This paper is a preprint (IEEE “accepted” status).

IEEE copyright notice. © 2009 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

DOI. 10.1109/TIC-STH.2009.5444401

Intelligent Wireless Distributed Network for Power Consumption Monitoring and Analysis

B.S. Popescu, A. Stancovici, V. Stangaciu, C. Certejan, and M. Marcu
"Politehnica" University of Timisoara
Timisoara, Romania
bogdan_cyboy@yahoo.com, mrandy1986@yahoo.com
stangaciu@gmail.com, certejan@gmail.com, mmarcu@cs.upt.ro

Abstract— **The environment sustainability is one of the main direction of intervention for all developed counties. Solutions for alternative energies generation and energy reduction are already discussed. Electronic domestic and office devices are an important part of energy consumers. With our work we try to achieve a very cheap and easy to implement solution for power consumption monitoring and analysis of electronic devices. In this paper we present our proposed design and requirements for such a solution.**

Keywords-power consumption monitoring, intelligent plug sosket, wireless communication

I. Introduction

Electric and electronic devices are present in every location and they are the main power consumers in both home and office buildings. Even in stand by mode, they usually consume energy, but this energy is not accounted anyhow. On the other hand, advances in semiconductor technologies and wireless communication are contributing to the rapid growth of the mobile devices market and sensor networks applications. They have a large number of applications but all of them depend on their battery lifetime. Therefore, we propose in this project a new solution that can be used to profile power consumption of both electric power connected devices and battery-powered devices running different applications.

One of the most important evolution directions of mobile as well as traditional computing and electronic systems during the last years are oriented towards power efficiency, either because of the limited battery capacity of these devices or in order to save energy. Among many others, Gartner made this assessment while publishing the "Top 10 Strategic Technologies for 2008" [1]. Gartner's reports show that IT is responsible for 2% of carbon dioxide emissions, which can be simply lowered by half by reducing the processing core frequency by 20% and adding processing cores. The European Parliament recently discussed about new standards for stand-by power consumption of electronic devices, therefore we have to admit that mobile devices power consumption is one of the most important and actual problem considering they will be used in many new applications [2].

A. Proposed solution

A system to monitor the electrical consumption of individual devices in a home or office building could help people make decisions that are more informed on how to alter their usage patterns and choice of electrical devices in order to conserve electricity [3]. We designed and implemented a better way for individuals or companies to see how much electricity each of the devices in their homes and offices consume, and get real time feedback on the effect on their overall consumption of changing their use of these devices. The idea can be seen most clearly by imagining an electric bill with a line for every individual device that details how much electricity that device used over the monitored period and shows a graph of the power consumption through time. This detailed picture would help users better identify those devices or usage patterns that are leading to needless electricity consumption. An electronic and computing device-level monitoring system would also be useful in communal environments, such as schools, laboratories, or businesses.

The power consumption measurement involves digital specialized devices to be connected to the monitored device's power supply line. Then, the data recorded by these measurement devices has to be put together, manipulated and analyzed in order to produce viable conclusions regarding the consumption. The option of using a PC to interface these measurement devices and process the data they produce, although possible, is not recommended. One reason is that the interfacing with a large number of devices is a tedious task and raises some limitations. Another reason would be that the processing power of an average present-day PC is much higher than needed in this instance. All these facts translate into an inefficient usage of resources, especially an unnecessary waste of electrical power. That is why an embedded device would better fit in the implementation of this project. By using embedded technologies, the measurement devices and the embedded processing unit can be organized into an ad-hoc network by attaching to each a networking module and designing an interface between the two. In this way, the power needed is much lower than in the case of a regular PC, while assuring enough processing power by using the embedded device.

B. Other existing solutions

Although there are a number of options for measuring power consumption at the level of a total building (the electric meter being the most obvious), a cheap scalable option for measuring power consumption of each device individually does not exist [3]. The current way one can measure power consumption of an electric system is to use an on-the-self device like Kill A Watt [4] or Watts Up Pro [5]. Both of these

978-1-4244-3878-5/09/$25.00 ©2009 IEEE

features make it difficult to develop a practical real time system specific picture of energy usage that contains many individual devices because the user must go to each monitor in order to record the data [3]. The author of [3] proposed an electrical power monitoring system containing distributed units that transmit power consumption data wirelessly via RF radios to a central base station.

The authors of [6] proposed to implement a virtual instrument for the electric power quality monitoring aiming to act in real-time for detecting, monitoring and recording all typical disturbances superimposed on the ideal signal. Their goal is to extract the voltage and power quality parameters for the power distribution network.

II. OVERALL EMBEDDED SYSTEM ARCHITECTURE

From an architectural point of view, our system is fairly simple. There are three main components which interact in order to fulfill the purpose of the solution. These parts are: a wireless sensor network, a central unit and a web server.

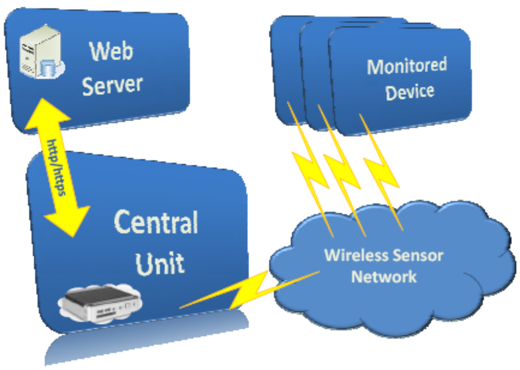

Figure 1. Overall system architecture

The sensor network is responsible for acquiring power measurements from the targeted devices and transmitting them safely to the central unit. There are two types of nodes in this network: the coordinator, which is connected to the central unit and end-points, which are connected to the monitored devices. An endpoint sensor node is equipped with a wireless communication device, a power measurement device and a micro-controller which controls the activity locally.

The central unit consists of the eBox 4300 embedded device, which has the wireless network coordinator attached through the serial COM port, in order to link it to the sensor network. It is here that all measured data is gathered, analyzed and structured in order to make it easier to access. It offers a visual interface for the setup process of the system, when all the devices that need monitoring are registered with the application and the functioning parameters are set. These parameters consist of an individual identification number and name for each device, the connection parameters for the link with the data server and the client account details.

The third unit of the solution is a server. Its main responsibilities are to store the measured data from all the clients of the service, and to make them available for analysis. In order to complete these tasks, the server has two main parts: a set of Windows Communication Foundation services for the communication with the central units and an ASP.NET application that provides the user interface. Additionally, a user account control system is implemented in order to increase security, such that not to allow one client access to another client's data.

Seen from another perspective, the system can be split into software and hardware. Each of these components exists in all the units discussed above, and has an architecture that maps over the one depicted in Figure 1. For a more detailed description, each component is taken separately.

In Figure 2, we designed a simple architecture for the hardware of the project. This design is built around the eBox which is considered the central embedded element of the system. This is a ready built device and no hardware alterations were done to it. Together with the sensor network coordinator, the eBox forms the central unit. The interface between the two is implemented through a serial COM port. Also, the components of a sensor node are specified in the figure. These are the Xbee module, an ATmega16 microcontroller and the measuring device adapted to the type of element which needs measuring. As far as the server is concerned, the hardware represents a normal Windows Server 2008 machine. The custom hardware components used in this project are further detailed in Section III.

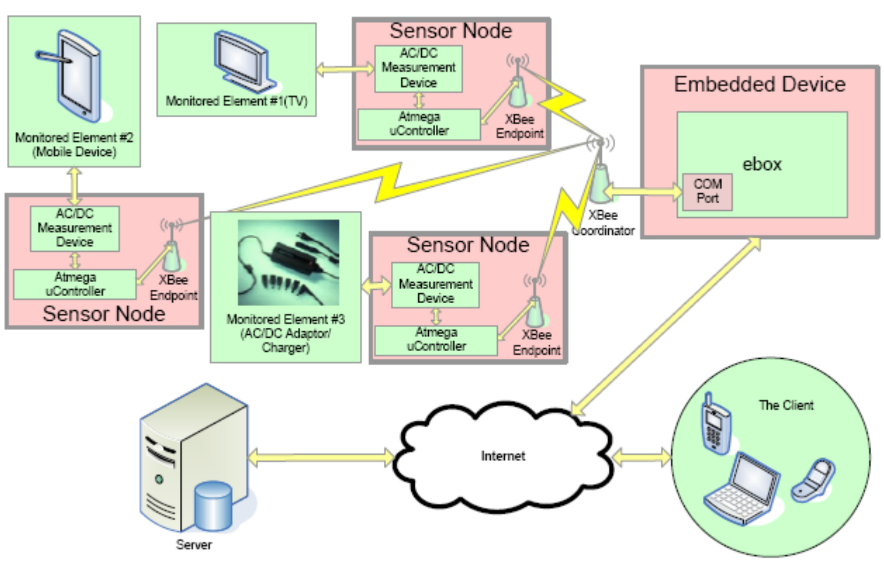

Figure 2. Overall hardware architecture

The software part of the system is described in Figure 3.

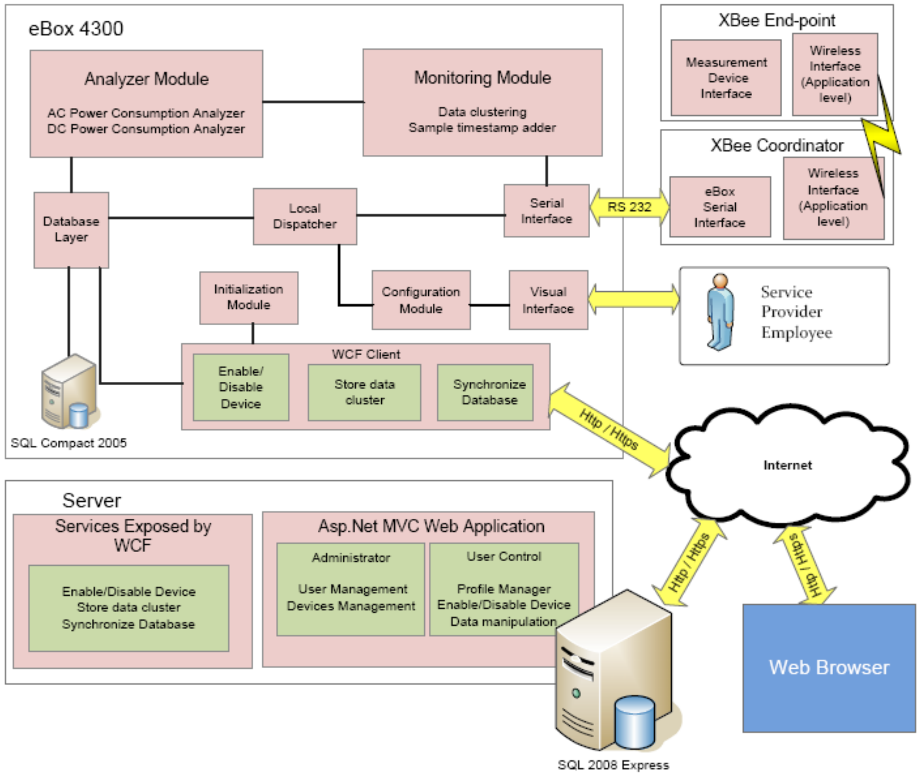

Figure 3. Overall software architecture

As mentioned before, the various components follow the general system architecture, with the code on the eBox being the one offering the main functionality. The applications written at different levels of the system have distinct characteristics which are described bellow.

First, we have the low level code that is written for the microcontroller in the sensor nodes. There is a separate part that regards the coordinator. This consists of the serial communication protocol for the connection with the eBox. The code on the endpoints differs from the one written to the coordinator, because the controller on the endpoint has to communicate with the measurement device and read data from it. So, the serial interface is replaced by the interface with the devices. Both types of sensor nodes implement a communication protocol on top of already existing layers.

Second we have the software on the eBox, build on top of a custom Windows Embedded 6 Image. The eBox gathers all data from the monitored devices and stores them using the compact version of SQL Server 2005 for embedded devices. In this way, the upload to the server can be managed separately and does not depend on the speed the data is collected from the devices. The serial interface implements the same communication protocol as the coordinator and represents the entry-point for the data into the eBox. A monitoring module aggregates data into clusters of measurements coming from the same source and adds them a timestamp.

Then a mild analysis is done to remove inconsistent data from the clusters, such as misreads or values that could not practically exist, but are reported due to specific events in the power line, especially in alternative current. After the analysis the data can be stored into the database. In this state, the data is structured into chronological data samples from the same source, and each source has its own data.

The security measures taken at this point are described next. First, the communication through the WCF is done using https with a self signed certificate, for the moment. More important, at each access from the eBox to the server, the registration protocol is used, meaning that the username and password stored in the data base of the eBox are transmitted to the server. Only after the data is checked, the server allows the communication to start. If the provided username and password are found in the database on the server, but the eBox serial number is not present, the system assumes that a new eBox was added for the specified user, and registers it.

Finally, there is the software running on the server. This is split into two parts. One consists of the discussed WCF Services: synchronization and data storing operate with the local SQL Server, while the user parameter setting transmits to the eBox the user preferences. The second part is a web application that addresses the users of the system.

III. Hardware Components

There are different hardware components and modules used throughout the system. From these, some are bought and integrated in the system, while others are built in-house using custom circuitry and development boards. The decision for each component concerning this aspect was taken considering costs, availability and functional reliability of the solutions on the market compared to what could be built by us.

The major work in terms of hardware was done at the sensor nodes. Each node has three major components, as mentioned before, and a circuit board that provides the proper connections between them, as well as the necessary power supply lines.

The wireless solution chosen for implementing the sensor network is based on XBee modules [7]. These are off-the shelf products and come in two editions: standard and Pro. We chose the standard XBee modules because their characteristics fit with our project and also come at a lower price. These modules are able to form a network between them as long as one device is programmed as the network coordinator. Another important aspect is that router nodes can be set up and integrated in the network, for situations in which the communication distances between two nodes exceed the specifications. This allows for covering a larger surface from a working location without adding extra control units, thus lowering the costs and simplifying system architecture. The access to these devices is provided through a standard full-duplex serial interface, where data can be read or written. They need a power supply of 3.3 V and their power consumption is reduced due to the usage of sleep modes, which are entered when the device is not needed. The module then wakes-up when it detects certain stimuli, like wireless or serial communication attempts.

Over this network we implemented an automatic registration process. New devices need to be detected only when installing the system and the access to the eBox is assured by an employee of the service provider. He issues a command to detect devices. This command sends a broadcast message to all the nodes. The program written on each node transmits back the same data when it receives this type of message. In this way the running devices are detected and then processed in the system.

Another component of a sensor node is the power measurement unit. This varies depending on the type of the monitored device, but once a node is built for a specific device, it is not meant to be reconfigured to work with another device. Allowing this flexibility would mean that each node has to be equipped with components covering all types of monitored devices. This would raise both the manufacturing and running costs of such a node, making it unfeasible, especially because the nature of our service does not imply changing the monitored device attached to a sensor node throughout the operating period.

Measuring the power in AC devices implies measuring the current on the hot wire. In order to safely access the electrical wires, the AC measuring device, together with the rest of the components in the sensor node are inserted in a case that is plugged into the existing AC socket and offers another socket in which the monitored device is plugged. In between there is a circuit that measures the analogue current and has a digital output of the measured value. The sensor node is presented in Figure 4, together with the schematics of the integration board. The actual measurement of the current is done by using a shunt resistor R, and by measuring the voltage over that resistor we can deduct the value of the current as I=U/R. The voltage value is directly available and needs no special elements. In order to obtain a digital value for these parameters, their signals have to be brought to the ADC on the micro-controller. But the ADC does not accept high voltages as inputs, so these signals have to be scaled down. This is done by using a voltage divisor which allows for scaling the voltage interval to [0÷5 V]. This new scaled analogue signal is inputted in the ADC and converted to a digital value. That is how a set of two values is obtained, one for the current and one for the voltage. The next step is normalizing them to obtain the real values, but this step is handled by the software on the microcontroller.

Figure 4. AC sensor node

The last component of a sensor node is the ATmega16 [8] microcontroller. This was chosen due to low cost and great hardware equipment that suited the needs of our project. The controller connects the XBee module and the measurement device and handles the activity of the sensor node. Data packets are transmitted through the XBee at regular 5 minute intervals. This value was found to provide a good balance between resource usage and data accuracy. The packets contain only one value for the current and one for the voltage, which represent the mean value computed from all gathered samples in that interval.

An important aspect is the data gathering algorithm implemented through this controller. According to the electrical engineering theory of power consumption, the active power a circuit element consumes is obtain as P_A=U*I*cos f , where *f* is the phase angle between the current and the voltage. In the case of passive consumers, this angle is 0, and a simple product between the voltage and the current gives the active power. But in the case of active consumers, i.e. devices with capacitors or inductances in their architecture, this angle modifies. That is why a different formula, which considers the effective values of the current and voltage in several points of a period, is used. This formula is:

$$X = \sqrt{\frac{1}{T}\sum_{k=1}^{n} X_k^2 \Delta t},$$

where X can be either the current I or the voltage U. The power over a period T results by multiplying the two resulted values. To apply the formula, n measurements are taken over a period of the sinusoidal signal. For Europe, where we have an AC power supply of 220V and 50 Hz, a period T is 1/50Hz = 20ms. This period is split into n-1 intervals for n measurements, so $\Delta t = T/(n-1)$. All these calculation are done on the microcontroller and thus the power value is obtained.

In order to reduce the power consumption of our system, we used a dynamic ADC sampling rate. This means that the interval between two samples changes continuously and its value is controlled by the variation of the samples. The controller retains the last two power values acquired in the last two periods, in order to determine the trend and variation of these values. The two samples are compared and the variation percentage is computed as $V_p = \left(\frac{S_1}{S_2} - 1\right) * 100$. The resulted value can be positive if an increase is shown, or negative, if a decrease has happened. That is why the absolute value of the result has to be retained. If this percentage is above a threshold (we chose 20 %), then the sample rate is increased, providing more samples per second. If, on the other hand is lower than another threshold (5%), then the rate is lowered to allow less usage of the measuring device. Obviously, if the value is between the two thresholds, the sampling rate remains unaltered.

IV. DATA FLOW AND PROCESSING

The main information flows through the system from the sensor nodes to the server, but it passes through several checkpoints along the way. For a better illustration of the routes on which the data circulates in the system, we have to consider several scenarios. Below there is a diagram for the flow of a sample of voltage or current from when it is acquired until it reaches the user. The actual value is obtained by the measurement device and passed to the ADC in the micro-controller. Then a mean value over a period of five minutes is sent to the eBox trough the XBee modules. The eBox filters the data, adding a timestamp and then organizes it into cluster for storage in the embedded data base. From here it is uploaded to the server using WCF. When the user wishes to view the information gathered by the system, it uses an internet browser to log in to the server, where the web application will provide the needed data. It is clear from the diagram that the flow of the data is linear, without loops or branches.

A second scenario consists of the case when the user enables or disables a device. This path is similar with the one before, aside from the fact that the direction is reversed and the meaning of the data is different. When the user changes the state of the device, it determines a change in the associated field of the table that contains the devices. This table is then synchronized with the one on the eBox through WCF. After this step the eBox reads the state in which the devices need to be and issues commands to the nodes accordingly. At nodelevel, the controller receives the message and turns the measurement device on or off depending on the received commands.

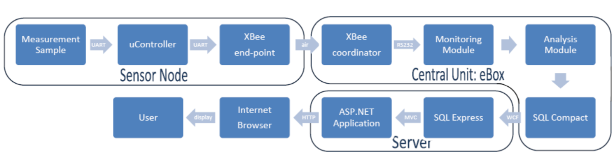

Figure 5. System data flow in the acquisition scenario

The communication protocol used in the wireless sensor network was developed to minimize packet length and assure a fixed length and format for all types of messages. Each packet circulating the network carries a message and has the following structure:

Figure 6. Packet architecture for wireless communication

The source and destination addresses are at the same time the unique identification number of each XBee Module and each one represents a hardcoded 64-bit serial number. The next byte serves for mapping the type of the message. The message can be a sample reading, a command or a registration notice. Message data can consist of the current, voltage and power readings in this order (each on 16 bits) for a reading message. The data field of a command message can be 1 for enabling the device, or 0 for disabling it. Finally, the registration message contains no data, because the source address and the message type code are enough to determine that a specific device has registered into the system.

There are several data processing points in our system, divided accordingly with the available processing power of that component. The first one is at a sensor node level. Here the coarse measurement of the electrical parameters is transformed into a usable reading with the algorithm discussed in previous section.

The eBox formats the received measurement in the sense that it forms an entry for the measurements table in the data base. In order to do this, the source address from the packet is extracted and translated into a Device ID. Then the message type is checked. In the case of a measurement message, each value is extracted from the packet. The timestamp is added at this level and a sample number is assigned. All these pieces of information constitute the entry fields and thus the entry is inserted in the database.

The messages containing the registration of a new device are treated differently. The source address is extracted and a query is issued to see if the device is already in the system. If it is not, a new device entry has to be inserted. The location and eBox SN are both values that the eBox stores for itself in global variables and are already available. As mentioned before, these messages appear only at system installation time and thus the access to the eBox is enabled. When this message is received, the eBox displays a form for adding the devices. The mentioned fields are already completed. It is only the device name that needs to be entered. After all information is collected, a new entry is inserted in the device table.

V. SYSTEM RESULTS AND PERFORMANCE ANALYZIS

The testing of a system with this many components is a tedious process. That is why we developed the project in increments that allow for testing smaller isolated components before integrating them together into bigger subsystems and testing the resulted unit, until it came to testing the system as a whole.

The hardware testing was necessary only in the sensor nodes, since the rest of the components are commercial products tested by their respective producers. Before physically linking all the components of a node together, we developed a schematic of the node in pSpice and ran some simulations to assure that when the node is connected to the AC power line, its components are not damaged. Also the voltage divisor and shunt resistor were analyzed to see if they meet the requirements. After the design was validated, the board was built and tested again to see if there are components that heat up over time. The same process was followed when attaching the XBee coordinator to the eBox.

The communication protocol in the wireless network was tested separately at first, meaning that the message handling or generation codes on the eBox and on the nodes was tested separately using a serial terminal. Through this terminal messages could be sent and received and then checked for correctness. After both sides of the communication acted accordingly to the specification, they were connected and tested as a whole.

The software testing needed the most attention. The code for handling the XBee communication was reused from previous projects and applications, and thus it was tested during the functioning in those instances. The low-level code running on the micro controller was tested manually by isolating each function and providing several sets of arguments in order to exercise as much code as possible.

On the main server the application can show different parameters for the observed devices: voltages, current consumption and power consumption. These parameters are recorded in real time and they can be further used in different reports and graphics. Power consumption for a desktop computer running some applications is presented in Figure 7. In this graphic the power variations related with the applications running in the system can be observed.

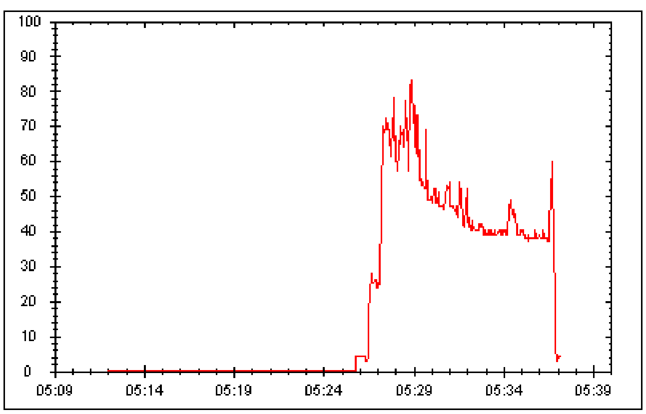

Figure 7. Power consumption graph

The performance criteria by which we benchmarked our system refer to specific metrics depending on the component being analyzed. The system as a whole has to be accurate, fast and scalable. During the tests, we observed the behavior of our system, paying special attention to faults occurred in the components of the system.

The communication distances between two XBees were measured. In open space these were of about 30 meters. When the modules are separated by a 25 centimeters thick concrete wall, the communication between them is interrupted. On the other hand, a thinner wall, of 15-20 centimeters, only shortens the distance to about 10 meters. Glass doors do not affect the communication in any way, which is a good thing especially when using the system in office buildings.

Regarding the transmission times, these are satisfactory, especially because we need to transmit a sample from a device every five minutes. This allows for enough time to gather measurements from all devices present and due to the processing power of the eBox, the samples are analyzed and stored in time, such that the workload on the eBox is not at constant high levels.

VI. CONCLUSIONS

In order for an environmental sustainability to be ensured, the least we can all do is to use the natural resources efficiently. Reducing power consumption or using the power in a more sustainable way affects our planet in two ways. The electrical energy production implies more or less CO emissions, depending on the source. As an example, the major sources of energy in the United States in 2008 were coal and nuclear, with almost 70% of the total [9]. So if we implement an efficient consumption of the electrical power, we do the same tasks with less energy, meaning we produce less pollution. By applying the same principle that we can do the same with less energy, we reach the conclusion that planning for efficient power consumption slows down the rate of resource consumption.

ACKNOWLEDGMENT

This work was supported by Romanian Ministry of Education CNCSIS grant 680/19.01.2009.

REFERENCES

[1] Gartner Inc, “Top 10 Strategic Technologies for 2008” http://www.gartner.com/.

[2] European Parliament, “Action Plan for Energy Efficiency: Realizing the Potential”, INI/2007/2106, Ian. 2008.

[3] Jerey LeBlanc, “Device-Level Power Consumption Monitoring”, http://www.awarepower.com/ubicomp07.pdf, 2007.

[4] Kill-A-Watt: www.p3international.com

[5] Watts-Up-Pro: www.doubleed.com/products.html

[6] F. Adamo, F. Attivissiomo, G. Cavone, and A.M. Lanzolla, “A Virtual Instrument for the Electric Power Monitoring in the Distributing Network”, 15th Symposium on Novelties in Electrical Measurements and Instrumentation, Romania, 2007.

[7] Xbee Wireless Modules Specifications http://www.digi.com/products/wireless/point-multipoint/xbee-series1-module.jsp

[8] ATmega16 Data sheet, http://www.datasheetcatalog.org/datasheet/atmel/2466S.pdf.

[9] Energy Information Administration, Electric Power Monthly February 2009 Edition http://www.eia.doe.gov/cneaf/electricity/epm/epm_sum.htm