@wdcb.ru)