Bringing Real World Applications for Wireless Sensor Networks into the Classroom: Telemetric Monitoring of Artificial Streams

RET: Research Experiences for Teachers on Sensor Networks Summer 2013
Zac Bunn, Carrollton-Framers Branch ISD, Michael McEver, Lewisville ISD, and Deliah Seastrunk, Denton ISD
Faculty Mentors: Dr. Shengli Fu, Department of Electrical Engineering, Dr. David Hoeinghaus, Department of Biological Sciences
Research Assistant: Yixin Gu, Graduate Student EE
University of North Texas, Denton Texas

Acknowledgments
Research conducted under RET (Research Experiences for Teachers), on Sensor Networks, Electrical Engineering Department, and Institute of Applied Sciences, UNT, Denton, Texas. This material is based upon work supported by the National Science Foundation under Grant No. 1132585 and by the IEEE Control Systems Society (CSS). 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 or IEEE CSS.

For further information visit
www.teo.unt.edu/ret/
http://untret2.blogspot.com
Abstract
This research report covers the use of a wireless sensor network (WSN) to remotely monitor water levels, pH, dissolved oxygen (DO), and temperature in an experimental stream facility. Field tests were conducted at University of North Texas (UNT) Water Research Facility using an experimental stream facility. Measurements were recorded every 10 seconds for water level, pH, dissolved oxygen, and temperature over a period of 24 hours.

Introduction
There are a wide variety of reasons why one may wish to monitor the quality of water in a stream, lake or river, such as environmental protection and public safety. The experiments conducted at the stream facility of the UNT Water Research Station are designed to test mechanistic hypotheses about species-environment interactions and effects on ecosystem functioning. In this case, environmental parameters such as those measured with the WSN both mediate species interactions as well as respond to those interactions via multiple direct and indirect pathways. Most of these experiments are using equipment that requires in situ data retrieval. Probes that transmit data wirelessly are becoming more common but such systems tend to be expensive. When operating on a research budget it is preferable to keep each piece of the project as cost effective as possible and thus WSN offer an opportunity to lower costs.

Background
The focus of this research project was to create a low cost, low power wireless sensor network to remotely monitor key indicators of the health of an aquatic ecosystem, namely pH, DO and temperature. One of the major challenges with collecting meaningful data in the field is the cost of the equipment used to monitor aquatic environments [14]. This cost is a barrier that can prevent researchers from being able to deploy sensors in a dense enough pattern to adequately monitor the environment and limits our access to such data only available through in situ retrieval [3, 4]. As such, our design needed to be low power and cost effective. The Research Experience for Teachers (RET) provided a platform for us to expand our personal understanding of electrical engineering and some of the difficulties facing researchers in the field. Putting ourselves into a project where we were forced to grow and learn in areas we were not especially comfortable with allowed us to step out of our roles as teachers and into the shoes of our students. This, in turn, will help us find relevancy in the topics we cover with our students.

Literature Review
The widespread use of WSNs is based on low cost, ease of installation and low maintenance. WSN applications in natural systems are diverse and growing in various fields. Stankovic et al. [12] write that WSNs are a new technology with great potential for improving many current applications in medicine, transportation, agriculture, industrial process control, and the military as well as creating new revolutionary systems in areas such as global-scale environmental monitoring [12].

A WSN can be considered as a potential solution to monitor water level and quality in fish ponds with high energy efficiency. Nasser et al. [9] use WSN because they have achieved widespread applicability in several environmentally monitoring applications. This group is interested in monitoring pH, temperature and the DO level for their project. The usage of a sensor based monitoring system would help the manager/owner to know about the water level as well as water quality in a real-time manner.

The quality of water is important to all living organisms, however, in aquatic environments the imbalance of certain factors can be detrimental to the plants or animals that live there and in turn impact humans. Water quality includes the physical, chemical and biological characteristics of the water.

Reliable, high quality, accurate real time water quantity and quality measurements are vital for monitoring. Consider for example the DFW area in Texas, USA. More than 90% of the water supply is surface water, and the “need” exceeds the available water supply during droughts. The use of the WSN can reduce time and effort required for monitoring those specific environments. The majority of water use in DFW is municipal/domestic (i.e. use around homes and businesses, including showers, flushing toilets and landscape irrigation). Current irrigation control systems, if present, typically have a single sensor (commonly place in gutters or in the soil) which does not apply well to a heterogeneous landscape and is an “all or nothing” approach. WSNs could have water-saving applications.

Agriculture is another field where wireless sensor monitoring can offer potential solutions to the problem of water supply. Efficient water management is a major concern in many cropping systems in semiarid and arid areas all over the world. Kim, Evans, and Iverson describe the use of WSN’s in the field of agriculture [5]. The development of a distributed in-field sensor-based site-specific irrigation system offers the potential to increase yield and quality while transmitting the data wirelessly to a base station. In certain cases, according to the article [5] the ability to apply site-specific irrigation management to match spatially and temporally variable conditions can increase application efficiencies, reduce environmental impacts, and even improve yields. A different approach is the development of a wireless agricultural environment measuring temperature, humidity, atmospheric pressure, soil moisture, and water level and light detection [8]. According to Mendez, Yanus, and Mukopadhyay [8] monitoring of environmental factors have increased in importance over the last decade. Their study investigates remote monitoring using Wi-Fi, where the wireless sensor nodes send data wirelessly to a central server. In the agriculture industry, this would eliminate the need for individuals to take measurements manually at different intervals. More important, the wireless technologies will significantly reduce the development and maintenance cost comparing with the wired solutions.

Aquaculture is another area where WSN’s would make an impact on the survival of fish. Lloret et al. [6] propose a group-based underwater wireless sensor network (UWSN) to be used to quantify and monitor the accurate amount of pollution that is deposited on the seabed. The success of their project would not only help sustainability in these facilities but also protect the environment. The use of the sensors would allow to protect the environment and to quantify the amount of wasted food.

The use of wireless sensor monitoring reduces the overall cost of the system and provides flexibility in terms of distance and location. Rasin et al. [11] describe the use of high transmission power of Zigbee based WSN for water irrigation control monitoring system. The system composed of multiple sensors is used to detect the level of water in the reservoir area and based on the signal from the sensors, the gate controlling the water flow will either open or close the gate. Zigbee is a communication standard for use in the wireless sensor network defined by the Zigbee Alliance. The standard is chosen due to its features that fulfill the requirement for a low cost, easy to use, minimal power consumption and reliable data communication between sensors.

**Designs**

**Arduino**
The Arduino microprocessor was developed with the hardware hacker, or “Maker”, in mind. To this end specific effort was put in to making the hardware as inexpensive and accessible as possible. The programs for Arduino are called “sketches” and they are based on the C programming language. There are a wide
variety of libraries that add functionality to the programs and a robust community of open-source developers who provide cookie cutter solutions that you can adapt to fit the needs of your project [1]. A number of companies also provide sample codes for the devices they create that allow for an almost plug-and-play functionality with the Arduino environment, a fact that came into play in choosing the wireless radio and the sensor probes.

![Arduino Uno](image)

**Figure 1 Arduino Uno Photo by Zac Bunn**

**XBee**

The selection of the radio transmitter for the wireless sensor network hinged on finding something inexpensive and open source that would work well with the Arduino microprocessor. The XBee radio was an ideal choice because they are cost effective, low power and come with a predeveloped shield that allows you to plug it directly into an Arduino. The XBee uses the ZigBee protocol to establish and control the wireless network. Zigbee is a high level communication protocol that complies with FCC rules for private transmission. It is designed for applications that do not have a demand for high bandwidth and a need to function in low power systems which makes it ideal for this setup, though it does preclude the ability to transmit video over the network [7]. Zigbee utilizes a mesh network layout that allows the radios to form ad hoc connections and transmit their data along that web rather than relying on one high power transmitter in the center of the network that is able to reach all the radios. The ad hoc nature of the ZigBee protocol makes it a robust solution that is well suited to field installation [13]. If a researcher wants to add another sensor cluster to the network all they should have to do is flash the Arduino embedded in the sensor cluster with the program and install the sensor cluster somewhere in range of the existing wireless sensor network. It should automatically be integrated and add to the data flow. On the other side, if a sensor cluster is lost or destroyed in the field it should not cripple the whole network or prevent any other sensor clusters from reporting their data since the network should adapt to the loss of that node and continue functioning.
Probes
Atlas scientific is one of the companies that creates sensor solutions that are inherently Arduino compatible. Atlas carries all three types of probes that we intended to integrate into our sensor cluster (pH, temperature and dissolved oxygen) as well as a wide variety of other probes in the event that expanding the sensor cluster becomes desirable. Considering one company had developed all the probes needed for this sensor cluster, and also provided sample code for the microprocessor that was selected, it was not a difficult decision concerning where to order the probes. It also helped that the cost for the Atlas probes was at or below what other companies seemed to offer. By selecting probes from the same source it was possible to greatly simplify the task of coding the program since the underlying logic of the probes was similar [2].

After completing our original design for our sensor clusters we decided to take our experiments one step further by including a sensor to monitor the water level. We are using an ultrasonic sensor that sits on top of the water tank and gives a reading as to how far the water is away from the sensor. The larger the value the lower the water level is in the tank and the opposite is true if there is a low value.

Software
We needed to develop a code for the Arduino that would communicate with each of the sensor probes. The first step was to familiarize ourselves with the programming language by programming the Arduino to do simple tasks. Tasks included getting LED lights to turn on and buzzers to sound. After our small success we focused on writing code that would allow us to gather data from each of the probes. The main
reason why we decided to use the Arduino as well as Atlas probes was because of the amount of support readily available. We were able to find sample code that worked for each of the probes. We took the code to make each of the sensors work and combined them to program that would allow us to take readings from each sensor and package them into a single line of data to be transmitted using the XBee.

After successfully completing our first sensor cluster we moved on to the next step of getting the information to transmit wirelessly. As mentioned earlier we decided on using the XBee as our radio transmitter. We used a program called XCTU to get all of the XBee’s to communicate with our coordinating XBee. This software allows you to create a network and assign each device with a specific purpose within the network. The two roles we are using are the coordinator and the router/end device. Each of the wireless sensor clusters will be assigned as router/end devices. We will have one Arduino and XBbee that will be attached to a netbook and it will be our coordinator. The function of the router/end device is to collect data and relay it either to another router/end device or to the coordinator, whichever it has a stronger signal with. The function of the coordinator is to receive all of the data packets and upload them onto the computer.

Once the sensors were ready to collect data we needed to create an efficient way to compile all of the data that we collected. We are using a program that was developed in LabView to take the string of numbers received from each cluster and separate them back into separate numbers for node number, pH, DO, and temperature. This allows us to upload the values into a database. Once the numbers are in the database we are now able to generate graphs and better interpret the data collected.
We have decided to take our experiment one step further by including a sensor to monitor the water level. We are using an ultrasonic sensor that sits on top of the water tank and gives a reading as to how far the water is away from the sensor. The larger the value, the lower the water level is in the tank and the opposite is true if there is a low value. Along with adding this monitoring sensor we have also incorporated a control system to bring water level values back within a given set of parameters. We modified our code so that we could control water level. If water level is too high a relay gives power to a pump that drains water from the tank and if the water level is too low a relay gives to power a valve that allows water to fill the tank.

System Development

The next step in our research was to develop a way to test our sensor cluster and control system. We used a ten gallon plastic tub filled with water to test our sensor cluster. The container had a water...
valve commonly found in home irrigation systems attached to one side which would allow us to add water to bring the water level up. On the other side of the plastic tub, we attached a water pump to drain water out of the reservoir if the water levels became too high. We drilled two small holes into the lid of the plastic tube for us to insert the ultrasonic water level sensor. After our initial trial we discovered that our water pump created a siphon and when we turned the power off it continued to drain. In order to fix this problem we had to attach the hose of the drain at a level that is higher than the water level. This action allowed for the water to stop flowing when the pump was turned off. Now that we have proved that this control system works on a small scale we can implement it on a larger scale tabletop stream model. This is useful because the tabletop stream model can be used in a classroom to simulate different aquatic environments.

Figure 7 Table Top Stream Model

Results
The results that we will be discussing will be from a deployment on July 10th at 12:00pm to 12:00pm on July 11th. We deployed all 5 of the sensor clusters at the Water Research Field Station at North Texas. At the water research center there are 24 stream units (riffle and pool), that in the current arrangement are linked in sets of 8 larger interconnected streams sections of 3 units each. Each stream has three tanks with riffles that allow fish to travel between the tanks. Dr. David Hoeinghaus and Luke Driver are currently running an experiment to test fish metacommunity and ecosystem dynamics in response to extreme drought when streams are subject to extreme drought; because of this we were unable to implement our water level control system. We were able to place one sensor in each of the three pools that were in the drought condition. The remaining two sensors were placed in the non-drought treatments at different distances from the three drought sensors.
Red = Drought Conditions  
Green = Flowing Conditions

*Figure 8 Water Research Tank Setup Not shown is the riffle for the upstream pool in each treatment*
Conclusion
Overall this project was a success and yielded positive results. The sensor clusters were able to produce consistent data under the controlled conditions of a lab. We set up the sensor clusters and had them run overnight. All five clusters sent viable data for the entire testing period. The resulting graphs presented data that matched expectations. There was a concerning amount of variance in some of the data but, upon further research, we discovered that this is a known error that is an artifact of the microprocessor and could be corrected easily with existing code. We deployed the WSN in the experimental stream system at Water Research field site. The data that we retrieved from the field site shows that all five sensor clusters remained connected and streaming over the entire 24 hour period. Further, the data is consistent with expected norms based on the conditions in which they were located. The range of the WSN was more than sufficient to cover the area required and could be expanded further as each device functions as both an end user and router as part of the ad hoc WSN. The control system was installed in a 30 gal. tub under lab conditions. The water level monitoring system functioned well. The control structure for flushing the
tank in the event of alarms related to pH, DO and temperature were more problematic. Unexpected interference forced a redesign of the circuit and the logic in the code underwent several revisions before the system was operational.

Future Research
There is always room for improvement and expansion, though. In this project one obvious area for expansion is the inclusion of more instrumentation. We chose the three probes we did based on the most common water qualities tested by researchers. There are a broad spectrum of other probes that would be useful as well. Adding turbidity sensors, spectrosopes, conductivity testers or similar probes would all expand the capabilities of the sensor cluster and make it applicable to a broader range of research. Continuing that line of thought it would also be a benefit to have a more modular design for the sensor clusters. The current design is hardwired and does not allow for variability. This is fine in a proof-of-concept model but for a viable field instrument it would like prove important to be able to select from a variety of probes and only include the ones that were directly applicable to your research. Developing a battery system that could power the sensors for a number of days would also be an improvement that would allow us to more fully utilize the wireless capabilities of our sensor clusters. We can currently send data over large areas and make it available through a website. Our deployment options are limited by the length of the power cord and the availability of power.

In addition to changes that could be made to the sensor clusters, there is also room for growth with the video streaming and the control system. Both function now but they are not ideal. The control system could be improved by taking the time to dial in the flow rate through the pump and the valve to make sure that the water level stays within a fixed range even when the system flush protocol is implemented. The video system is an off-the-shelf solution that we installed and tested. It would take significant work to modify the user interface on the cameras to incorporate it in the data monitoring website that already exists. It is already possible to control the cameras but in order to observe the stream visually and compare it with the data that is currently streaming it is necessary to have two separate windows open.

Acknowledgments
Funding for this work came from the National Science Foundation Grant No. NSF-1132585 to University of North Texas for Research Experiences for Teachers Site. We thank our colleagues for their efforts and support that have made this work possible. In particular, Yixin Gu who greatly aided in building the sensor cluster and wrote the code for the probes. Dr. Fu for introducing the Arduino and the Xbee protocol. Dr. David Hoeinghaus and Luke Driver for participation in their field study and assisting with the data. Dr. Miguel Acevedo and Dr. Rudi Thompson for hosting the RET program. Dawn Chegwidden for help with lesson plans, probes and supplies.

RET (Research Experiences for Teachers) Site on Sensor Networks, Electrical Engineering Department, and Institute of Applied Sciences, UNT, Denton, Texas. This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 1132585 and the IEEE Control Systems Society (CSS) Outreach Fund. 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 NSF or the IEEE.

Works Cited
Appendix A: Arduino Code for Sensor Cluster

```c
#include <SoftwareSerial.h>    //add the soft serial library
#include <LiquidCrystal.h>      //add LCD1602 support file
#define ECHOPIN 6             // Pin to receive echo pulse of ultrasonic
#define TRIGPIN 7             // Pin to send trigger pulse of ultrasonic
SoftwareSerial phport(A4,A3);   //For lack of digital ports, use ADC ports RX(A4),TX(A3) for pH circuit
SoftwareSerial doport(A2,A1);   //For lack of digital ports, use ADC ports RX(A2),TX(A1) for Do circuit
const int numRows=2;           //For LCD screen, it has 2 rows
const int numCols=16;           //For LCD screen, it has 16 columns
LiquidCrystal lcd(numCols,numRows);    //For LCD, use 5 pins to transfer data
int led1 = 8;                   //Relay1 and LED1
int led2 = 9;                   //Relay2 and LED2

String sensorstringph = "";       //A string to hold the data from the Atlas Scientific pH circuit
String sensorstringdo = "";       //A string to hold the data from the Atlas Scientific DO circuit
char phbuffer[4];              //Buffer space for converting pH from String to number
char dobuffer[4];              //Buffer space for converting DO from String to number
float sensorph=0;              //pH value
float sensordo=0;               //DO value
float v_out;                   //Temp sensor read out value, analog
float temp;                    //Temp final temp value
boolean run1=false;            //Whether either valves operated

void setup(){                    //set up the hardware
  Serial.begin(9600);             //set baud rate for the hardware serial port 9600
  phport.begin(38400);           //set baud rate for software serial port ph 38400
  doport.begin(38400);           //set baud rate for software serial port do 38400
  lcd.begin(numCols,numRows);    //Setup LCD
  pinMode(ECHOPIN, INPUT);      //Setup ultrasonic ECHOPIN as echo input
  pinMode(TRIGPIN, OUTPUT);     //Setup ultrasonic TRIGPIN as ultrasonic signal output
  pinMode(led1, OUTPUT);        //Setup LED1 as output type
  pinMode(led2, OUTPUT);        //Setup LED2 as output type
  lcd.setCursor(0,0);              //LCD set cursor at row0, column0
  lcd.print("PH00.00");            //LCD display PH00.00 at position 0,0
  lcd.setCursor(9,0);              //LCD set cursor at row9, column0
  lcd.print("DO00.00");           //LCD display DO00.00 at position 9,0
  lcd.setCursor(0,1);               //LCD set cursor at row0, column1
  lcd.print("TMP00.00");          //LCD display TMP00.00 at position 0,1
```
lcd.setCursor(9,1);       //LCD set cursor at row9, column1
lcd.print("US000");      //LCD display US000 at position 9,1
lcd.print("cm");
}

void loop(){                          // main code
  // delay(1000);                     // delay 1 second
digitalWrite(A0,LOW);               // open ADC port
delay(10);                         // wait 10ms for stability
v_out = analogRead(0);               // read temp sensor from ADC port A0
v_out*=.0048;                      // standard calculation for temp
v_out*=1000;
temp=0.0512 * v_out -20.5128;       // Temperature reading and convert to degree centigrade
lcd.setCursor(3,1);                   //LCD set cursor at row3, column1
lcd.print(temp);                     //LCD display value of temp

re_ph:                            //Jump to here, tag: re_ph
sensorstringph="";                  //Clear it for next use
sensorph=0;                       //Clear it for next use
phport.print("\r\n");                  //Send read ph command
phport.listen();                     //Begin to receive
delay(500);                        //Wait for data ready
while (phport.available()) {            //If data received
  sensorstringph +=(char)phport.read();   //Accumulate string
}

if (sensorstringph.toFloat() > 1){           // if ph doesn’t equal to 0
  sensorph=sensorstringph.toFloat();      //Convert and send it to sensorph
}  
else
  goto re_ph;                           //If receive fails, jump back to read pH again

lcd.setCursor(2,0);                     // LCD set cursor at row2, column0
if (sensorph >1 && sensorph<10){       //If ph value greater than 1 and less than 10
  lcd.print("0");                       //Add 0 before display pH value, like 09.99 not 9.99
  lcd.print(sensorph);                  // Display pH value
}  
else if (sensorph > 10){                  //If pH greater than 10
  lcd.print(sensorph);                  //Directly display
}

re_do:             //Jump to here, tag: re_do
sensorstringdo="";            //Clear it for next use
sensordo=0;           //Clear it for next use
doport.print("r\r"); //Send read ph command
doport.listen(); //Begin to receive
delay(500); //Wait for data ready
while (doport.available()) { //If data received
    sensorstringdo += (char) doport.read(); //Accumulate string
}
if (sensorstringdo.toFloat() > 1) { // if do doesn’t equal to 0
    sensordo = sensorstringdo.toFloat(); //Convert and send it to sensordo
} else
    goto re_do; //If receive fails, jump back to read do again

crc.setCursor(11, 0); // LCD set cursor at row11, column0
if (sensordo > 1 && sensordo < 10) { //If do value greater than 1 and less than 10
   crc.print("0"); //Add 0 before display do value, like 09.99 not 9.99
    crc.print(sensordo); // Display do value
} else if (sensordo > 10) { //If do greater than 10
    crc.print(sensordo); //Directly display
}
digitalWrite(TRIGPIN, LOW); //Clear pin
delayMicroseconds(2);
digitalWrite(TRIGPIN, HIGH); // Send command to emit ultrasonic wave
delayMicroseconds(10);
digitalWrite(TRIGPIN, LOW); // Send pin low again, terminate command
int distance = pulseIn(ECHOPIN, HIGH); // Read in times pulse
distance = distance / 58; // Calculate distance from time of pulse
delay(50); // Wait 50mS before next ranging

crc.setCursor(11, 1); // LCD set cursor at row11, column1
if (distance < 0) { // If response distance is negative, display 000
    crc.print("000");
} else if (distance < 10) { //If response distance less than 10, display like 007
    crc.print("00");
    crc.print(distance);
} else if (distance < 100) { //If response distance less than 100, display like 077
    crc.print("0");
    crc.print(distance);
} else {
if (distance > 20) { // If distance greater than 20cm
    digitalWrite(led1,LOW); //Turns on relay2(LED2), refill water
    digitalWrite(led2,HIGH);
    run1 = true;
}

else if (distance < 10) { //If distance less than 10cm
    digitalWrite(led1,HIGH); //Turns on relay1(LED1), drain water
    digitalWrite(led2,LOW);
    run1 = true; //Tag, means relay has been operated
}

else if (distance == 15 && run1 == true){ // If water level equals to 15 and the relay just operated,
    digitalWrite(led1,LOW); //Turn off both relays
    digitalWrite(led2,LOW);
    run1=false; //Release tag for next use
}
else if(run1==false){ //If none relays has been operated, water level adjustment finished
    if (sensordo < 3 || sensorph < 6 || sensorph >10 ){ // Either do<3 or ph<6 or ph>10 will flush water
        digitalWrite(led1, HIGH); //Turn on both relays
        digitalWrite(led2, HIGH);
        for (int i=6; i>=0 ;i--){ //Loop 6 times
            lcd.setCursor(8,0); //LCD at position 8,0
            lcd.print(i); //Display number 6,5,4,3,2,1
            delay(3000); //For each number, delay 3 seconds, so each loop cost 18 seconds
        }
    }
    else { //If no problem with do and ph
        digitalWrite(led2, LOW); //Flushing finish
        digitalWrite(led1, LOW);
    }
}
else
Appendix B: Circuit Schematic