# Software Platform for Reporting, on the Internet, Data Stored from Sensors Used to Monitor Production Processes

Catalin Mateias and Adrian Nicolescu

***Abstract*— This paper presents a software platform developed for recording and storing data from sensors installed in locations that require constant monitoring of environmental parameters. The data is stored in a relational database built using Oracle XE. The secured web pages making the data available to Internet user are developed using Oracle APEX. Each Internet user will have access only to the measured values from sensors they own. If parameters limits are exceed, the software platform will send warning e-mails and phone messages. The software platform is developed using open source or freeware software.**

***Index Terms*—communications, data management system, process monitoring, smart home**

## I. Introduction

To ensure optimal and safety conditions in production, certain production processes require constant monitoring. This is achieved by installing sensors like thermometers, hygrometers, lightmeters, manometers, anemometers or any other type of measuring device according to the production process technology in the location where the production process takes place.

## II. Problem Formulation

Sensors are acquired from the manufacturer with basic software for storing the measured values on a computer hard drive. Usually the measured values are stored in text files or databases: Microsoft Access, Microsoft SQL and MySQL.

In the location where the production process takes place several sensors are installed from different manufacturers, then on the computer where the data is centralized, the software for each sensor must be installed. In order to view the measured values, the user will have to access each sensor software [1]. A problem arises when the data must be used for statistical purposes (display maximum temperature per day or per month) which might require the data to be centralized in a global format. Even if all the manufacturer sensor software export data in text files, the data from the files will have a custom format according to the manufacturer, meaning that it will take a lot of time to centralize data from the files in one file with a global format, even if all the sensors measure the same type of physical phenomena [2].

## III. Problem Solution

This paper describes a software platform developed using open source and freeware software, able to merge and centralize data from several sensors made by different manufacturers, installed in locations that require special environment conditions, in a single database and centralize the results in one user interface that can be accessed from

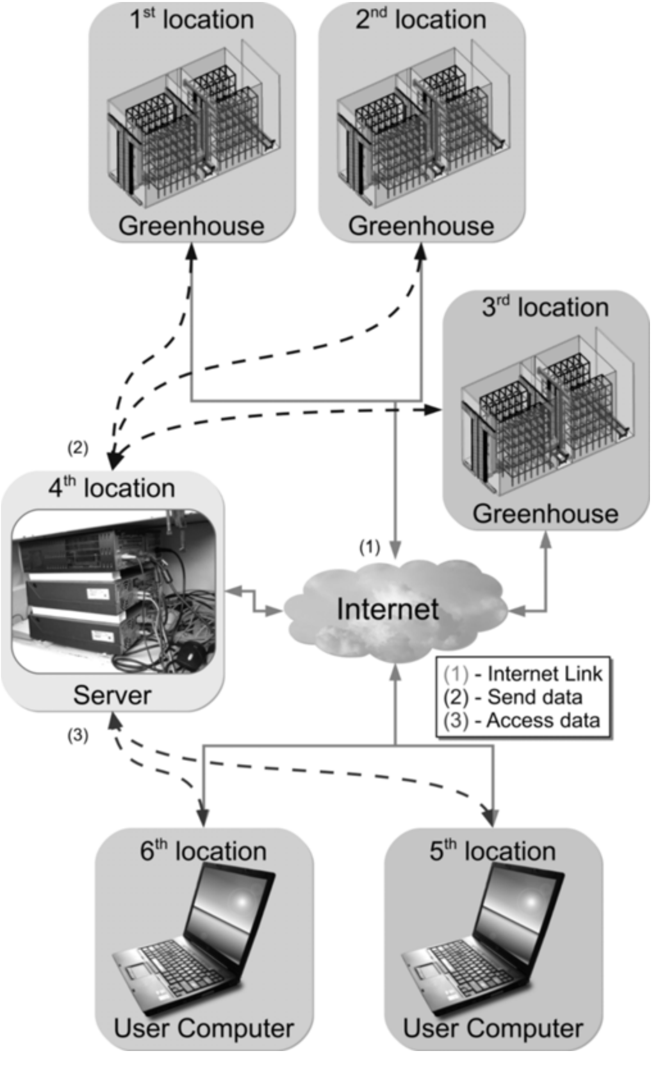

Fig. 1. Research description

Manuscript submitted for review February 28, 2012; revised March 19, 2012.

C. Mateias is with the University Politehnica of Bucharest, Bucharest, 060042 Romania (corresponding author: phone: 074-695-5245; e-mail: cata.mateias@gmail.com).

A. Nicolescu is with the University Politehnica of Bucharest, Bucharest, 060042 Romania (e-mail: afnicolescu@yahoo.com).

the Internet[3][4][5].

In figure 1, there are three locations that require environmental monitoring of temperature, humidity, air flow, pressure and light intensity. The data measured by the sensors is sent via Internet to an Application Server (4th location). If any of the users want to see the measured values, then they will access the web portal on the Application Server. The 1st and 2nd location belong to user 1, while the 3rd location belongs to user 2. According to the user security credentials each user will have access to data only from the locations they own.

This type of software application implies joining freeware and open source sensor drivers, software libraries, own developed relational database structure, own developed procedures and functions, own developed interlink software modules between sensors and database, own modules for data representation (graphs, reports, warnings), own modules for the management of users, locations, and measuring devices.

The commercial solutions of this type of software application that can report on the Internet data from sensors are too expensive, or do not process certain range of sensors, and the compiled code and license do not allow to adapt the code to the production process specificity.

This research was started as a side project for a fully automated mushroom cultivation greenhouse, to monitor temperature, humidity, pressure, air flow, light intensity [6].

### *A. Solution Concept*

From the development stages set for this software platform, this paper will present the database structure and the module for inserting into the database, the measured values from the sensors. Starting from the top in figure 2, several sensors from different manufacturers are used for measuring environmental parameters for a location that requires monitoring. The sensors are connected to a computer in order to store the measured values by LAN, WLAN, Bluetooth or special adapter. The measured values are grabbed from the sensors by the sensor manufacturer software or by the own developed software. The sensor used in this research (Fig. 5) has 3 communication protocols: ModBus, SOAP and SNMP.

If using the own developed software (line 3), the data is stored in a relational database developed using Oracle XE. The own developed software is developed using JAVA, it uses SNMP to communicate with the sensor, it uses OJDBC library to communicate with the Oracle XE database and runs as a service in Linux Fedora Core (the computer operating system). The software first connects to the database, then it loads the sensors configurations, next it interrogates each sensor found when loading the configuration, to get the temperature, humidity, pressure and/or dew point values(according to the configuration), and last it inserts the measured values into the database. In this research the temperature, humidity, pressure and dew point are measured once per minute, every day.

If using SOAP protocol, then the sensor is sending XML messages containing the measured values once per minute to the Data Storing Computer configured with Apache web server and PHP. A PHP script will insert the measured

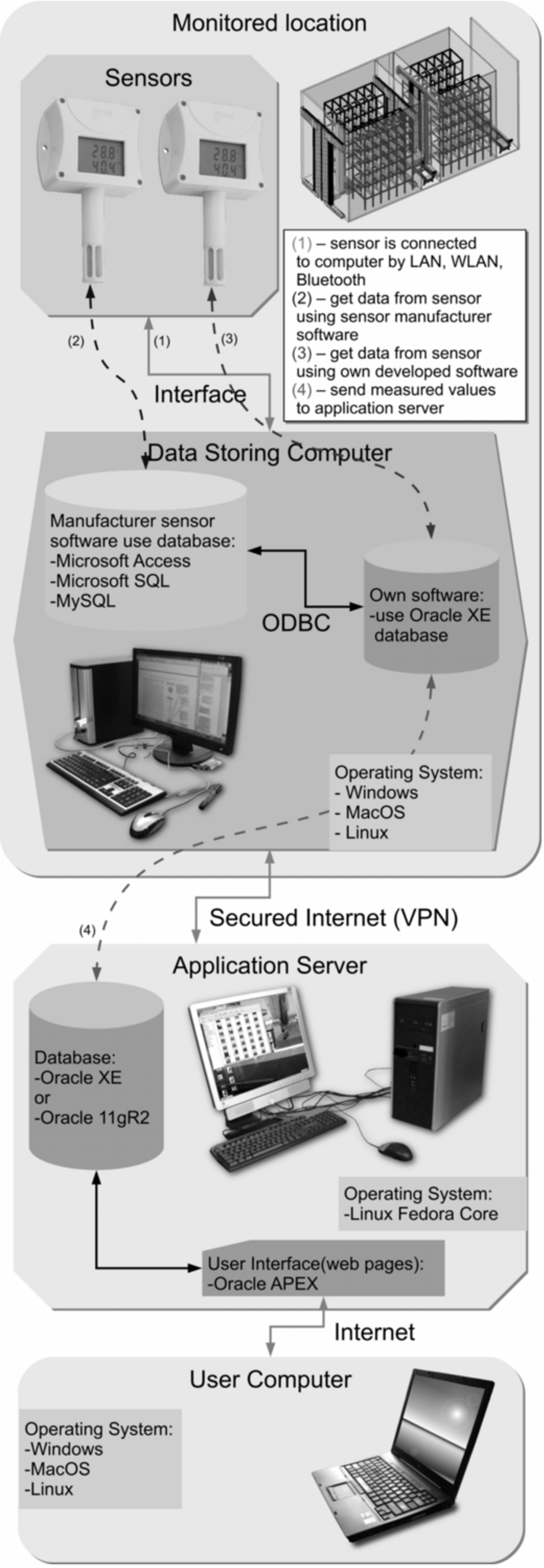

Fig. 2. Concept

values in an intermediate database developed using MySQL. Then the Oracle XE database will connect to MySQL using ODBC or own developed software in JAVA or C++, selects the measured values and then deletes the records from the MySQL database. The intermediate database is used for security reasons, because the PHP script contains the username and password used for connecting to the database. In other words, the intermediate database is accessed by the PHP script to write data, and it is accessed by the Oracle XE to copy and delete data.

If using the sensor manufacturer software, the measured values are usually written in a text file (*.txt) or a commercial database (Microsoft Access, Microsoft SQL). Mostly, the sensor manufacturer develops the sensor software for Windows operating system, so in this case the Data Storing Computer will have Windows installed. The data is accessed by Oracle XE from the commercial database using ODBC or read from the text files by own PL/SQL developed procedures from Oracle XE.

On the Data Storing Computer the measured values are available only locally. To make the data available on the Internet, the Oracle XE database installed on the Data Storing Computer will send the data to the Internet Application Server using a secured Internet connection. On the Internet Application Server the received data is also stored in an Oracle database. In present state of the research, on the Application Server an Oracle XE database is installed, but if required an Oracle 11gR2 will be used. The Application server has a static IP address and a registered domain name which makes the data available to Internet users by web pages containing graphs and reports. The web portal is developed using Oracle APEX which can integrate CSS, XML, JavaScript, and Java. To sum up, the Internet Application server is formed of 4 computer servers.

One computer is the firewall, the second computer is the DMZ(Demilitarized zone) where the DNS server, the mail server, and the Internet Server are configured, the third is the database server where the Oracle database and Oracle APEX are installed, and the fourth is the backup server.

### *B. Getting data from the sensor*

In SQL, the data is stored in tables. These tables are grouped in schemas. The tables and schemas are owned by users that according to the user privileges, can create, alter or drop tables from the schemas. A “tablespace” is a logical space to store the users, schemas, objects (tables, procedures, functions, packages,…).

Figure 3 shows how the measured values from the sensor are inserted in the database. The piece of software that does this was developed using Java and runs as a service in Linux Fedora Core. After the operating system loads, the service first loads the configuration from the “/etc/” folder. The configuration file contains the ip address, service name and listener port to locate the database, and the username and password in plain text, in order to connect to the “buffer schema” inside the database. Then it loads the sensors configuration (like the ip address in order to locate the sensor in the network) and using the communication protocol specified when loading the sensor configuration, it queries each sensor for the measured values. After

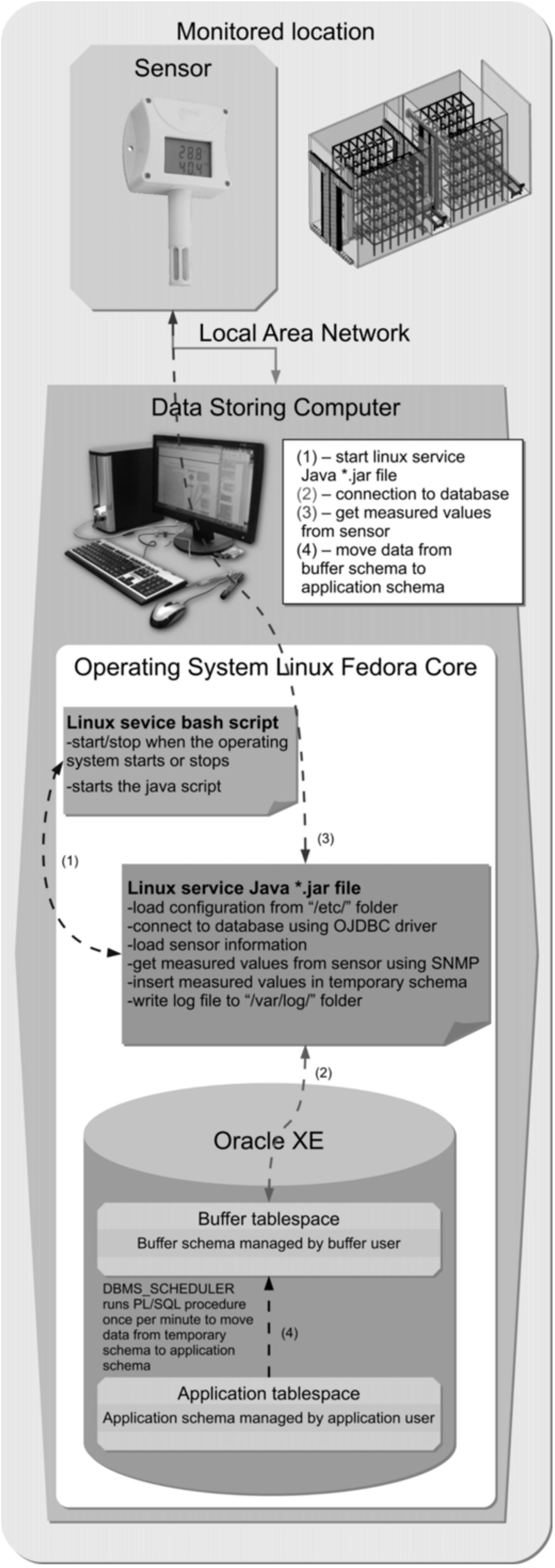
Monitored location
Sensor
Local Area Network
Data Storing Computer
(1) – start linux service Java *.jar file
(2) – connection to database
(3) – get measured values from sensor
(4) – move data from buffer schema to application schema
Operating System Linux Fedora Core
Linux sevice bash script
-start/stop when the operating system starts or stops
-starts the java script
(1)
(3)
Linux service Java *.jar file
-load configuration from “/etc/” folder
-connect to database using OJDBC driver
-load sensor information
-get measured values from sensor using SNMP
-insert measured values in temporary schema
-write log file to “/var/log/” folder
(2)
Oracle XE
Buffer tablespace
Buffer schema managed by buffer user
DBMS_SCHEDULER runs PL/SQL procedure once per minute to move data from temporary schema to application schema
(4)
Application tablespace
Application schema managed by application user

Fig. 3. Getting data from the sensor.

successful querying, it inserts the measured values in the tables from the “buffer schema” owned by the “buffer user”. If it fails to get the measured values from a certain sensor after the timeout period, it writes a report in a log table from the “buffer schema” and in a log file from “/var/log/” folder.

Inside the Oracle database this software application was developed using two schemas. A “buffer schema” to be used for inserting the measured values from the sensors into the database, and the “application schema” used for storing the data from sensors. The “buffer user” is the owner of “buffer schema” and has the following privileges: connect (connect to the database schema), select (used to get the information about the sensors), insert (for inserting the measured values).

The “application user” which is the owner of “application schema” has select insert, update and delete privileges into the “buffer schema” granted by the database administrator. The “buffer user” and the “application user” cannot grant themselves, or to other users privileges, or revoke privileges they have or from other users.

The “application user” has access to the “buffer schema” which facilitates a PL/SQL procedure owned by “application user” to copy the data from “buffer schema” into the “application schema” and remove the records from the “buffer schema”. The PL/SQL procedure is run once per minute by Oracle DBMS_SCHEDULER package.

The “buffer schema” is used as a security measure, because the username and password are stored in plain text in the configuration file from the “/etc/“ folder. The “buffer schema” contains only the tables that contain necessary information to identify the sensors, and the tables where to insert the measured values, unlike the “application schema” that contains all the tables and procedures to manage the Internet users, the locations where the monitoring takes place, and other necessary modules.

### *C. Application database structure*

Figure 4 highlights the main modules with the main characteristics for insert, update, delete data in the “application schema” from the database. Each module contains more than one PL/SQL table. PL/SQL (Procedural Language / Structured Query Language) is Oracle Corporation’s procedural extension language for SQL and it is used to manipulate data (select, insert, update, delete) from tables and write procedures and functions in Oracle relational database [2].

The “Security Schema” module manages the username and password for each user that logs in to the Application Server database. The password is stored in an encrypted format, not plain text. The privileges for each user are also managed by the security schema. A privilege means granting access to a certain user to manage specific data. For example a user will have privileges to add new locations to the database, while another user will have only the privilege to view the locations, but not to make modifications.

The “Location Definition” module stores the locations where the monitoring takes place. It stores the location name, address, blueprint and the owner of the location. The location depends on the users from the Security Schema because the owner of the location is a user.

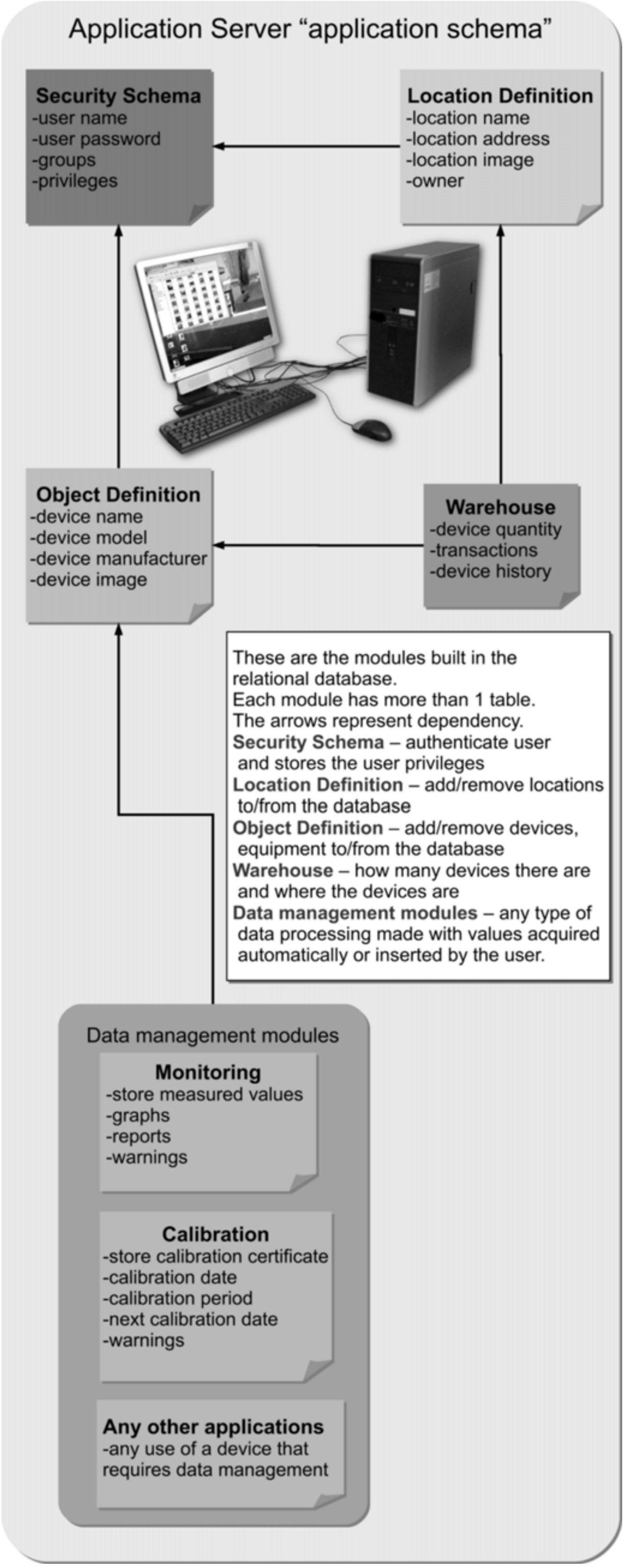
Application Server “application schema”
Security Schema
-user name
-user password
-groups
-privileges
Location Definition
-location name
-location address
-location image
-owner
Object Definition
-device name
-device model
-device manufacturer
-device image
Warehouse
-device quantity
-transactions
-device history
These are the modules built in the relational database.
Each module has more than 1 table.
The arrows represent dependency.
Security Schema – authenticate user and stores the user privileges
Location Definition – add/remove locations to/from the database
Object Definition – add/remove devices, equipment to/from the database
Warehouse – how many devices there are and where the devices are
Data management modules – any type of data processing made with values acquired automatically or inserted by the user.
Data management modules
Monitoring
-store measured values
-graphs
-reports
-warnings
Calibration
-store calibration certificate
-calibration date
-calibration period
-next calibration date
-warnings
Any other applications
-any use of a device that requires data management

Fig. 4. “application schema” modules.

The “Object Definition” stores the sensor name, type, manufacturer, image of the sensor and technical data of the sensor. The Object Definition module depends on the Security Schema because of the user privileges. Some users are allowed to modify sensors data others only to view it.

The “Warehouse” module manages the number of devices in the database and the locations where the devices are placed. It stores how many sensor of a type there are and where the sensors are located. Also it holds a history of the device when the location of the device changes. The “Warehouse” module depends of the “Object Definition” and the “Location Definition” because it merges devices and locations data.

The “Data management modules” represent any type of data processing that can be done with values measured by a certain device defined in the “Object Definition”. For the current research it represents the “Monitoring” module that contains tables that store the measured values from the sensors and generates reports and graphs and issues warnings if parameters limits are exceed. Another application is sending email messages before the calibration period of the sensor expires. The calibration period of a sensor is usually 1 year, after which it must be recalibrated by a metrological laboratory.

All the modules highlighted in figure 4 can be extended to fit application specificity. The “temporary schema” contains only a smaller version of the “Monitoring” module.

### D. Current state of the research

The next figures present screenshots from the current stage of the development of the software application. The sensor used in this research installed in a designated location is displayed in figure 5 and the technical specifications can be seen in table 1. All graphs and reports in the next figures contain values actually measured by the sensor. The sensor is connected to a local area network and can be accessed for configuration and gathering of data using its IP address. The temperature, humidity, dew point, pressure are measured once per minute every day.

In figure 6, the temperature graph is displayed. The temperature values can be seen by placing the mouse cursor over each dot. The measured values can be downloaded as a *.csv file that can be opened with Microsoft Excel, LibreOffice Calc or OpenOffice Calc.

In figure 7, the average, minimum and maximum temperature values per hour in the last 24 hours can be seen. The 1st line (green) represents the maximum values per hour, the 2nd line (blue) represents the average values per hour and the 3rd line (red) represents the minimum values per hour.

The last figure 8 presents the graph for the average, minimum and maximum temperature values per month.

## IV. Conclusion

This paper presents one of the development stages of a software platform used for storing and reporting over the Internet, data from sensors installed in locations that require constant monitoring (greenhouses, server rooms, food warehouses, metrological laboratories, weather stations) of environmental parameters like temperature, humidity, pressure, air flow, light intensity, or/and any other environmental parameters.

Unlike other commercial solutions this research implies the development of the software platform using open source or freeware software. The original contributions of this research are:

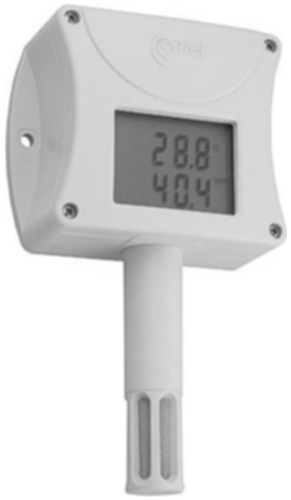

Fig. 5. The sensor used in this research.

TABLE I
SENSOR SPECIFICATIONS

| Characteristic | Value |
|---|---|
| Manufacturer | Comet |
| Model | T7510 |
| Temperature | -30 to 80°C |
| Humidity | 0 to 100%RH |
| Pressure | 600 to 1100hPa |
| Ethernet | RJ45 |

Intelligent barometric pressure, humidity and temperature probe with Ethernet interface.

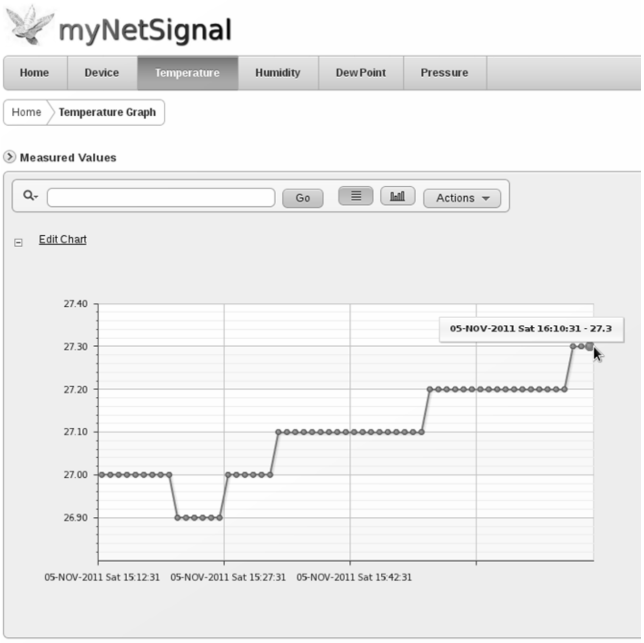

Fig. 6. Graph displaying the temperature. On Saturday, the 5th of November, 2011 at 16 o’clock, 10 minutes and 31 seconds the temperature was 27.3 °C.

--The development of procedures and functions to join freeware sensor drivers and software libraries, for automatic acquisition and storing of data from different sensor types.

-- The development of the software that sends the stored data from the Data Storing Computer from the monitored location to the Internet Application Server.

--Own developed database structure for the Data Storing Computer and the Internet Application Server.

--The development of modules for customer locations,

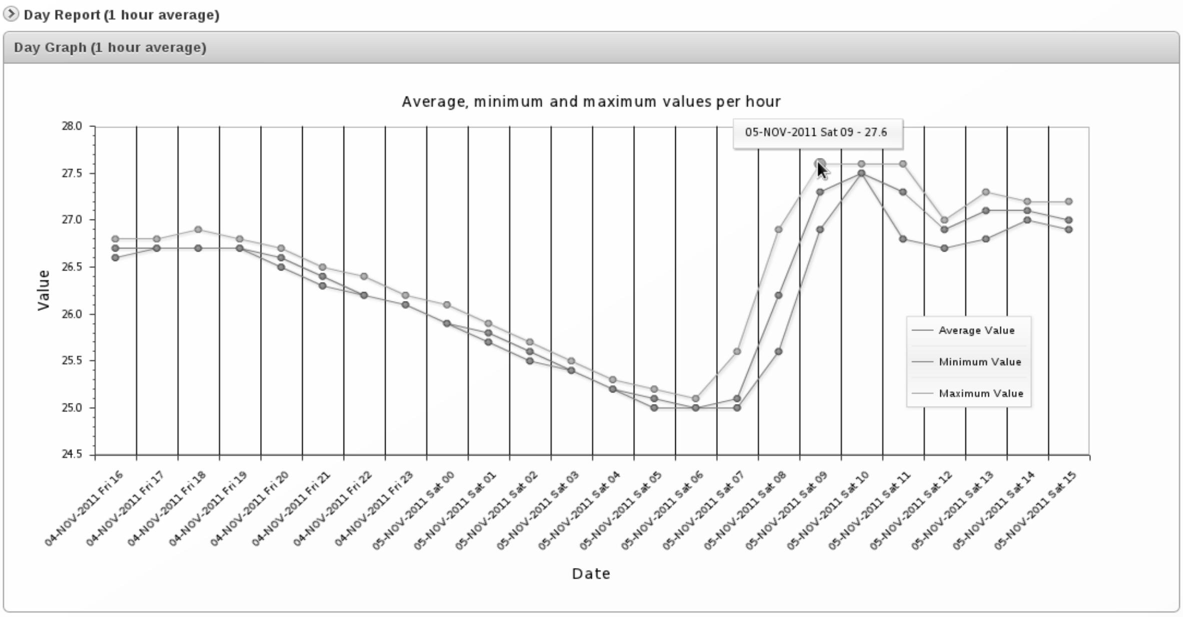

Fig. 7. Graph displaying the average, minimum and maximum temperature per hour. On the 5th of November 2011, Saturday at 9 o’clock the maximum temperature was 27.6°C.

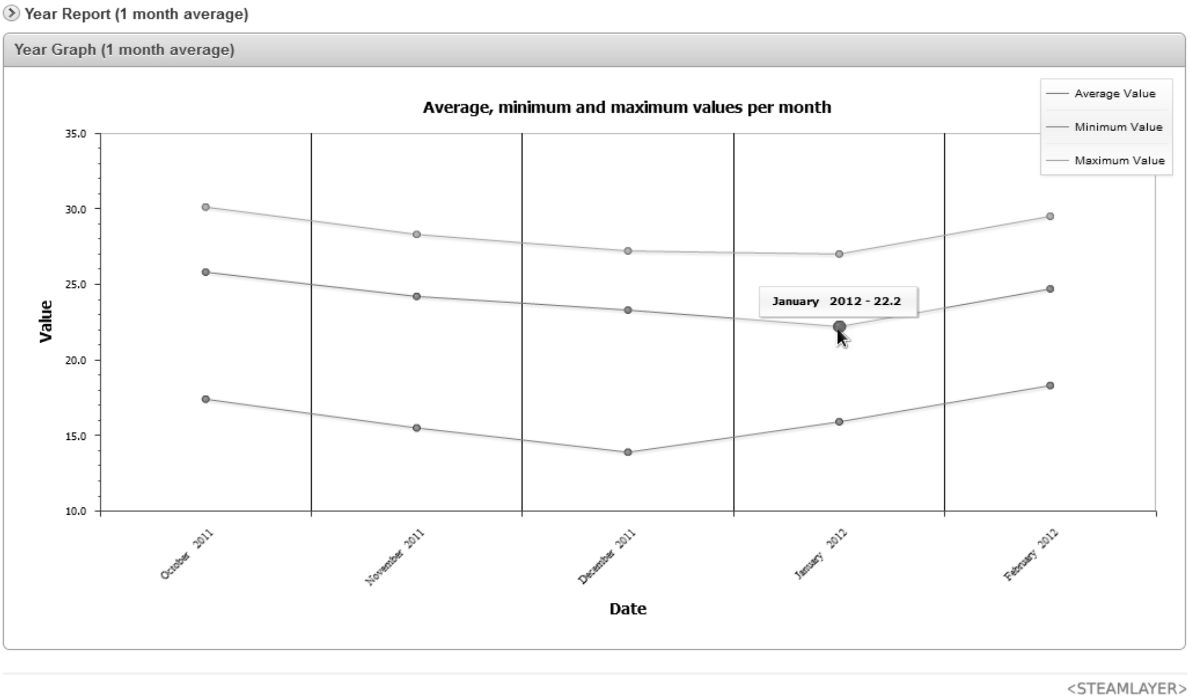

Fig. 8. Graph displaying the average, minimum and maximum temperature per month. The average temperature in January 2012 was 22.2ºC.

sensor installation, data processing and data representation (graphs, diagrams, tables, reports, warnings).

--Web portal for customer registration.

The use of open source and freeware software facilitates other developers beside the original developers to adapt, improve and fix bugs very quickly when compared to property software [7].

## REFERENCES

[1] N. Wang, N. Zhang and M. Wang, “Wireless sensors in agriculture and food industry – Recent development and future perspective”, *Computers and Electronics in Agriculture*, no. 50, pp. 1–14, Sep. 2005.

[2] A. Nicolescu, C. Mateias, A. Dorin “Software platform for online processing of data from sensors used to monitor environmental parameters”, *Proceedings in Manufacturing Systems*, Bucharest, vol. 6, pp. 29–36, Nov. 2011.

[3] R. Carniel, M. Cecca and O. Jaquet, “A user-friendly, dynamic web environment for remote data browsing and analysis of multiparametric geophysical data within the MULTIMO project”, *Journal of Volcanology and Geothermal Research*, vol. 153, pp. 80–96, Aug. 2005.

[4] H. Ding, B. Zhang, Y. Ding and B. Tao, “On a novel low-cost web-based power sensor via the Internet”, *Sensors and Actuators*, no. A136, pp. 456–466, Nov. 2006.

[5] H.J. Hermans and M. Vollenbroek-Hutten, “Towards remote monitoring and remotely supervised training”, *Journal of Electromyography and Kinesiology*, no. 18, pp. 908–919, Oct. 2008.

[6] C. Mateias, A. Nicolescu, M. Petre, A. Dorin “Developing a software platform for online data processing” in *Annals of DAAAM for 2011 & Proceedings*, Vienna, pp. 1301–1302, Nov. 2011.

[7] A. Neto, H. Fernandes, A. Duarte, B.B. Carvalho, J. Sousa, D.F. Valcarcel, M. Hron, C.A.F. Varandas, “FireSignal - Data acquisition and control system software”, *Fussion Engineering and Design*, no. 85, pp. 1359-1364, Feb. 2007.