End Of Internship Report

Introduction

I am a recent graduate from the University of Sheffield and I have just been awarded an MEng in Electronic and Communications Engineering. Besides my degree specialisation, I am also interested in software development and am self-teaching those skills. I applied for this internship to gain the kind of practical problem solving experience that will be relevant to my career. My fourth year group project was to design an ultra-low power sensor node, and my task was to implement a message passing algorithm for the nodes, so this internship was an opportunity to do further work on something in which I was already interested.

In the first two weeks I had to decide a project direction, and I eventually settled on carrying out an investigation into how IoT may be applied to optimise waste collection in cities.

Project Background

I started by researching smart cities and found that it was surprisingly difficult to extract the necessary information required to have an understanding of the problems faced by smart cities and the solutions that are being worked on with internet searches alone. I spoke to Chris Dymond of Unfolding, who is involved in the Sheffield IoT scene, who spoke in high-level, non-technical challenges and requirements of smart city technology. He spoke about the difficulty of negotiating contracts because the decision makers are simply not technically literate; how there was a “big deficit in [IoT] innovation”; the issue of marketing IoT solutions; and the presence of security threats. Whilst not challenges that I was equipped to take on, our discussion did lead me down certain avenues of thought that led me to set myself the tasks that I did.

I had posed three questions to Steve Jubb too:

1. What are the main medium to long ambitions of the engineers working on smart cities?
2. Can LoRa technology meet the requirements of all the big projects that are going on, or are there some which require some other protocol/technology?
3. What have been the main technical challenges in implementing these sorts of networks?

The size of the scope of my questions was about inversely proportional to my prior knowledge, which is to say I had very little. Quickly, I became aware that although I had a vague conceptual understanding of IoT and LoRa etc, most of it was completely new to me, and whilst my degree had given me the skills to work on this project, I realised that I needed a lot of knowledge, which I was lacking, to even begin. I organised a call with Steve and invited fellow interested interns and Steve gave us some answers.

I decided to work in the direction of smart waste collection for a few reasons:

* Current waste collection methods in cities may be efficient enough for cities to function, but they inefficient enough to be ripe for optimisation
* There are companies trying to do so already, which suggests that it is viable
* Security concerns, whilst present, did not seem to be a threat to the validity of my work

My Project: Investigating Application of IoT for Smart Waste Collection

I wanted to develop an IoT system to serve as a proof of concept which could applied to smart waste collection. The idea was that an urban environment may be measured to detect real-time waste levels and predict the optimum times for waste collection. Both smart buildings and outdoors spaces may be considered as “environments”, and a smart building may use data from an outdoor space for its own predictive models. Realistically, prediction would most likely rely on machine learning, but what I was capable of doing at home was capturing and storing data.

I decided that I wanted to set up a local network where a node could sense bin level or human movement, transmit that data and for it to be stored somewhere. I was to establish a network that dealt with local communications and communications with TTN.

Issues to do with regulations prevented hardware from being sent out so I purchased the following:

* **LOLIN D1 Mini (ESP8266):** a Wi-Fi enabled micro-controller, programmable with Arduino IDE
* **Heltec LoRa 32 (ESP32):** a Wi-Fi and LoRa enabled micro-controller, programmable with Arduino IDE
* **An assortment of sensors**
* **Raspberry Pi 3+:** To be used as an IoT server

Firstly, I had to set up the micro-controllers to be used with Arduino IDE. This was straightforward, simple and there are many sources online with the same simple instructions. The same can be said of connecting the sensors to the micro-controllers. But with all of these devices one cannot just plug it in and expect it work, there is *some* setup to do. I was able to read sensor data from the serial port, which let me know the hardware was working as expected.

Second, I wanted to be able to transmit that data and store it. Only at this point did I realise that I would need some software that I had never used, or heard of. I spent a long time watching and reading tutorials so that I would be able to plan properly and not run into totally unforeseen obstacles and the prevailing advice was to get a Raspberry Pi and have the micro-controllers communicate with it. I used an OS called Home Assistant on the Raspberry Pi. On Linux machines, Home Assistant has add-ons which eliminate the need for a lot of programming. I was able to use the ESPHome add-on along (again, following tutorials) to establish a Wi-Fi link between the Pi and D1 Mini and all of the code was hidden from the user, who simply selects the sensor connected and specifies ports and interval between transmissions. InfluxDB and Grafana were also add-ons. The Pi itself has no GUI when running Home Assistant, only the command line. The user is expected to run make the changes described in a web browser at the Pi’s IP address on another machine. Within two days this IP address stopped displaying the webpage. Home Assistant may be very user friendly when it is working, but the opposite when it needs to be fixed. Much time was spent trying to give the Pi a static IP address but this turned out not to be the issue. The time saved by using Home Assistant was more than repaid in trying to fix it. I never worked out the issue, but the only microcontrollers I had were not being put to good use so I abandoned the program.

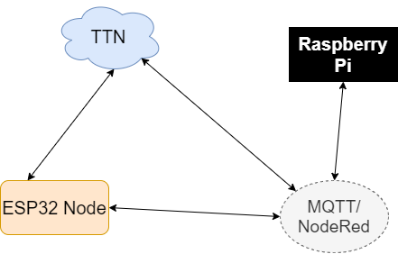
Third was a connection to TTN. I worked through this partly with Thomas Edwards who was using a different ESP32 board. Soon after purchasing the Heltec board, I saw a video by Andreas Spiess, a YouTuber whose videos I know that several of us in this cohort have relied on, which showed how the antennas for the Heltec boards perform very poorly at 868MHz. This was corroborated by my attempts to connect to TTN. The example sketch indicated that data was being transmitted but TTN did not receive anything. I travelled to a cluster of gateways, thinking that a distance of about 600m was surely close enough. I had to be within about 10m of the building for packets to be received, even after first connection. Tom and I were both using an example sketch using the LMIC library which sent a default message at a fixed interval and we talked about our experiences and traded tips and eventually I worked out how to edit the message and change the interval between transmissions. Initially, I had problems getting even the example to compile because of out of date header files in the library. Ultimately, after being sent a gateway by Steve, I was able to transmit any data to TTN at any time. For example, it could be set such that an alert was sent every time motion was sensed or the number of motion events in a certain time frame could be counted and sent to TTN.

Lastly, I revisited the Wi-Fi network. Walter Bassage had recommended using Docker to run Influx and Grafana on the Raspberry Pi so I did just that. Running Docker on Raspbian was a much more sensible way of doing what Home Assistant had tried to do. Docker made installing the programs very easy. Mike Faulks spoke to myself and another intern about MQTT and NodeRed, and with that I was able to set up a flow.

Advice to an Intern at the Start of this Project

To somebody in my position ten weeks ago I would say that after deciding on a direction, that before any action is taken, they should make themselves aware of every step required to reach the goal and make a thorough list of all hardware and software required, if they are to progress quickly. It is only possible to learn by doing past a certain level of familiarity. If a device sends data over Wi-Fi, where is it being sent? Do you really understand the Wi-Fi protocol in any detail? What hardware/software do you need to make it send the data you want it to? Questions like these should be answered as far as possible before starting a new task and it is best to see how it is done to answer these questions and minimise the amount of troubleshooting. I found that when I was troubleshooting, I often had to learn something rudimentary about the hardware or software to even understand the problem. This was especially true for the Raspberry Pi. I would find the problem I was having online somewhere, but the solution would use technical language totally unfamiliar to me which meant that I had to the look up those terms and learn quite a lot simply as a basis for being able to make the Pi recognise my node or return some information I needed.

Diagram of Final Setup



Useful Links

<https://github.com/matthijskooijman/arduino-lmic> : The LMIC library (fairly well documented)

<https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino-ide/> : NodeRed/MQTT Tutorial