Exploring the Internet of Things:

The Greenhouse Manager Module

Group Members: Derrick Alders, Daniel Longo, Michael Wholey

Due: May 10th, 2019

**Abstract**

**Given the task of developing a multi-sensor, app connected project that would display our skills in manipulating and exploring the idea of the Internet of Things (IoT henceforth), our group did not waste much time in developing a concept. All three of us have had, or are still actively, grown plants with a mixed data set of success and failure. When Daniel mentioned that he had specially ordered some sensors, the idea came to use to create a “smart greenhouse” concept that can be put to use in household, educational, or industrial use. We called it the “Greenhouse Module Manager” (GMM henceforth) and set to work.**

**The GMM was designed to be variable size, allow for the addition and subtraction of sensors as needed, and can be accessed wirelessly from an Android smart device such as a cell phone or tablet. The information updates in real time with an easy to read GUI and at-a-glance indicator lights for the casual or fast paced user. Should an alert be necessary in case something goes wrong in the GMM, push notifications can be enabled for round-the-clock observations. Using the combination of an Arduino, Raspberry Pi, NodeRed, and MIT App Inventor, we succeeded in our endeavor.**

**Introduction**

**Our project concept started with personal desire from the group members to develop a more autonomous, smarter way to grow household plants and spices. Every member has had experience with losing a plant from some kind of neglect whether it be water, light, temperature, or a slew of other issues that can happen while attempting to grow a plant from a seed state. We developed the GMM to address these issues and increase chances of success for botanists both fledgling and experienced.**

**Motivation**

**After speaking about our project idea with our instructor present, he notified us that not only is there personal and industrial interest in a project idea like ours, but that several of the professors on campus keep plants in the university greenhouse and had mentioned interest in a concept like ours. With the submission for a campus-wide coding competition coming up, we decided to pursue this and submit it for both that competition and for our final project.**

**Significance**

**Growing plants from seeds can be challenging. This project aims to make that process simpler.**

**Goals and Objectives**

**The goal of this project is to hybridize everything we’ve learned thus far in our 290: Internet of Things class.**

**Features**

**Wood or plastic exterior, determined by the location that the module would be kept, with all sensors being contained within the confines of the apparatus. Sensors and software that were used are:**

* **Soil Moisture Sensor**
* **Thermometer**
* **Barometer/Humidity Sensor**
* **UV Detection**
* **Light Intensity**
* **Arduino IDE (for Arduino Uno use)**
* **Python**
* **NodeRed**
* **MIT App Inventor**

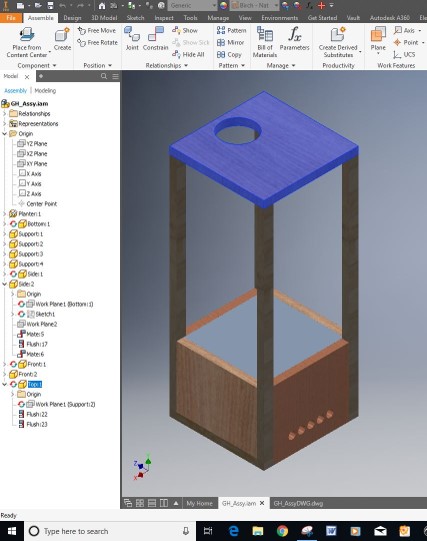
**As every sensor was smaller than a playing card, the size of the module was not limited by the technology. Furthermore, planned additions to the sensor array would increase connectivity. We also used a set of different microcontrollers to arbitrate the flow of information in the project. The sensors were connected directly to an Arduino Uno due to its resilience to electrostatic discharge, feedback current, and physically robust design. The other microcontroller chosen was the Raspberry Pi 3. This model of the Pi has a decent processor speed and its ability to use the Raspbian operating system allows it to manage more complex data scenarios. This is why we chose the Arduino to act as the body of our project and the Pi as the brain.**

**Proposed Methodology**

**The workload was split up based upon our group strengths. Daniel had access to hardware and the software required to make professional drafts ahead of time, so he took on the largest task of constructing the module. Michael is the strongest programmer in the group, and had successfully gotten our groups Pi and Arduino to communicate effectively previously. Finally, Derrick created the app that requested the information, manipulated it and made it human readable and user friendly while also being able to send commands back to the apparatus to control certain environmental aspects.**

**This project progressed smoothly since its inception. We were given two weeks to work on the project but day-to-day student and professional requirements set us back nearly a week. However, that didn’t stop us from finishing everything on time. The only real issue we encountered in the creation of the GMM was during the night of the presentation, when we had a hiccup in our lines of communication.**

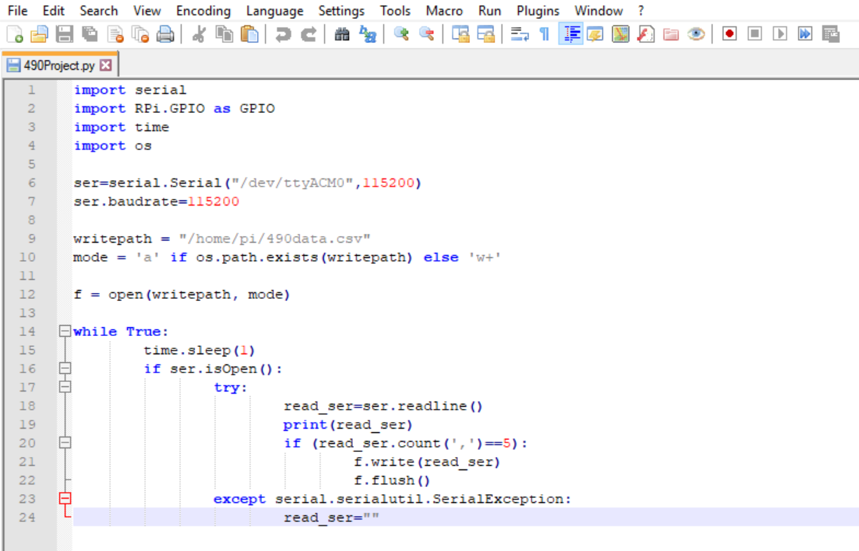
**Implementation**



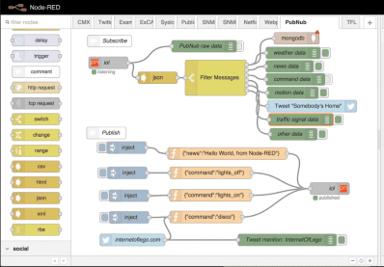
**To begin, I’d like to direct you to some video media of the working project:**

* <https://www.youtube.com/watch?v=QndK-gy7p18> **Part 1**
* <https://www.youtube.com/watch?v=BIkk1ZyzQYU> **Part 2**
* <https://www.youtube.com/watch?v=C3AOwtNZFKU> **Part 3**

**The above picture shows a draft done by Daniel while he was preparing a physical apparatus for our in-person demonstration. You can see the hole on the top that will be for an exterior mounted, DC fan that will pull air from within the module when it becomes too hot or humid for the plant to thrive. Ventilation is ensured by the holes in the bottom that will allow for new air to be pulled in.**

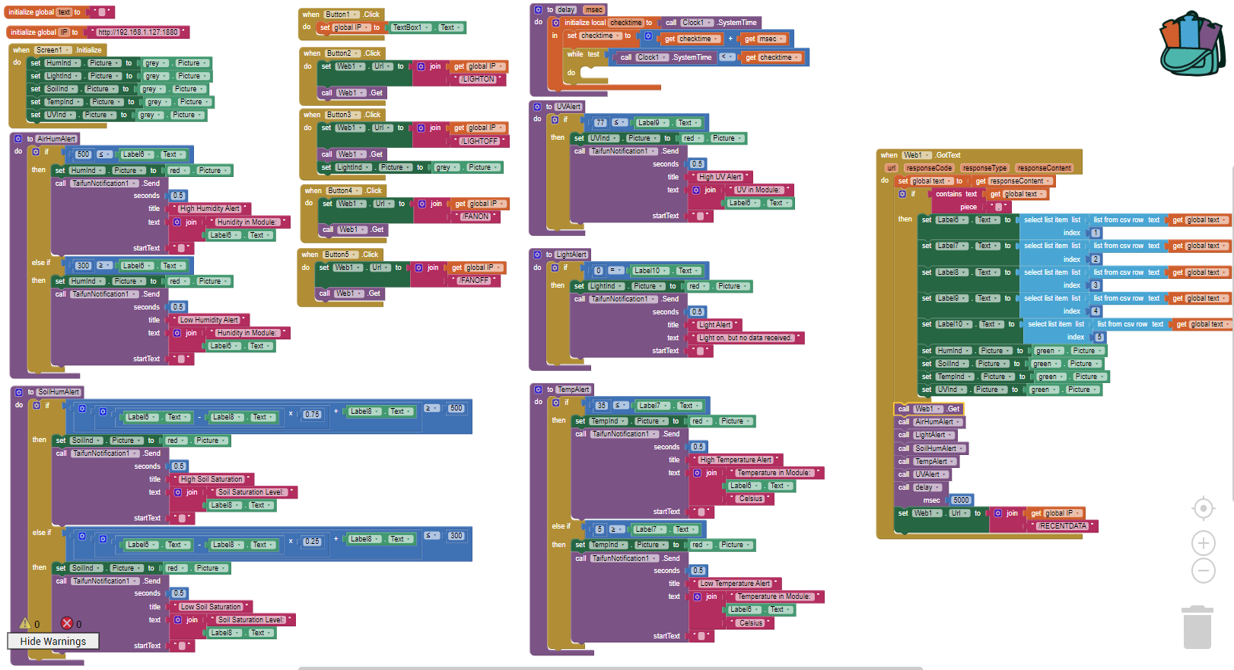


**This is a sample of the python code ran by the Raspberry Pi to read and send values to the NodeRed server. The Pi operated as the brain of the entire project and communicated with the Arduino which pulled in the individual sensor readings. This allowed for a structured workflow between the signal producers and signal consumers in the network.**

****

**This is a sample of the NodeRed server that communicated wirelessly with our Raspberry Pi. Intrinsically, NodeRed has a number of ways to smoothly receive and transmit data. JSON and CSV are the two easiest and most commonly used ways to encode and decrypt this information and we decided for the lack of time and necessity of simplicity to use this format too, and you can see above that the Flow takes in a single JSON formatted string and parses the information into the CSV sections needed to be processed. It is these CSV values that the app accesses.**

**The following code was written using the software called MIT App Inventor and, presently, only works with android devices. It collects the CSV formatted data from the hosted NodeRed server via the Pi and then displays that information appropriately in the app’s GUI. It updates its information every second and stores only the last 10 updates locally. However, older data can be accessed from the server in the form of a graph.**

****

**The format of the app GUI was designed to be both related to the overall project idea and easy to read for users of any skill level. The first step is to connect to the IP address of the server that the Pi is hosting. Every second or so, the app will update the information displayed below the appropriate title in the middle of the screen. If the received values fall within a user defined specification, the indicator light below the current value will be green. If the sensor is off, the light will be gray. However, should the value deviate from those preset values, the indicator light will turn red, and a custom push notification will be sent to the phone depending upon which sensor deviated. Some physical aspects of the apparatus can be controlled remotely by pressing “LIGHT ON/OFF” and “FAN ON/OFF”, and these options will primarily be used to regulate temperature and humidity within the enclosure. A sample, blank GUI is included below.**

**A screenshot of a cell phone

Description automatically generated**

**Applications**

**In addition to the practical applications mentioned before, ew have possible future plans that we would like to add at some point. These additional features include graphical interfaces on the app instead of just on the online UI. This would give a more readily accessible understanding of the state of the module. Also, a slider may be added to take the module out of what we call ‘automatic’ mode into ‘manual’ mode. In manual mode, the user wouldn’t need to preprogram the alert thresholds, but could update them dynamically from time to time. Finally, the ability to manage multiple modules could be added.**

**To build on this idea, the WiFi module used has the ability, thru RFID, to detect proximity of the device accessing the information. MIT App Inventor allows for the reuse of code and multiple screens could be used for multiple apps. Similarly, the hosting process of the Pi via NodeRed isn’t not particularly taxing and multiple servers could be hosted simultaneously. This would allow those previously mentioned household, educational, and industrial applications to include a variety of plants.**

**Conclusion**

**In conclusion, we developed an interactive experience for Andriod users that want to manage a nearly limitless number of possible botanical projects. With a robust, and innovative design, the physical apparatus can be as imposing or subdued as the user wishes and can be adapted to the target flora’s requirements. With the possible number of sensor and control additions that can be made, there’s room for growth with our design. A relatively seamless, user-friendly device and app helps even the most inexperienced botanist to grow plants successfully. Through the marriage of the Arduino, its sensors, the Pi, and an Andriod device, we made it possible to monitor and manipulate a controlled growth environment from afar.**