2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops

A Prototype Wireless Sensor Network for Precision Agriculture

Jonathan Jao, Bo Sun*
Dept. of Computer Science
Lamar University
Beaumont, TX 77710
fondyjao@gmail.com, bsun@lamar.edu

Kui Wu
Dept. of Computer Science
University of Victoria
Victoria, British Columbia, Canada V8W 3P6
wkui@cs.uvic.ca

***Abstract*—Wireless Sensor Networks (WSNs) have become the ideal candidate to provide effective and economically viable solutions for a large variety of applications ranging from health monitoring, scientific data collection, environmental monitoring to military operations. In this paper, we present a proof-of-concept WSN to collect soil moisture content, which is one of the most fundamental data required for precision agriculture. Leveraging existing off-the-shelf hardware (*MicaZ* motes, *MDA300CA* data acquisition board, and *EC-5* soil moisture sensors), the dominant open source embedded operating systems software (*TinyOS 2.1.1*), and *MViz*, we have built a prototype WSN to collect soil moisture. We present the detailed design and implementation of *MDA300CA* driver for *TinyOS 2.1.1*. Note that our architecture is general, so it is easy to integrate the driver of other sensor probes, thus collecting more types of data for different research purposes. Limited battery supply is a major concern when utilizing WSNs to build realistic applications. Therefore, we utilize solar panels and rechargeable battery to address this challenging problem. We illustrate our detailed design to package WSN nodes for outdoor development and present collected data to validate our design. Specifically, using sand soils with different water content, we demonstrate the result of our system. With enough details for researchers to understand and thus improve our design, our system could be a good starting point for their own research.**

***Keywords*-Wireless Sensor Networks, TinyOS, MicaZ, MDA300CA Driver**

I. Introduction and Motivation

Wireless Sensor Networks (WSNs) have become ideal candidates to provide effective and economically viable solutions for a large variety of applications ranging from health monitoring, scientific data collection, environmental monitoring to military operations. Although much research has been carried out in various aspects of WSNs, unfortunately, it is still a daunting task to customize low-cost and efficient WSNs for multiple cross-disciplinary domains.

Most existing work in WSNs addresses their fundamental challenges, including power supply, limited memory, processing power and communication bandwidth and focuses entirely on their operating system and networking protocol design and implementation. However, it is not easy to find the technical details on how to make a wireless sensor node work well in an outdoor environment. Although some academic research products are available, these products either adopt uncommon wireless modules, or obscure the details of how to design a node to collect sensor values for a specific discipline. Commercial products, such as *eKo* systems from *Memsic* [1], are available. Unfortunately, their implementation is not open source and it is therefore difficult for researchers to understand the details of how to obtain data. In the meantime, the lack of documentation and coding styles resulting from its design principles (resource use minimization and bug prevention) make it rather challenging to provide an open source solution based on *TinyOS* [3]- one of the most dominant operating systems for WSNs.

The open source nature of *TinyOS* indicates that most applications are voluntarily contributed by its users. *CrossBow* (acquired by *Memsic* in 2012) contributed a *MDA300CA* driver based on TinyOS 1.x. However, they did not continue to release a *MDA300CA* driver for TinyOS 2.x. A close look at TinyOS mailing list indicates that this has created a lot of problems for research community using TinyOS to build their applications, because MDA300 provides a versatile data acquisition platform which is compatible with *MicaZ/Iris* [16] motes. In the meantime, it is hard to repeat experiments based on *MDA300CA* drivers provided by voluntary users in our lab environment. Like a black box, *MDA300CA* drivers involve understanding hardware working mechanisms, numerous low-level operations and nontrivial debugging techniques. TinyOS mailing list has only very limited discussions about *MDA300CA*, which makes it challenging to understand its working details. *MDA300CA* drivers are necessary for most realistic scientific WSN applications. For example, it is greatly benefited to utilize WSNs to collect a large amount of *in situ* soil moisture data in precision agriculture. Without a proper *MDA300CA* driver, it is impossible to fully realize the great potential applications promised by WSNs.

Motivated by these observations, we present a proof-of-concept WSN to collect soil water moisture, with enough technical transparency to help others follow our design. Leveraging existing off-the-shelf hardware (*MicaZ* motes, *MDA300CA* data acquisition board, and *EC-5* soil moisture sensors), the dominant open source embedded operating systems software

*This material is based upon work supported by the National Science Foundation under Grant No. 0922888. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

978-0-7695-5023-7/13 $25.00 © 2013 IEEE
DOI 10.1109/ICDCSW.2013.10

(*TinyOS 2.1.1*), and *MViz*, we have built a prototype WSN to collect soil moisture. We also show how those *components* in *TinyOS* are *wired* together. Note that our architecture is general, so it is easy to integrate the driver of other sensor probes, thus collecting more types of data. We also present a suitable design to enclose WSN nodes so they can be deployed in a outdoor environment. We utilize *MicaZ* motes from *Memsic* to carry out single node experiment and multiple node experiment. Results demonstrate that the *MDA300CA* driver can work well under *TinyOS 2.1.1*.

The following of the paper is organized as follows. Section II presents relevant work. Section III presents the details about hardware, software, and wireless sensor network prototype setup. Section IV presents preliminary experiment results in a lab setting. Section V concludes this paper and points out future work.

II. Literature Review

A. TinyOS

TinyOS is an open source operating system built for the research and development of wireless sensor network applications [3]. It allows researchers to build and test applications on a wide variety of sensor node hardware.

TinyOS applications are written in the *nesC* programming language [3]. *NesC* is an extension of the C programming language and was designed to enhance the core concepts and execution model of *TinyOS*. *NesC* is an event driven language that organizes logic code into modules and connects those modules via interfaces. Modules in nesC are similar to classes in the *Java* programming language. All communication between nesC modules must go through defined interfaces. *NesC* interfaces specify which of a module's functions are to be made public to other *nesC* modules.

B. Outdoor WSN

Outdoor WSNs have been deployed in a wide range of scenarios and always been reported as a challenging task. They have been applied to monitor active volcanoes [5], pipeline infrastructure [6], redwood trees [7], precision agriculture [8], sea monitoring [9], environment observation and forecast system (CORIE) [10], ZebraNet [11], patient monitoring [12], groundwater transport models [13], and ecology [14]. Unfortunately, no details are revealed about how to build these networks. It is still challenging for other researchers to utilize these research for their own research work.

Yang *et al.* [15] presented the details about environmental monitoring systems using wireless sensor networks. Their systems utilized *MDA300CA* and their sensor nodes are enclosed in an outdoor environment to obtain sensor data. However, their system is based on *TinyOS 1.x*, a previous version of *TinyOS*. Because the majority of sensor network research is now based on *TinyOS 2.x*, which is significantly different from *TinyOS 1.x*, it is therefore necessary to upgrade traditional *TinyOS 1.x* code to *TinyOS 2.x*. Our system borrowed help from their work. We also make significant modifications.

III. A Prototype Wireless Sensor Network for Precision Agriculture

We present the detailed design and implementation of our prototype Wireless Sensor Network with a focus on precision agriculture in this section. Specifically, we illustrate hardware, application software and kernel software, networking protocol, and how to make assembled wireless sensor node suitable to be deployed in an outdoor environment.

A. Hardware

We use the popular off-the-shelf *MicaZ* motes coupled with *MDA300CA* data acquisition board manufactured from *Memsic* [1] and *EC-5* [17] soil moisture probes from *Decagon*. *MicaZ* motes represent the third generation wireless modules for low-power sensor networks. Its *CC2420* transmission modules follows IEEE/ZigBee 802.15.4 standard and has a 250kbps data transmission rate. *MicaZ* processor is based on low-power microcontroller *ATmegal128L* [18].

MDA300CA is a versatile data acquisition board with multifunction direct user interface. It has up to 11 Channels of 12-Analog inputs and onboard sensor excitation. *MDA300CA* is compatible with *MicaZ* motes. The combination between *MDA300CA* and *MicaZ* through its 51-pin expansion connector enables a wide range of applications.

EC-5 is a sensor probe that can measure water content in any soil media with minimal salinity and textural effects. It determines Volumetric Water Content (VWC) by measuring the dielectric constant of the media using capacitance/frequency domain technology. Compared to other soil moisture sensor probes, *EC-5* is easy to install and has a relatively low price.

EC-5 is compatible with *MDA300CA* in terms of data acquisition in that *MDA300CA* provides a voltage excitation to *EC-5* and then obtains the readings. *MicaZ* nodes then package this information and transmit the data to a base station through networking protocols.

B. Software

We use *TinyOS* because it is the dominant open source operating system for embedded devices. Many research projects have been designed and implemented based on TinyOS due to its popularity. The two major principles followed by TinyOS designers, resource use minimization and bug prevention principles, help TinyOS achieve near-optimal RAM overhead while enabling large and complex software systems [2]. However, these principles also lead to software complexity and many fine-grained components in a single TinyOS application resulting in the linking of many tiny pieces of codes spread across many files. These make it rather challenging to develop third-party codes under TinyOS.

Numerous research efforts have been carried out aiming at an updated *MDA300CA* driver for TinyOS 2.x to enable various WSN applications. Among these efforts, Charles Elliot may provide the relatively most complete *MDA300CA* driver for TinyOS. Unfortunately, a close look at TinyOS mailing list reveals that many researchers are unable to make this driver work successfully. There are also difficulties to obtain readings from the data board as the drivers written by Charles Elliot were incomplete. Because the original *MDA300CA* driver provided by Charles Elliot was based on TinyOS 2.x, we decided to continue our work based on his work.

It is always non-trivial to design and test a device driver, especially for embedded system, in which low-level operations are involved. The complex interactions among TinyOS, hardware and external devices make it challenging to debug and test the software. Special devices are needed to observe the excitation from the *MDA300CA* to the *EC-5* probe.

After extensive analysis, we place *10ohm* resistors on the acquisition board to enable specific voltage transfers allowing the sensors to be read. Through extensive research and experiments, we obtained the moisture sensor functions through the use of a constant voltage excitation, the attachment of the ground wire for use as a reference for the *EC-5* itself, and a red wire to feed back the changed voltage. The voltage is then read by TinyOS which passes it through a normalization equation and finally calibrated. The hardware connection between *EC-5* and *MDA300CA* is illustrated in Figure 1. Figure 1 also indicates how resistors are utilized in *MDA300CA*.

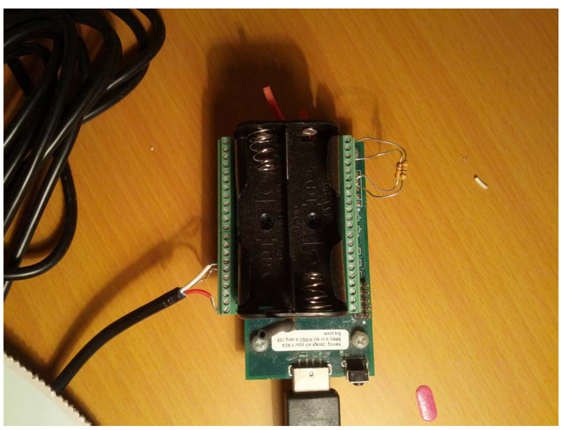

Fig. 1. Hardware Connection.

Although the process itself is relatively straightforward, due to the lack of documentation for the contributed driver, we are faced with quite a bit of difficulty obtaining correct readings. One of the challenges in the process of implementing the *MDA300CA* driver is the inaccurate and inconsistent readings obtained from *EC-5*. Because of many factors are involved in this process (hardware, low level driver, and high level TinyOS application) and the difficulty of debugging TinyOS application, we have to resort to various techniques to check. We first used an oscilloscope to check whether the *MDA300CA* produces a 2.5 voltage excitation. One pin of *MDA300CA* E2.5 produces a 2.5V excitation when it operates normally. *EC-5* probe generates its output when detecting this 2.5V excitation. The oscilloscope showed an occasional spike in voltage which was likely noise from the bare wire.

We then looked for a solution, and later found that the voltage excitation had to be enabled through a bus on the board. The following code snippets demonstrate this: $MVizC.TWO\_HALF\_VOLT \rightarrow MicaBusC.PW2$, $call\ TWO\_HALF\_VOLT.set()$. The voltage excitation is turned on in $Boot.booted()$. After that was enabled, readings were passed back and found to be extremely consistent with Volumetric Water Content.

Our design and implementation employs Oracles's *VirtualBox* [19]. TinyOS was installed in the virtual machine based on *VirtualBox*. All the applications and drivers were installed and tested within the system.

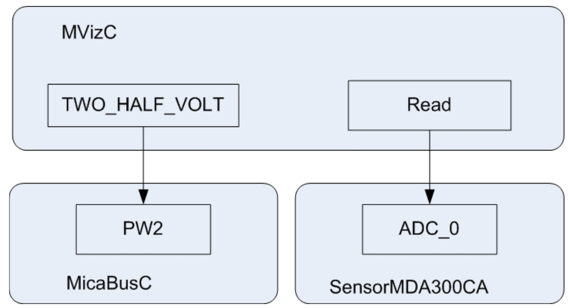

Fig. 2. Application Wiring.

We utilize *MViz* to start building our application. The high level wiring scheme is illustrated in Figure 2. *ADC_0* is the interface from which the sensor data is read. *MicaBusC.PW2* provides the functionality to initialize the voltage excitation which is wired to *TWO_HALF_VOLT* excitation. We skip other irrelevant parts of *MViz* in order to focus on the *MDA300CA* part. The purpose of *MicaBusC* is to provide an interface to turn on the voltage excitation. *TWO_HALF_VOLT* in *MViz* application utilizes *MicaBusC* interface to enable voltage excitation. The *Read* interface is wired to *ADC_0* in *SensorMDA300CA*. The read operation is a split-phase operation. After obtaining the *EC-5* reading, we integrate this driver into *MViz*, a popular application to set up a self-organized multihop WSN based on TinyOS. We modify the *Read()* function in *MViz* and wire it to *MDA300CA* driver.

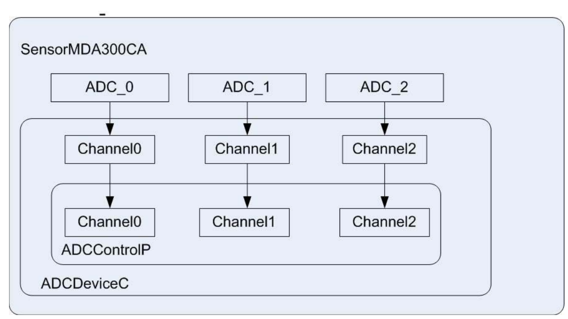

Fig. 3. *MDA300CA* Wiring.

The high-level description of component *SensorMDA300CA* is illustrated in Figure 3. The *Channels* are provided as *read()*

interfaces and the line arrows represent references to inner component interfaces.

In *MViz*, all the nodes are connected through Collection Tree Protocol (CTP) [25], a tree-based collection protocol through which all sensor nodes form a self-organized network. TinyOS provides an open-source implementation of CTP. Therefore, it is desirable for researchers to continue their research based on CTP.

After the raw readings are obtained, they are calibrated through normalization equation *VWC (m3/m3) = 0.00119 * millivolt - 0.401*. Here m_3 denotes cubic meter. *millivolt* is the raw reading from the *MDA300CA* driver. An example is provided in Section IV.

We have also made our TinyOS code available online. Interested users may download them for their research purposes. *MVizMoistureSensor.tar.gz* (available at http://galaxy.cs.lamar.edu/~bsun/mda300/MVizMoistureSensor.tar.gz contains the complete *MViz* application which integrates *MDA300CA* drivers. *MoistureSensorv1.1.tar.gz* (available at http://galaxy.cs.lamar.edu/~bsun/mda300/MoistureSensorv1.1.tar.gz) contains the normalized equation using an in-program formula.

C. Packaging Sensor Node for Outdoor Development

Many realistic WSN applications like precision agriculture require deploying sensor nodes in an outdoor environment for a long time. We present our initial efforts in this respect. This is an ongoing process, as better products will be integrated into our systems when they are available.

Energy is one of the most challenging issues for WSN deployment. After investigating and trying numerous types of batteries, we finally decided using *LiFePO4 18650* battery. It is a Li-Ion rechargable battery module and is made of two pieces. LiFePO4 18650 Battery is one of the most popular batteries used so far. *LiFePO4 18650* provides a voltage of 3.2V and has a capacity of 3000 mAh, which meets the requirement of *MicaZ* and *Iris* mote. It has a max charging current of 7A with the rate of 9.6Wh [20]. *MicaZ* motes require an external power of 2.7V - 3.3V. The 3.2V provided by LiFEPO4 18650 is within this range. It has low voltages and high cycles. Also, it has a capacity of 3000mAh, which is reasonably large.

To harness proliferate solar energy to charge *LiFePO4 18650* and to provide energy, we resort to solar cells. We have several options when picking up proper solar cells. After investigation, we narrow the selection down to two candidates: 1) Two solar panels SolMaxx OEM rated at 2V and 200mA [21] are serially connected to provide a 4V voltage output; 2) One 4V 200mA solar panel [22] is utilized. It has a 4V voltage and 200mA amperage. These specifications can meet the voltage requirement of *LiFePO4 18650* battery.

To prevent the energy from flowing back to the solar cell during night when no sunlight is available, we use a diode *IN5818* [23], as shown in the figure. We solder them together and put them in an enclosure, as illustrated in the Figure 4. In Figure 4, $+$ denotes positive, while $-$ denotes negative. Superglue is used to glue the solar cells to the inside of the enclosure. Regarding the enclosure, we use a transparent enclosure with a dimension of 7.09 x 4.33 x 3.54 inches manufactured by *Altech* [24].

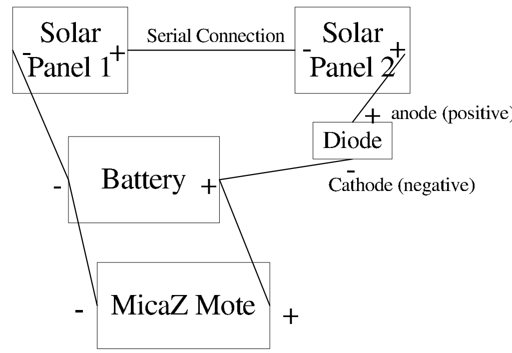

Fig. 4. Using Solar Cells and Rechargeable Battery to Provide Energy for *MicaZ* Motes.

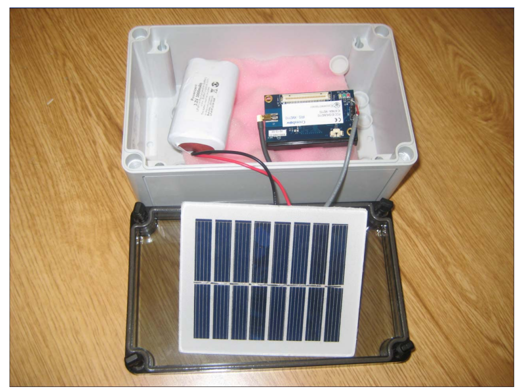

Fig. 5. Enclosing a Sensor Node with Rechareable Battery.

With all these, our assembled node is illustrated in Figure 5.

IV. Experiment

In this section, we present our experimental results. We carried out the experiment to demonstrate the performance of our implemented *MDA300CA* driver. We further utilized multiple nodes to set up a simple wireless sensor network that collects soil moisture data.

A. Single Node Experiment

We use three types of sand soil with different water saturation, namely *Dry*, *Moist*, and *Water-Saturated*. We focus on the *MDA300CA* driver in this research. Therefore, we only give a preliminary preparation of these different types of soil. We first use three cups to hold the same type of soil. To prepare the *dry* soil, we let the soil exposed to the air for two days. To prepare the *Water-Saturated* soil, we pour three cups of water into the cup. To prepare the *moist* soil, we pour one cup of water into the cup. We then insert the *EC-5* probes into each of the three cups, respectively. For all these types of soil, we

Fig. 6. Experiment to Collect Soil Moisture Data.

take the reading once every 15 minutes. We the obtain these readings, as illustrated in Figure 7.

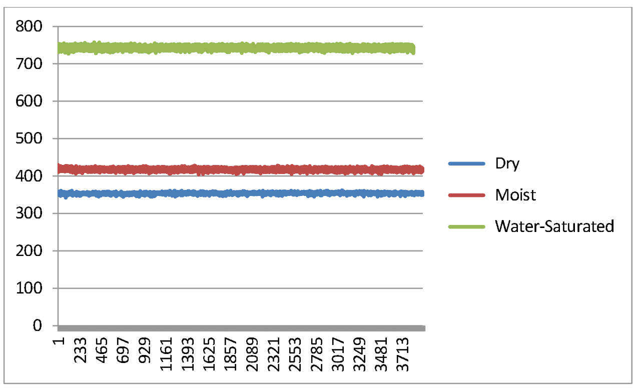

Fig. 7. Readings from Three Types of Soils with Different Water Saturation.

These readings are what we expect. As we can see from Figure 7, when the soil is *dry*, the reading values fluctuate around 355. When the soil is *moist*, the reading values fluctuate around 418. When the soil is *water-saturated*, the reading values flucturate around 743. We also observe an increased standard deviation of the readings when there is more water in the soil, 2.28, 4.10, and 5.50 for these types of soils, respectively. This is because *EC-5* measures the volumetric water content (VWC) in the soil. With more water in the soil, it will have greater variation. Our experiment also demonstrates sensing inaccuracy.

The reading could then be normalized using the equation provided by *Decagon*: VWC (m^3/m^3) = 0.00119 * millivolt - 0.401. For example, if a reading is 385, the VWC should be 0.00119 * 385 - 0.401) = 0.05715. m^3 denotes cubic meter.

B. Multiple Node Experiment

Following the single node experiment, we build a prototype of multihop wireless sensor network based on *MViz* application bundled with *TinyOS 2.1.1*. Its screenshot is illustrated in Figure 8.

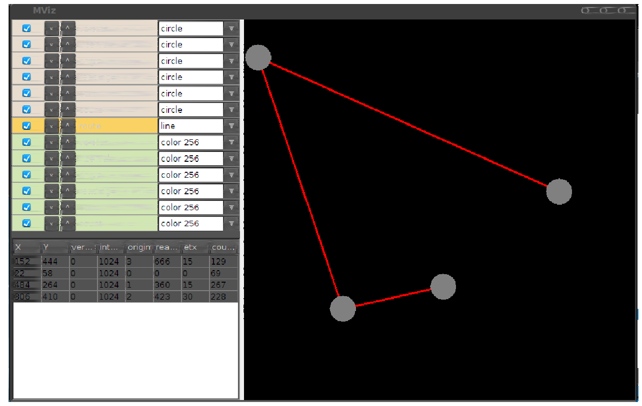

Fig. 8. Screenshot of *MViz* Output.

Figure 8 demonstrates four nodes which forms a multihop wireless sensor network. The left-top box presents the display of the network topology. Readings from each node is shown in the left-bottom box. Network topology is illustrated in the righ panel. Interested readers may refer to *MViz* manual for more detailed information.

V. Conclusions and Future Work

A. Conclusions

Folowing existing work, we present the details of design and implementation of *MDA300CA* driver based on *TinyOS 2.1.1*. We bridge the gap between existing work in this respect and present not only a complete driver, but also all the environment to test this driver. Through concrete examples, we walk out all the details about how to integrate an external sensor to *MicaZ* through *MDA300CA*, the popular data acquisition board. We also present details about how to enclose *MicaZ* and MDA300CA board to make them suitable to be deployed in an outdoor environment.

We make our driver and environment publicly available, so other researchers can duplicate and extend our research. We believe our work will benefit existing research on TinyOS and its application development.

B. Future Work

Working with open source software like TinyOS is always challenging because of the lack of documentation and its ever-changing nature. We plan to improve the stability of the *MDA300CA* driver and systematically test its performance using more scenarios. We also plan to test and evaluate our designed nodes for outdoor WSN deployment.

References

[1] http://www.memsic.com

[2] Philip Levis, "Experiences from a Decade of TinyOS Development", In Proceedings of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI).

[3] http://www.tinyos.net

[4] Gnawali, O., R. Fonseca, K. Jamieson, D. Moss, and P. Levis. 2009. Collection Tree Protocol. *SenSys '09*, New York, NY, 1-14.

[5] Huang R., Song W.-Z., Xu M., Peterson N., Shirazi B., and LaHusen R., "Real-World Sensor Network for Long-Term Volcano Monitoring: Design and Findings", IEEE Transaction on Parallel and Distributed Systems, Volume 23, Issue 2, 2012.
[6] Stoianov I., Nachman L., Madden S., and Tokmouline T., "PIPENETa wireless sensor network for pipeline monitoring", IPSN '07 Proceedings of the 6th international conference on Information Processing in Sensor Networks, pp. 264-273.
[7] Tolle G., et. al. "A Macroscope in the Redwoods", Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems, pp. 51 - 63.
[8] Langendoen K., Baggio A., and Visser O., "Murphy Loves Potatoes: Experiences from a Pilot Sensor Network Deployment in Precision Agriculture", Proceeding IPDPS'06 Proceedings of the 20th International Conference on Parallel and Distributed Processing, pp. 174-174.
[9] Yang Z., Li M., and Liu Y., "Sea Depth Measurement with Restricted Floating Sensors", IEEE RTSS, Tucson, Arizona, USA, December 2007, pp.469-478.
[10] Dang T., Frolov S., Bulusu N., and Feng W.-C, "Near Optimal Sensor Selection in the Columbia River (CORIE) Observation Network for Data Assimilation using Genetic Algorithms", Proceeding DCOSS'07 Proceedings of the 3rd IEEE International Conference on Distributed computing in Sensor Sytems, pp. 253-266.
[11] Juang P., Oki H., Wang Y., Martonosi M., Peh L.S., and Rubenstein D. "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet", ASPLOS-X conference, October, 2002. San Jose, CA, pp. 96-107.
[12] Patel S., Lorincz K., Hughes R., Huggins N., Growden J., Standaert D., Akay M., Dy J., Welsh M., and Bonato P., "Monitoring Motor Fluctuations in Patients With Parkinson's Disease Using Wearable Sensors", IEEE Transactions on Information Technology in Biomedicine, November, Volume 13, Number 6, 2009, pp. 864-873.
[13] Barnhart K., Urteaga I., Han Q., Jayasumana A., and Illangasekare T., "On Integrating Groundwater Transport Models with Wireless Sensor Networks", *Ground Water*, 2010, doi: 10.1111/j.1745-6584.2010.00684.x
[14] John Porter, Peter Arzberger, Hans-Werner Braun, Pablo Bryant, Stuart Gage, Todo Hansen, Paul Au;hanson, Chau-Chin Lin, Fang-Pang Lin, Timothy Kratz, William Michener, Sedra Shapiro, and Thomas Williams, "Wireless Sensor Networks for Ecology", *BioScience*, July 2005 / Vol. 55 No. 7, 561-572.
[15] Jue Yang, Chengyang Zhang, Xinrong Li, Yan Huang, Shengli Fu, Miguel Acevedo, "An Environmental Monitoring System with Integrated Wired and Wireless Sensors", Wireless Algorithms, Systems, and Applications Lecture Notes in Computer Science Volume 5258, 2008, pp 224-236.
[16] http://www.memsic.com/products/wireless-sensor-networks/wireless-modules.html
[17] http://www.decagon.com
[18] http://www.atmel.com/devices/atmega128.aspx
[19] https://www.virtualbox.org/
[20] http://www.batteryspace.com/lifepo418650battery32v2700mahflat864wh7arate.aspx
[21] http://www.siliconsolar.com/oem-solar-panels-p-500962.html
[22] http://www.siliconsolar.com/4v-200ma-oem-solar-panel-p-16530.html
[23] http://www.digikey.com/product-detail/en/1N5818/1N5818FSCT-ND/1626123
[24] http://www.altech.com
[25] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. "Collection Tree Protocol", In Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys), 2009.