**Remote Weather Station**

**Andrew Schaaf, Adam Baldwin, Hieu Luong**

**Draft #1**

**September 9, 2016**

**CNT 4104 Software Project in Computer Networks**

**Instructor: Dr. Janusz Zalewski**

**Department of Software Engineering**

**Florida Gulf Coast University**

**Ft. Myers, FL 33965**

Copyright © 2016 by Florida Gulf Coast University

**1. Introduction**

Having access to real-time weather data is imperative for both businesses and individuals and is something that many applications and programs depend on. Users demand that data be accurate, up-to-date and dynamic in order to suit their personal needs.

We will be extending the Remote XBee Weather Station which is an extension of three previous projects. The project was started in 2009 by Brad Konert, was extended by Christopher McCoy in 2011 to have internet accessibility [1] and was last worked on by Sergio Pais in Spring 2015 who showed that data could be stored on a simple server and sent to a client. [2] We will be creating a cloud-enabled weather station and an Android application to serve as a client. We will be extending the previous work both at the hardware and software level. We will be using a new sensor to get extended dynamic weather data to our client. The sensor will transmit data to the Arduino board via the XBee transceiver. The server will connect the to the Arduino board via USB and store data on the cloud using Amazon RDS. The client Android application will receive weather data from the server via Web API calls which will request both real-time data from the server and long-term data from the Amazon database.

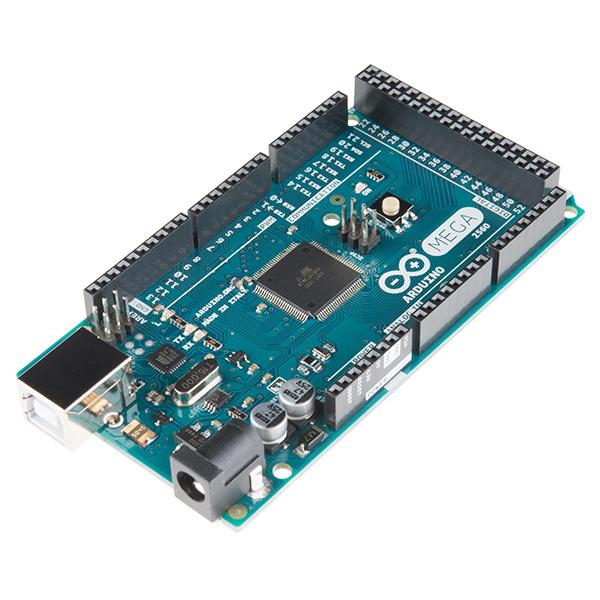


Figure 1.1: Arduino Mega 2560



Figure 1.2: XBee on shield

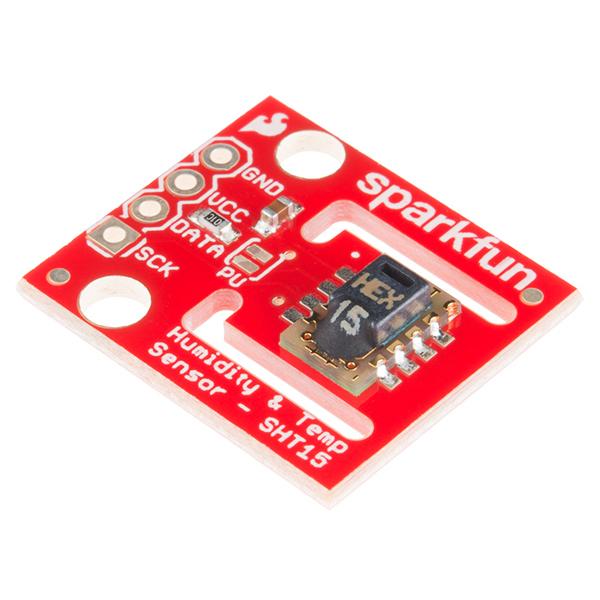


Figure 1.3: SHT15 Temperature and Humidity Sensor

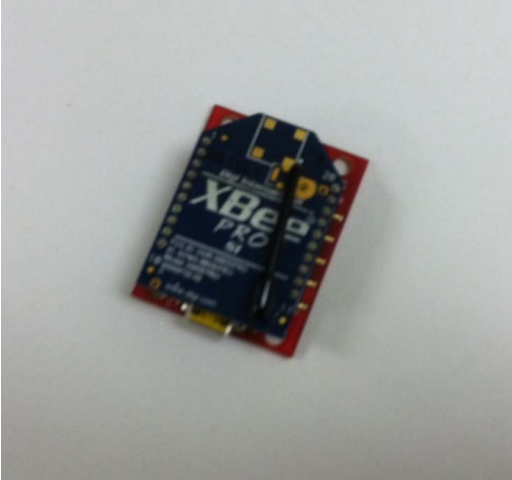


Figure 1.4: XBee Pro on development board

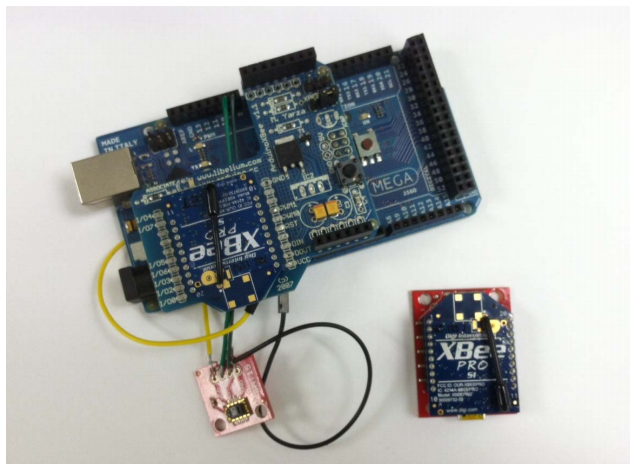


Figure 1.4: Assembled components

**2. Software Requirements Specification**

**2.1 Project Objectives**

This project is a continuation of three previous semesters of work. The previous work served as a proof of concept that weather data could be read using a sensor, stored in a database, and be sent to a client. Currently, the sensor reads temperature and humidity data, sends it a simple server set up, and sends it to an ASP.NET web form which serves as the client. This semester we will be reconfiguring the sensor, the server and database system, and the client application. The software will store data on an Amazon RDS database and receive requests from a client via API calls as shown below in Figure 2.1.1

Untitled Diagram.png

Figure 2.1.1: Context diagram

**2.2 Physical Diagram**

The weather will be polled by the weather sensor and transmitted to the Arduino board via the XBee transceiver. The Arduino board will be connected to the server via USB and the server will store the most recent hour of information locally and at the top of each hour, will send an average of the previous hour to the Amazon RDS database which will be stored for possible long term weather forecasting. The server will send both local real-time data and long term data from the Amazon database to the Android client applications. The physical connections are shown below in Figure 2.2.1.

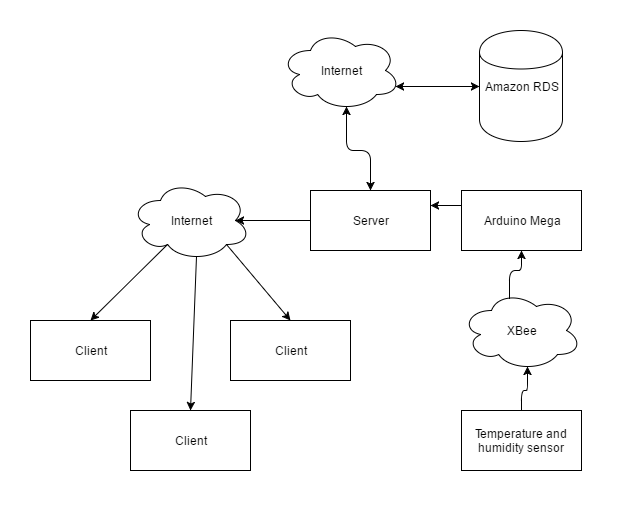


Figure 2.2.1 Physical diagram

**2.3 Requirements Specification**

**2.3.1 Arduino Requirements**

A1) The software shall poll the temperature and humidity sensor every 15 seconds.

A2) The software shall restart the polling process when hardware is rebooted.

**2.3.2 Server Requirements**

S1) The software shall store the most recent hour of data locally.

S2) The software shall store an average of the previous hour data to the cloud database every 60 minutes.

S3) The software shall receive API requests from client and send data in JSON format.

S4) The software shall restart when the server is restarted.

**2.3.3 Client Requirements**

C1) The software shall display most recent weather data by making API calls to server.

C2) The software shall display the averages from each hour of the previous 24 hours

C3) The software shall make a request to the server to update weather data every 15 seconds.

C4) The software shall update the weather data when refresh button is pressed.

**6. References**

[1] C. McCoy, Remote XBee Weather Station, FGCU CNT 4104 Software Project In Computer Networks, December 2011.

[2] Sergio Pais, Remote XBee Weather Station Maintenance Report, FGCU CEN 4935 Senior Software Engineering Project, April 2015.