

May 2018

# How IoT Devices can be used in Rented Accommodation to Collect Meaningful Data

Cardiff University School of Computer Science and Informatics



Joshua Denton

## 1.1 Acknowledgments

Firstly, I would like to thank Professor Omer Rana for his guidance in his role of supervisor.

Additionally, I would like to thank United Welsh for allowing me to perform this project researching for them and giving me access to a property to test within, with special thanks to Louise Shute at United Welsh for giving me her insight on the needs of both tenants and United Welsh.

Finally, I would like to thank Cardiff University School of Computer Science for the loan of all equipment used, with special thanks to Laurence Semmens for his assistance with this.

## 1.2 Abstract

This project looks at implementing a wireless sensor network within rented accommodation to investigate whether it can benefit both the housing association and the tenants of the building. It gives an overview of wireless sensor networks and their associated technologies before going into detail about the Libelium Wasp mote and how this can be programmed and used with Zigbee to act as sensor nodes connecting to a Libelium Meshlium gateway. This report found that a Libelium wireless sensor network could be set up in rented accommodation to yield useful results and it sets the path for further advances in the area.

# Table of Contents

1.1 Acknowledgments .....	1
1.2 Abstract.....	2
1.3 Table of Figures .....	4
2 Introduction .....	5
2.1 Project Aims .....	5
2.2 Project Beneficiaries .....	5
2.3 Project Scope.....	5
2.4 Key Objectives of the Project.....	6
3 Background.....	7
3.1 Internet of Things .....	7
3.2 Wireless Sensor Networks.....	9
3.3 Communication Methods.....	11
3.4 Current sensors .....	<b>Error! Bookmark not defined.</b>
4 Approach .....	14
4.1 Libelium Waspote .....	14
4.2 Waspote Software Development Kit .....	16
4.3 Meshlium .....	17
4.4 United Welsh Test Bed.....	18
5 Implementation.....	19
5.1 Setting up Waspotes .....	19
5.2 Setting up of Meshlium.....	22
5.3 Programming Waspote For Wireless Communication.....	23
5.4 Setting up MySQL Server.....	26
5.4.1 Connecting Remote Database to Meshlium.....	26
6 Results .....	28
7 Future Work .....	32
8 Conclusions .....	33
9 Reflection on Learning .....	34
Bibliography.....	37
Appendix.....	39
Note from United Welsh.....	39

## 1.3 Table of Figures

Figure 1 – IoT Visualisation .....	7
Figure 2 – WSN Topology Diagram .....	10
Figure 3 – Range Vs. Data Rate for IoT Wireless Technologies .....	11
Figure 4 – Meshlium Network Diagram.....	17
Figure 5 – New Waspote Sketch .....	20
Figure 6 – Sensor Output Snippet.....	20
Figure 7 – Deep Sleep Snippet.....	21
Figure 8 – Xbee ASCII Frame Structure .....	24
Figure 9 – Adding a value to a frame snippet .....	24
Figure 10 – Sending a frame snippet.....	24
Figure 11 – Queries to setup database .....	26
Figure 12 – Recorded Data in Workbench .....	27
Figure 13 – Results Day 1 .....	29
Figure 14 – Results Day 2.....	29
Figure 15 – Results Day 3.....	30

## 2 Introduction

### 2.1 Project Aims

The aim of this project is to obtain a proof of concept that a wireless sensor network can be implemented into a house hold environment to yield useful data that can be used by both a housing association and the tenants at the properties with sensors installed. It is not an attempt at a full-scale deployment in the home instead an initial test to investigate if sensors currently on the market can be deployed in a home environment to work successfully and determine what readings could be taken in future.

### 2.2 Project Beneficiaries

United Welsh Housing Association requested that a project of this nature be done to assist them in the decision-making process of whether to install IoT sensors within their properties. Please see a note from a representative in the appendix of this report that explains United Welsh and their requirement.

### 2.3 Project Scope

The specialist technology used in this project will be that found in the Libelium Wasp mote Evaluator Kit along with the Libelium Meshlium ZigBee-AP. These will be used alongside open-source software associated with or recommended by Libelium. Any recommendations or conclusions drawn about specific results of testing or any code related ideas cannot be guaranteed to yield the same results using other technologies available. However, any conclusions drawn about the use of wireless sensor networks generally can be applied in a general manor to other technologies available.

## 2.4 Key Objectives of the Project

There are several key objectives to this project, firstly, become familiar with wireless sensor networks and the communication technologies they use. Secondly, learn how Waspnotes work and how the programs run on them are structured as well as gaining an understanding of C++. Thirdly, create a wireless sensor network that sends data to an external database that allows data to be accessed and used in other programs. Next, install the wireless sensor network into a rented accommodation setting and record data. Finally, draw conclusions based on the network implementation on whether it shows that a wireless sensor network can be implemented in rented accommodation to give meaningful results.

## 3 Background

### 3.1 Internet of Things

In its most broad sense, the term internet of things just refers to any device connected to the internet, however in recent years it has come to refer more to devices connected to the internet that are communicating with one and other. The possibilities for these devices only end with the human imagination whether you are sensing light levels to determine when street lamps should turn out, monitoring radiation levels at a nuclear powerplant, even to keeping track of when your toaster is used at home. The Internet of Things is a very large area of industry and has naturally formed multiple disciplines including “industrial IoT” which covers IoT applications associated with the manufacturing process or monitoring of utilities. You also have IoT devices that are categorized by their device type such as “wearables” and “smart appliances” for example your smart watches and your internet connected toaster. As well as these you have IoT devices categorized by integrated location-based implantations such as “smart homes” and “smart cities” [1]. These being sensor networks and smart devices in homes or across cities.

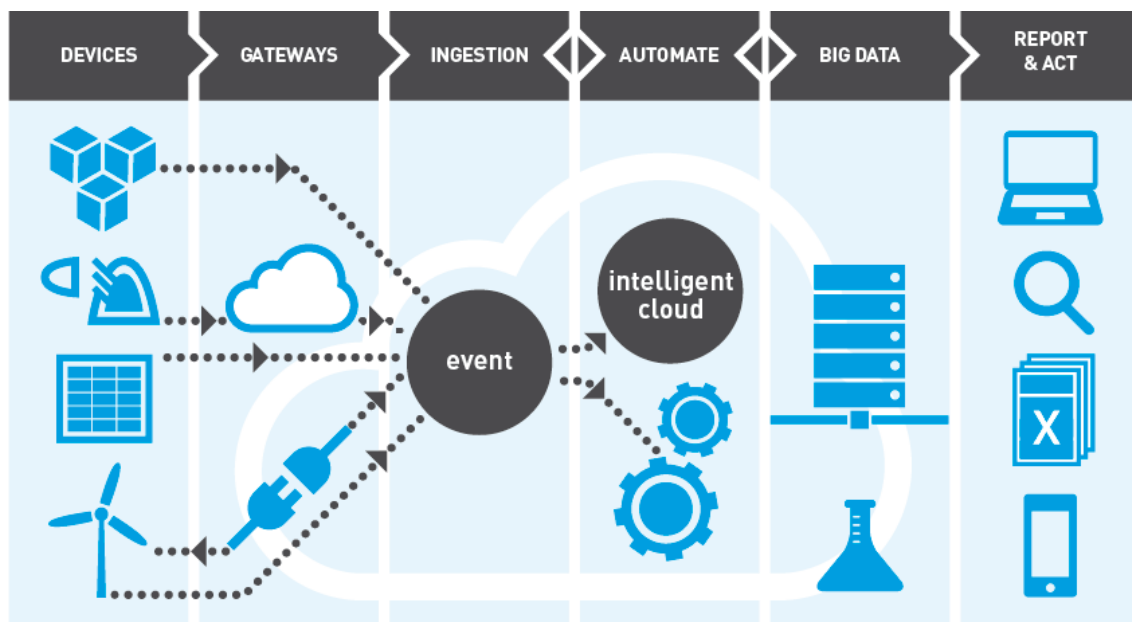


Figure 1 – IoT Visualisation

There is one school of thought that for a device to be truly IoT it is required to have 4 features. It should firstly have the ability to sense, this sensing could be could be anything some form of data capture from the outside world. Secondly, the device should have a method of communication, this could be via LAN, WAN or PAN using any of whole range



of communication technologies as long as it has the means to transmit the information from device level to cloud level. Thirdly, there should be some form of cloud-based capture and consolidation of data. The gathered data should be aggregated with other cloud-based data whether it be other devices, internet sources or defined parameters so that it can provide useful information to the user. Finally, the delivery of information should be such that useful information is given to the user such that it is seen in as simple and transparent way as possible to allow the greatest benefit to the user. [2] In figure 1 above, you can see these 4 aspects in motion.

## 3.2 Wireless Sensor Networks

Wireless sensor networks are one of the key technologies of IoT, they are networks which consist of a multiple, usually large numbers of, sensor nodes. Each node will be fitted with a sensor to detect a real-world event such as temperature, CO<sub>2</sub>, presence, etc. They may also cooperatively control the environment they are in based on the readings they sense for example the node may be measuring temperature and be programmed to adjust the heating of the room based on those readings. Within the network there will be a gateway which connects the network of nodes to the outside world via the internet so that the information that has been collected can be seen and used effectively. [3]

The first notable largescale sensor network was developed in the 1950s by the United States military. It was a sound surveillance system designed to track and detect Soviet submarines. It began as a 1,000ft array of 40 hydrophones, however after successful testing the system was extended to cover both the east and west coast of the United States, all these hydrophones were sending data back to the coast line to be monitored by hand. This system is still in use today but with a more peaceful function of monitoring wildlife and tectonics. [4] In a continuation to the investments of the 1960s and 1970s to develop hardware for what we today call the internet, the United States Advanced Research Projects Agency (DARPA) started the Distributed Sensor Network (DSN) program in 1980. This was set up to formally investigate how to design and implement distributed and wireless sensor networks. From this research both military and civilian, Wireless Sensor Networks began to appear in applications such as air quality monitoring, forest fire detection and so on. Then as students made their way from universities where these sensors were being employed for research they took them in to the technology giants of the day and promoted the use of Wireless Sensor Networks in heavy industry such as power distribution and factory automation. This only increased the demand on research for this area which has brought wireless sensor technology to where it is today. [5]

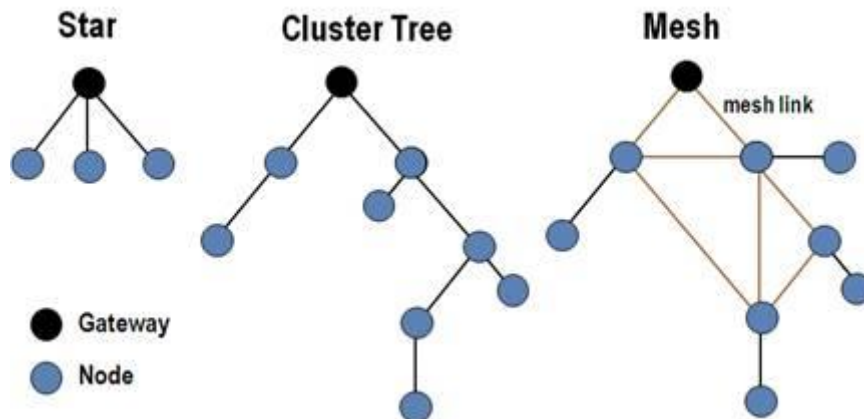


Figure 2 – WSN Topology Diagram

<http://www.ni.com/white-paper/7142/en/>

There are three standard network topologies for a wireless sensor network as shown above. Firstly, you have a star network where each node connects directly to the gateway. This is the most fault tolerant of the three for if one node fails the network remains fully operational. It is the simplest, but as all the packets between devices must come through the gateway these packets may become bottlenecked at the gateway. You then have the cluster/tree topology, this is based on the idea that you have a central gateway with several routers (parent nodes) and end devices (children). The children are only able to communicate to the parent node therefore if the parent fails then so do the children. Compared to a star network you can have a greater geographical range from the gateway however the complexity of the network is much higher. Finally, you have a mesh topology, here nodes can connect to multiple nodes in the system and the data is passed through the most reliable path available. The network is seen as self-healing as during transmission, if the chosen path fails, the child node will find an alternative route to the gateway. The range of a WSN using a mesh topology can be increased by adding more nodes to the network. This topology uses a much more complex routing protocol compared to a star network. A mesh network is very reliable however it increases latency with the size of the network and the number of 'hops' between nodes to get to the gateway. [6] [7]

### 3.3 Communication Methods

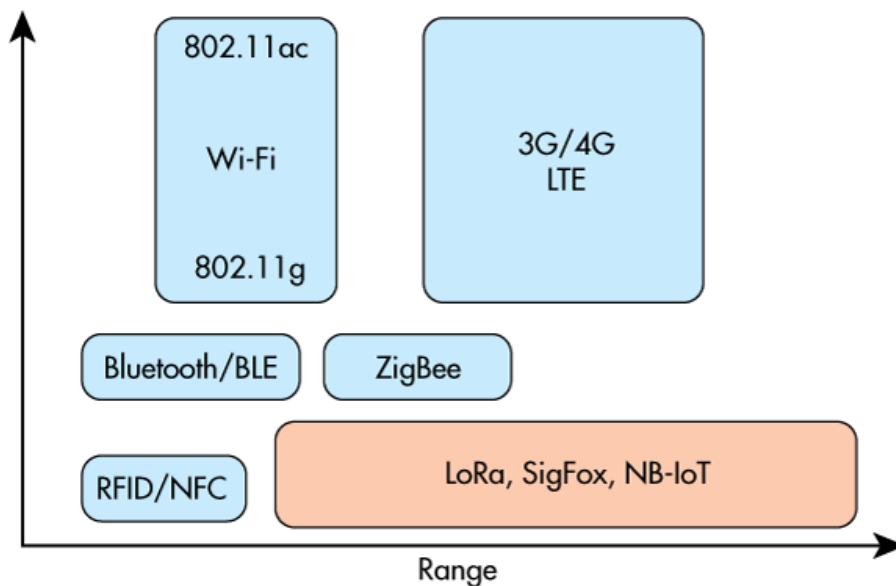


Figure 3 – Range Vs. Data Rate for IoT Wireless Technologies

There are many communication technologies used in IoT networks, the most notable can be seen in figure 3 above. The figure shows how the range of the signal compares to the data rate. Firstly, you have NFC (Near field communication). The technology for NFC was developed from that of RFID, you have an NFC chip in one device and once it comes into proximity(>10cm) to a second chip small amounts of data can be transferred. The chips run on very low amounts of power and can also run passively making it much more power-efficient than other methods of wireless communication however the tiny distance in which you can do this is a very limiting factor. Although this can be seen as an advantage in terms of security as you cannot intercept the traffic at range. [8]

There are two major versions of Bluetooth technology at present, Bluetooth Low Energy and Bluetooth Basic Rate/Enhanced Data rate. Low energy has been developed for very low power operation using the 2.4Ghz frequency band. The data transfer has been based on a frequency-hopping spread spectrum approach over 40 channels. An advantage of this is that the signal is much more resilient to interference and more difficult to intercept than standard static frequency transmissions. The technology can support data rates from 125 Kb/s to 2Mb/s. BT Low Energy supports multiple network topologies, which include point-to-point data transfer, broadcast, and can be used in a mesh topology for creating large-scale device networks, such as a wireless sensor network especially with its theoretical line of sight range of up to 400m however in practice it would be less. The technology uses

only up to 5% of the amount used by Bluetooth Enhanced Data Rate. The Bluetooth Basic Rate/Enhanced Data rate technology uses 20 times for energy than its counterpart however this allows for a much greater data rate ranging from 1 Mb/s to 3 Mb/s. It is even more resilient to interference using 79 channels for its frequency hopping spread spectrum set-up. However, these advantages come at the cost of greater power consumption and decreased range no more than 100m at best, this technology supports point-to-point data transfer and is optimized for audio streaming. [9]

Zigbee has been traditionally used mainly in an industrial setting, however recently with the influx of IoT and the desire for wireless technologies it is branching out of this area. Zigbee Pro, one of the many Zigbee profiles, is based on the IEEE802.15.4 which is the industry-standard wireless networking technology which operates at 2.4GHz. It is aimed at applications that require moderately infrequent data exchange at low-data rates within a range of approximately 100m. Zigbee offers low-power operation, high security and high scalability allowing high node counts. This makes it very good as a communication method between IoT devices such as sensors. [10]

Cellular data transfer as a communication method is very well-known via use in mobile phones. For the purposes of IoT devices it can be suitable for operation over very long distances using GSM/3G/4G. These technologies are capable of sending high quantities of data, more so 4G, however the cost of this both financially and in terms of power consumption are very high and often too much for IoT applications with the exception of projects such as sensor-based low-bandwidth data at a long range. The range on the devices is up to 35km using GSM and up to 200km using 3G and these rely on infrastructure already being in place within that range. [10]

LoRaWAN is a MAC (Media Access Control) protocol for wide area networks. The name is based on Long-Range Wide-Area-Network. It is one of the rising stars in the IoT world as it is designed to allow devices to communicate with internet-connected applications over long distances wirelessly. The range of communication is usually between 5-15km however there is a world record for 202km line-in-sight-connection. The technology uses very low power but with this low power comes low bandwidth of only 51 bytes per transmission. But this is ideal for projects such as sensors as all you need to transmit is readings back to a central server. LoRaWAN specifies that all devices using it must fall into one of 3 classes. All LoRaWAN devices must implement class A. A class A device supports bi-directional communication between a node and a gateway. Uplink messages may be sent

from the node to the gateway at any time, however responses from the server via the gateway can only take place during two windows immediately after uplink if these are misses then the next opportunity will be after the next uplink. Class B devices add to Class A by adding scheduled receive windows in addition to those after uplink. Class C devices extend on A by keeping receiving windows up unless they are transmitting making them effectively available all the time unless transmitting. [11]

Sigfox is a company that's runs a similar technology to that of LoRaWAN. However, the major difference is that rather than installing your own LoRa gateway, Sigfox provides coverage on a rental basis and has systems in place to handle the data on receiving. Like LoRaWAN, Sigfox uses ultra-narrow band radio modulation to transmit messages. Each message is 100 Hz wide and is transferred at a data rate of 100 or 600 bits per second. Sigfox is tailored to handle small messages only, meaning less energy consumption and longer battery life. Unlike cellular protocols a device does not attach itself to a specific base station, instead it broadcasts messages to be received by any base station in range, therefore reducing the risk of losing connection. [12]

## 4 Approach

### 4.1 Libelium Wasmote

A sensor node or mote as it is sometimes referred can be defined as *“A basic unit in a sensor network, with on-board sensors, processor, memory, wireless modem, and power supply.”* [13] The nodes accessible for this project are those developed by a company specialising in sensor networks, Libelium Comunicaciones Distribuidas S.L.. [14] The sensor node created by Libelium is called the ‘Wasmote’, it is an incredibly versatile node and is the base device that nearly all of Libelium’s sensors connect to via various sensors connect to through interchangeable sensor boards.

The device itself is reasonably small with the dimensions 73.5 x 51 x 13mm and weighing 20g. It uses the ATmega1281 microcontroller which is high-performance and low-power which is ideal for a sensor node. It has a 128KB flash memory which is more than large enough for the size of program that are required to operate a sensor node. [15] The Wasmote has a micro SD card slot which can take up to 8GB. It has a mini USB port which can be used for both data transfer as well as powering the device, there is an LED to show that USB power is in use. There is a power switch as well as a hibernation switch, by using hibernation as the method of energy saving it can allow the node to operate for up to a year without recharging as the power drops from 17mA when running to just 7µA in hibernation. As well as the battery input you have an input for a solar panel which can charge a battery, this would allow the Wasmote to run indefinitely without USB charging if conditions were correct. There are three LEDs that can be programmed to flash or come on with the programming of the node. Built-in on the board there is an accelerometer which operates in both normal and low power mode. [16]

The Wasmote has three main connectors for attaching peripherals. There are two radio sockets, in socket one you are able to connect the following modules: 802.15.4/Zigbee, LoRaWAN, Sigfox, LoRa, WiFi Pro, GSM/GPRS, 3G, 4G, Bluetooth low energy, Bluetooth for device discovery, RFID/NFC. To use the second socket, you are required to connect an expansion radio board to the socket. Using the expansion board allows you to make combinations of communication modules which could be used for many reasons for example Multifrequency Sensor Networks: (2.4GHz – 868/900 MHz). The third connector is for connecting sensor and sensor boards. There are twelve different sensor boards that

can be connected to the Waspote: Gases, Gases Pro, Events, Smart Water, Smart Water Ions, Smart Cities Pro, Smart Parking, Agriculture, 4-20mA current Loop, Video Camera, Radiation, and prototyping. This range of sensor boards allows for a huge amount of scope, the radiation board can only be connected to a Geiger tube so is quite limited but something like the Smart Cities Pro is compatible with over twenty sensors. [16]



## 4.2 Wasmote Software Development Kit

Libelium offer a software development kit in order to assist the programming of the Wasmotes. The SDK consists of the Wasmote Pro API and the Wasmote Pro IDE. The API is made up of 56 open-source libraries which allow for ease of programming with all Libelium Sensors and communication technologies. Wasmote sketches are written in the C++ programming language, these do not need to be done using the IDE however the IDE allows for very easy compiling and uploading of the sketches. As well as writing, compiling, and uploading programs, the IDE includes a serial monitor which is very useful for debugging as you can program the Wasmote to send data via USB and see if the sensors are returning the correct data. Or alternatively you can use a USB gateway to test the setup and programming of the radio module on the Wasmote for example checking to see if you have set up the Zigbee module correctly. You could also use a program to store the data coming in through serial if you didn't want to create a sensor network and only wanted data from one sensor. The IDE also supports exporting the sketch to an OTA file. OTA is Over The Air programming and it allows for Wasmotes to be updated over a network via FTP. This would be ideal if you needed to update Wasmote on a large network or if the Wasmotes location is inaccessible. [17]

The programs that run on the Wasmote have a distinctive syntax. Rather than have 'main()' function as the initialisation of the program as Wasmote sketch has two main functions. Firstly, you have a 'setup' function, this function is only run once when the code is initialised or if the Wasmote is reset. The initialisation of the modules such as the sensor boards and comms modules need to go in this function as well as any other code that should be run on start up. The second function is the 'loop', this function runs continuously forming an infinite loop. This functions purpose is to measure and send information and it should also include a method of sending the Wasmote to sleep and monitoring interrupts to wake it back up. [18]

### 4.3 Meshlium

Libelium offers its own Access Point/Gateway called Meshlium. There are 13 models, all models act as a WiFi(2.4Ghz) Access Point with ethernet and then each of them has a different configuration of the following technologies: Wifi Mesh (2.4/5Ghz), 3G/GPRS/4G, Zigbee, and some acting as Gateway nodes. There is a model that includes all technologies. In addition to this, Meshlium can also have a GPS module integrated into it allowing for mobile and vehicular applications. It is powered by Power Over Ethernet,

Meshlium Storage Options

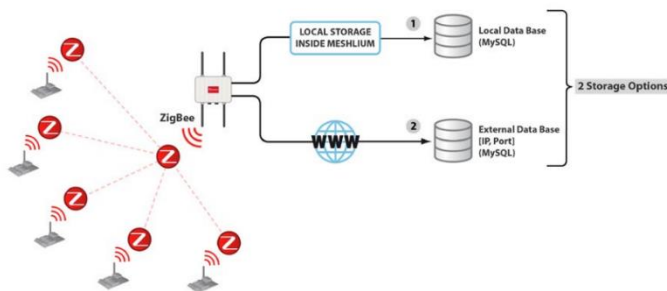


Figure 4 – Meshlium Network Diagram

for storage of data collection. The operating system for the device is Debian Linux and it runs the open source Meshlium Manager System, you are therefore able to install any application or service required on to the device. The Meshlium interacts with the Waspmotes by receiving the frames sent via Xbee, parsing these frames, and then either storing them in a local MySQL database, and External MySQL database, or sending the information directly IoT services on the internet as you can see in figure 4. Depending on the needs and expertise of the user the Meshlium can be controlled by an SHH console as direct access to the shell console is enabled.

Libelium offers connectors allowing this to be mains, solar or battery powered. It is contained within in aluminium IP-65 enclosure allowing it to be placed outdoors. The device can have disk memory of up to 32GB installed within it, this is for the system and

## 4.4 United Welsh Test Bed

The house belonging to United Welsh was surveyed to identify what possible sensor readings could be obtained from the building and how these readings could be useful for United Welsh and if these would be useful for the tenants. The property was three stories with two flats on each floor. The property at the start of the project was unoccupied awaiting final renovations. The intended occupants would be specially selected as people who were trying to get into work after a period of unemployment. On discussion with a representative from United Welsh about the needs of the company, occupants, and what equipment we had, that the testing of the proof of concept should be done in two parts. Firstly, setting up a Libelium wireless sensor network that took base readings of the temperature within the house whilst empty. This data could then be used by the building maintenance team to identify properties at risk of degrading faster than others by a drop in temperature. Then secondly, once occupied, look at temperature readings on the ground and top floor of the building to see how people using the heating on different levels of the property affects the temperature of the building as a whole. This would be achieved by getting data from the boiler smart meters. Then also having a PIR sensor watching the front door to indicate building temperature effects of the front door being opened and people moving around the entrance. This data could then be used to see if all how tenants are paying for their heating is fair or if for example if the ground floor tenant can pay more than others if the heat rises, or has to leave the heating on for longer in the year to maintain temperature.

## 5 Implementation

The implementation of this project has been split up into seven key tasks:

1. Set up Wasmotes and code to return readings via USB.
2. Set up the Meshlium Access Point.
3. Program Wasmotes to send data to Meshlium.
4. Setup a MySQL server to collect sensor data.
5. Direct sensor traffic from the Meshlium to a database in the MySQL server.
6. Run the first test of a single Wasmote in an unoccupied house.
7. Run the second test with three Wasmotes with an occupied house.

### 5.1 Setting up Wasmotes

The initial step in setting up Wasmotes was downloading and installing the Wasmote API and IDE, this is available from the Libelium website and is referenced in all their documentation. It was then the case to physically set up all the Wasmotes. For all three they would be using the Events sensor board V20. Therefore, each Wasmote would require one events sensor board, one 4GB microSD card, one battery pack, and either a temperature or PIR sensor. At this stage there was not a comms module connected as the focus is on ensuring the code is correct for taking readings rather than for communication. Once all these had been set up correctly the Wasmote factory code was uploaded to the Wasmotes. This code is found as an example within the IDE and was uploaded to ensure that there they were working correctly. These all proved to be working correctly once connected to the PC via USB, the only item of the factory test that failed was the networking as no radio module was detected on the board. This was expected as one had to be connected at this stage.

```
// Put your libraries here (#include ...)

void setup()
{
  // put your setup code here, to run once:
}

void loop()
{
  // put your main code here, to run repeatedly:
}
```

*Figure 5 – New Waspnote Sketch*

degrees centigrade from the returned voltage which is as follows.

$$Temperature(^{\circ}C) = (Voltage - 0.5) * 100$$

The setup part of the code is relatively small, just the initialisation of the USB return, initialisation of the sensor board and initialisation of interrupts from the sensor board. The interrupts initialisation is done by setting a number is volts to equate to a sudden temperature change that should trigger a reading. In this case it has been set to 0.77 which through the above equation is equivalent to 27 degrees Celsius. This may seem very high, but this is due to the desire to record readings by time rather than change on temperature so as to allow for chronological comparisons between sensors. But the threshold is kept in case of a sudden increase in temperature then the time of the incident can be noted.

The main loop is made up of three distinct parts. Firstly, you are just reading the temperature of from the sensor. This is done as seen below, the important part is the ‘SENS\_TEMPERATURE’ as this uses the library and reads the value through the equation for temperature. The value can be read without this however it would then just return the read voltage. This value in this case is then just printed via USB and can be seen through the serial monitor.

```
// Read the sensor output
value = SensorEventv20.readValue(SENS_SOCKET5, SENS_TEMPERATURE);
```

*Figure 6 – Sensor Output Snippet*

The second part on the main loop is to send the Waspnote into deep sleep. The length of sleep is entered in ‘Days:Hours:Minutes:Seconds’, in terms of this testing stage days would seem excessive but for long term applications you may only want one sensor reading per day and keeping the device in deep sleep increase the battery life of the device.

Once the time entered has elapsed, in this case ten seconds as seen below in figure 7, then it automatically wakes up and continues running the code in the loop.

```
PWR.deepSleep("00:00:00:10", RTC_OFFSET, RTC_ALM1_MODEL, SENSOR_ON);
```

*Figure 7 – Deep Sleep Snippet*

The final section on the main loop firstly checks to see if the interrupt is from a sudden change in temperature, if so a function is run that is outside of the main loop which prints the temperature at the interrupt via USB to the connected computer. Once it has done this the interrupt is cleared and the code in the main loop continues. Secondly it checks to see if the interrupt was just caused by the timer on deep sleep was expiring. If this were the case, then it would clear that flag and doing so would complete the loop and so it would return to the start of the main loop.

The programming for the PIR sensor is largely the same as that of the temperature sensor as mentioned above with only sensor related differences. Firstly, in the setup you need to stabilise the sensor. This is just where the its gets its base level of IR in the room. This stabilisation is done automatically by the sensor itself, but code needs to be put in place to ensure no readings are taken whilst this is happening, this is done by running a while loop until it is complete. Unlike the temperature sensor which is analogue the PIR sensor gives a digital signal back which makes this process easy as the while its stabilising it returns a voltage of 1. The major difference comes in the main loop as its reversed on which interrupts are important. In the temperature code the time was the important method of getting data. However, with the PIR sensor you still need a time interrupt to allow the IR reading to be updated as it will change over time with the conditions of the room. But when a major change happens, for example a person walking into the room, you want this to be recorded. So, in the separate interrupt function for the PIR this is where you would find the useful data of a presence being detected. [19] [18]

The programming of the first Waspnote took significantly longer than the first as it required a lot of review of the documentation provided by Libelium. Also with the Waspnotes being programmed in C++ it took quite some time to understand the syntax of the language and then understanding the Waspnote API as it did not used the C++ standard library as it was a language I'd never experienced before.

## 5.2 Setting up of Meshlium

Meshlium is an out of the box piece of equipment and so the setup is reasonably simple. The Meshlium available for use has four antenna sockets on it, two for WiFi (2.4GHz) and two for Zigbee Pro 802.15.4. The box come with only 2 antennas and as for this project both WiFi and Zigbee were in use one was placed in one of each socket. Meshlium is powered by Power Over Ethernet(POE), Libelium provide a POE injector with the box and this was plugged into the mains via a 12V DC power supply. Once connected to Meshlium via ethernet it takes around 90 seconds to boot up. There are no lights on the device but there are two audible beeps once booted up. Once booted Meshlium can be connected to via WiFi to allow configuration. Once connected to the open network the Manager system can be connected to through the URL: <http://10.10.10.1/ManagerSystem>. Once logged in a WPA-PSK passkey was assigned to prevent unauthorised connect to the network when deployed. This was the only AP setting that needed to be altered. After this the XBee-Pro 802.15.4 settings needed to be configured as to allow connection from Waspnotes. PAN ID (Personal Area Network ID) is the identifier for the network and needs to match that of the nodes of the network. This was just set to 1234 to make programming the Waspnote Zigbee Modules easier. The option for an AES encryption key for the Zigbee transmissions can be entered here, however one was not entered for the testing as no sensitive data or over the air programming was in place. Not was taken of the mac address for the Zigbee network card Meshlium as this would be needed later. The inbuilt MySQL database comes ready to go with Meshlium and so does not require any configuration. [20]

### 5.3 Programming Wasmote For Wireless Communication

There are 2 main steps for enabling the Wasmotes for communication, firstly setting up Zigbee Modules and installing them on Wasmote. Secondly, adjusting the programming of the Wasmotes to send sensor data via ZigBee.

In order to set up the Zigbee modules I downloaded a program called XCTU from Digi [21]. XCTU allows you to configure radio modules both wirelessly and connected to a computer via USB. The first step was to connect the first Zigbee Module to the USB gateway and connect this to the PC. On doing this a scan could then be run to search for the module via comm ports. Once detected the profile for the module is loaded allowing configuration of the various settings. As mentioned previously the PAN ID was set to 1234 on the Meshlium and so this was entered here. You are able to enter any encryption for the network through the program. If encryption was being used this would be the better place to enter rather than in code as the natural method for entering the encryption key with Wasmote in code is just in plaintext. Once the configuration is complete before removing the module a configuration profile was saved. This saved file was then loaded on to the next two modules to ensure they are configured in the same way. [22] Setting up the modules became a very time consuming task, it seems a simple task however all the documentation suggests that the modules would not need programming due to be configured correctly for Meshlium as sold. However, as these modules had been used several times previously they had been reprogramming from there factory state and this ended up being a time consuming part of the project.

Once the Zigbee modules were all programmed they were then connected onto the Wasmotes via the Radio 0 socket and had their antennas attached. The code written for the Wasmotes previously then needed to be adjusted to send data via Zigbee to Meshlium. This wasn't a major change in the coding and was the same for both the PIR and temperature Wasmotes. Firstly, had to define the destination MAC address, this is the MAC address of the Zigbee network card within the Meshlium, this is found on the Meshlium Manger in the radio settings. A Wasmote ID variable needed to be assigned, this needed to be different for each node, in this case used the convention of 'node\_1','node\_2' etc. As well as the Zigbee module the WaspFrame library was required. The Frame library allows you to form packets of data that can be sent wirelessly in a format that is understood by Meshlium. Frames can be in binary or ASCII format, the default is ACSII you would only tend to use



binary frames when concerned about bandwidth. The ASCII frame structure can be seen below in figure 8.

HEADER								PAYLOAD						
<=>	Frame Type	Num Fields	#	Serial ID	#	Wasmote ID	#	Sequence	#	Sensor_1	#	Sensor_2	#	... Sensor_n #

Figure 8 – Xbee ASCII Frame Structure

The frame type is generally whether the frame is ASCII or Binary, however there are specialist cases such as Alarm frames. The number of fields is setting the length of the payload, how many sensors are sending data. The serial ID refers to an ID that is hardwired into the Wasmote and cannot be changed, similar to a MAC address. The Wasmote ID is a user set plaintext description of the device. The sequence is the frame number and is used to check if any frames are missing when they arrive at the destination. The payload itself is whatever the sensor wants recorded for example with a temperature sensor it would be maybe “22°C”. In the setup part of the code the frame ID needed to be set, the frame ID is what would be the Wasmote ID in the ASCII frame. This is then put into EEPROM memory. Finally in the setup the Zigbee module needs to be initialised, this is done with the line ‘xbee802.ON ( ) ;’. In the ‘loop’ part of the program once a sensor reading has been taken a frame first needs to be created and the frame type assigned. Once the frame has been created it does not have a payload and is just the header with the number of fields set to 0. To add the sensor data, you need to add both the sensor type and the value as seen below in figure 9.

```
frame.addSensor(SENS_TEMPERATURE, value);
```

Figure 9 – Adding a value to a frame snippet

The type of sensor is important as it defines how many fields of data there are whether it is a simple sensor with one output such as temperature or a complex one with multiple parameters. Also, once sent to Meshlium the sensor type gets stored along with the value so you don’t need to guess or look up what has been sensed. You can add multiple sensors to the frame, in the case of these Wasmotes there is only one sensor on each. The complete frame is then sent to Meshlium with the following code.

```
xbee802.send( RX_ADDRESS, frame.buffer, frame.length );
```

Figure 10 – Sending a frame snippet

This code using the Zigbee library to send the frame just created. `RX_ADDRESS` is the MAC address of the Meshlium defined at the start of the code, the `frame.buffer` is the created frame and the `frame.length` is just the total length of the frame.

This frame creation and sending was then put it every place that a value was sent to USB in the original code. In addition to this all code that sent readings and messages to a PC via USB had to be removed to allow the program to run on a Wasp mote that was running on battery power with no computer connected else the program would crash.

## 5.4 Setting up MySQL Server

To be able to make use of the data from the sensors the data needed to be retrieved from the Meshlium. In order to do this in a way that would work on large scale as for the proof of concept for this project it seemed that the best solution would be to get the Meshlium to send the data to an external database.

To set up the server, MySQL Community server 5.7 [23] was downloaded, this is an open source database download which is compatible with Meshlium. The full version of the server was downloaded including MySQL Workbench 6.3 which would be used to administer the database. Once installed the wizard was used to set up a new server, including opening port 3306 to allow MySQL traffic through the firewall.

Once the server was operational the database needed to be created. This database needed to follow the schema of the data parser within Meshlium, on the Meshlium Manager System it gives the following MySQL queries to create the database. [24]

**Just copy paste:**

```
CREATE database MeshliumDB;
```

**Just copy paste:**

```
CREATE TABLE IF NOT EXISTS `sensorParser` (  
  `id` int(11) NOT NULL auto_increment,  
  `id_wasp` text character set utf8 collate utf8_unicode_ci,  
  `id_secret` text character set utf8 collate utf8_unicode_ci,  
  `frame_type` int(11) default NULL,  
  `frame_number` int(11) default NULL,  
  `sensor` text character set utf8 collate utf8_unicode_ci,  
  `value` text character set utf8 collate utf8_unicode_ci,  
  `timestamp` timestamp NOT NULL default CURRENT_TIMESTAMP,  
  `raw` text character set utf8 collate utf8_unicode_ci,  
  PRIMARY KEY (`id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
```

*Figure 11 – Queries to setup database*

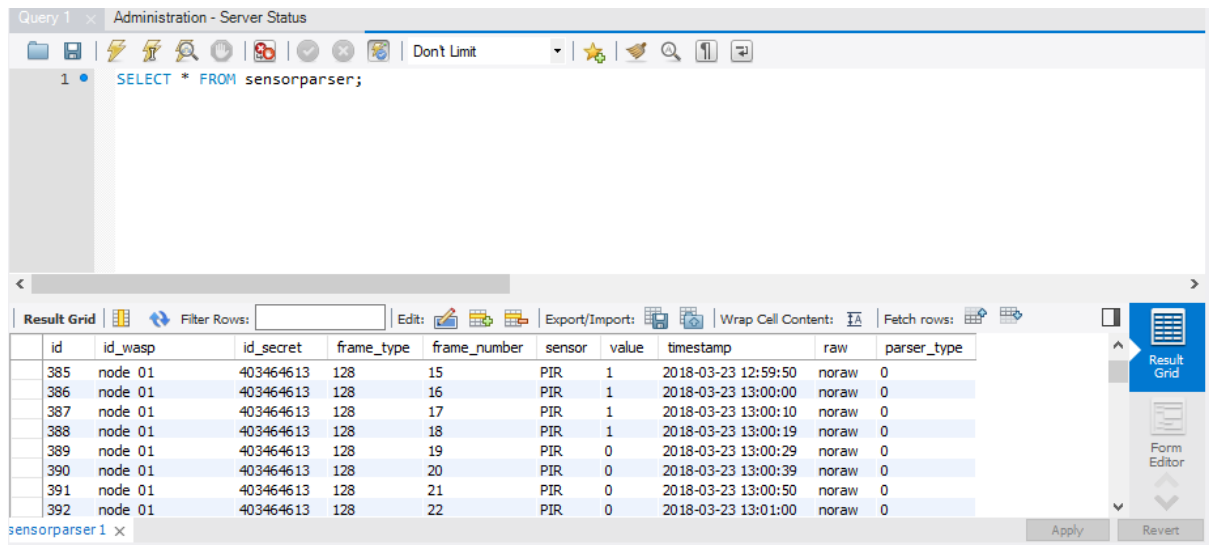
Once both the MySQL queries above had been executed there was a blank database identical to that of the sensor parser database within Meshlium.

### 5.4.1 Connecting Remote Database to Meshlium

Devices connected to Meshlium are dynamically assigned IP addresses unless a static one is set. In order for the database to be reachable a static one must be set for the server. Free IP addresses begin at 10.10.10.10, as there will be no other wifi devices connected to the network this one was chosen. With Meshlium powered up the laptop with the server

running on it was connected to the Meshlium Wifi network. Then in the System Manager for Meshlium there is an option for connecting to an external database. Using the details from the above MySQL query in figure 11 you can get the information required to connect and then test the connection. It's was then just a case of ticking the box to store frames in an external database and set a time of how often to synchronise. Once done the database was ready to receive sensor data.

To test the database the PIR sensor Wasmote was turned on to see if data arrived.



The screenshot shows the MySQL Workbench interface. The top toolbar includes icons for file operations, query execution, and window management. The query editor contains the command: `SELECT * FROM sensorparser;`. Below the editor, the 'Result Grid' tab is active, displaying a table of sensor data. The table has columns: id, id\_wasp, id\_secret, frame\_type, frame\_number, sensor, value, timestamp, raw, and parser\_type. The data shows PIR sensor readings from node 01 at various timestamps, with values alternating between 1 and 0.

id	id_wasp	id_secret	frame_type	frame_number	sensor	value	timestamp	raw	parser_type
385	node 01	403464613	128	15	PIR	1	2018-03-23 12:59:50	noraw	0
386	node 01	403464613	128	16	PIR	1	2018-03-23 13:00:00	noraw	0
387	node 01	403464613	128	17	PIR	1	2018-03-23 13:00:10	noraw	0
388	node 01	403464613	128	18	PIR	1	2018-03-23 13:00:19	noraw	0
389	node 01	403464613	128	19	PIR	0	2018-03-23 13:00:29	noraw	0
390	node 01	403464613	128	20	PIR	0	2018-03-23 13:00:39	noraw	0
391	node 01	403464613	128	21	PIR	0	2018-03-23 13:00:50	noraw	0
392	node 01	403464613	128	22	PIR	0	2018-03-23 13:01:00	noraw	0

Figure 12 – Recorded Data in Workbench

As you can see above there are PIR readings that have made it from the Wasmote to the MySQL server through Meshlium. The readings are at id 385 as there are lots of records from testing with just the Meshlium internal database. Now that this was in place the wireless sensor network was fully operational and ready for deployment.

## 6 Results

Now that the wireless sensor network was fully operational it could now be put in to the United Welsh Testbed to try and identify if useful data could be recorded from within the home. The first test to be conducted as mentioned previously will be with the building empty and just monitoring how and if the temperature fluctuates in an empty property with no heating on and no bodies entering the building. This was hoped to be done during the mid to end of March 2018, however due to disputes between United Welsh and building contractors this goal could not be achieved and it was not until mid-April. The interim time was used to ensure the second test could be run immediately following the first depending on the state of renovations and acquiring tenants for the property.

Access was gained to the property on Connaught Road in Cardiff between 09:00 and 10:00 on Wednesday 11<sup>th</sup> April 2018. This was only access available until the sensors would be retrieved at the same time of Friday 13<sup>th</sup> April 2018. On access the Meshlium was connected and allowed to boot up. Once this was complete the laptop running MySQL server was connected to the Meshlium and the server was started up. One of the temperature Waspnotes was connected the laptop and the program was reuploaded to ensure that the correct program was running. After this it was disconnected and running off just battery power. Just before needing to leave the property a query was run on the MySQL server via MySQL workbench to select all records from that day, this showed that temperature readings were being taken every minute and relayed through Meshlium to the MySQL server.

On arriving back at the property, I performed a quick check to ensure data was still being recorded to the MySQL server. Then powered down the Waspnote and Meshlium, a copy of the recorded data was backed up on a USB flash drive as a .csv file from the MySQL database. The data could then be analysed away from the test bed.

Using MySQL workbench three queries were run to get a data set for each day the temperature sensors were running inside the house. These datasets were then exported to .csv files. From these files visualisations of the data could be created as seen below.

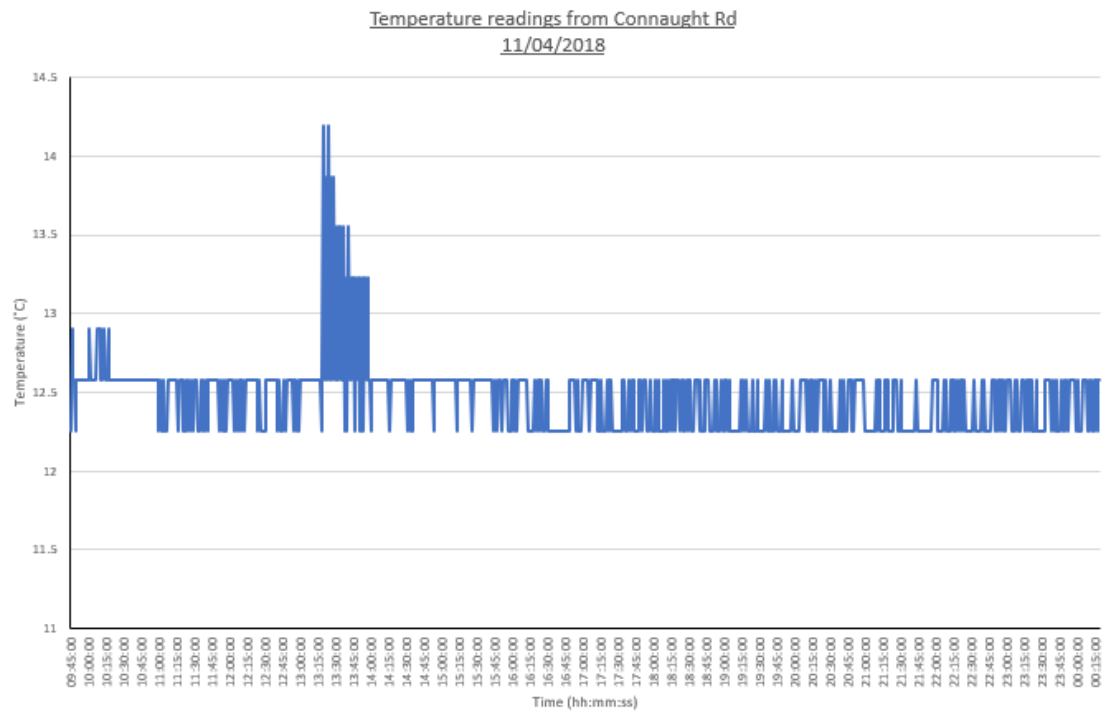


Figure 13 – Results Day 1

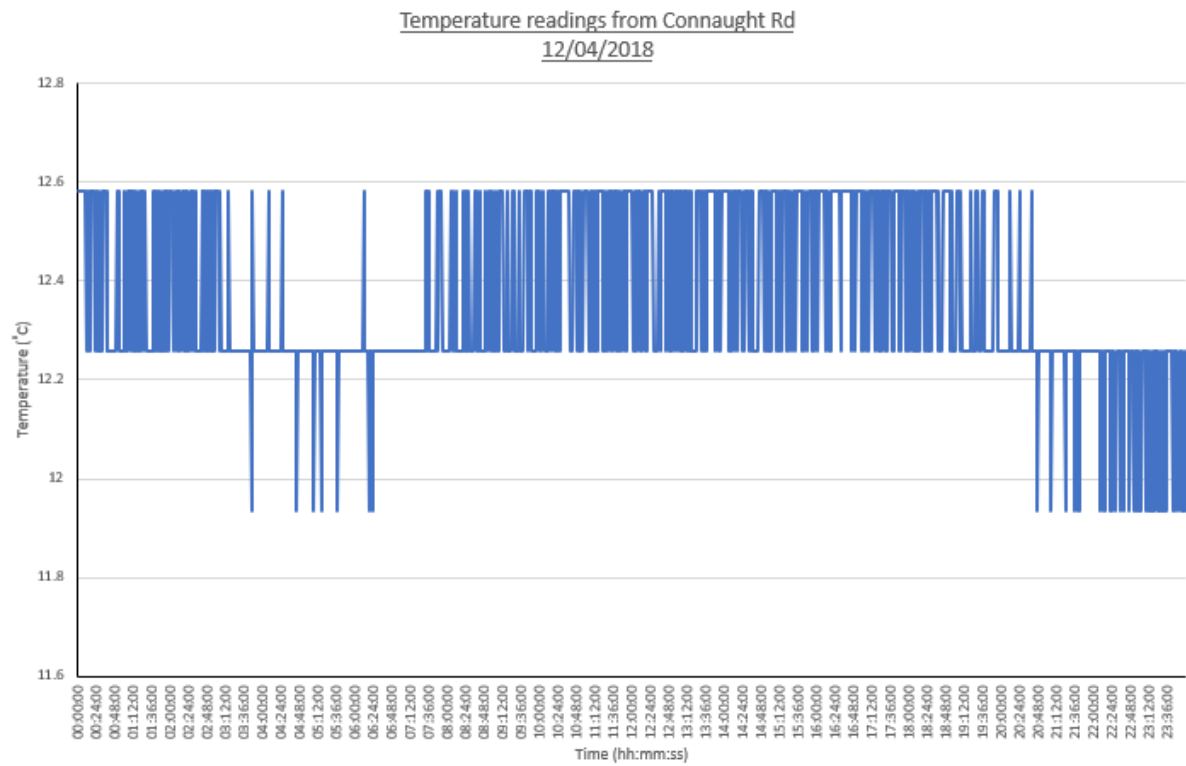


Figure 14 – Results Day 2

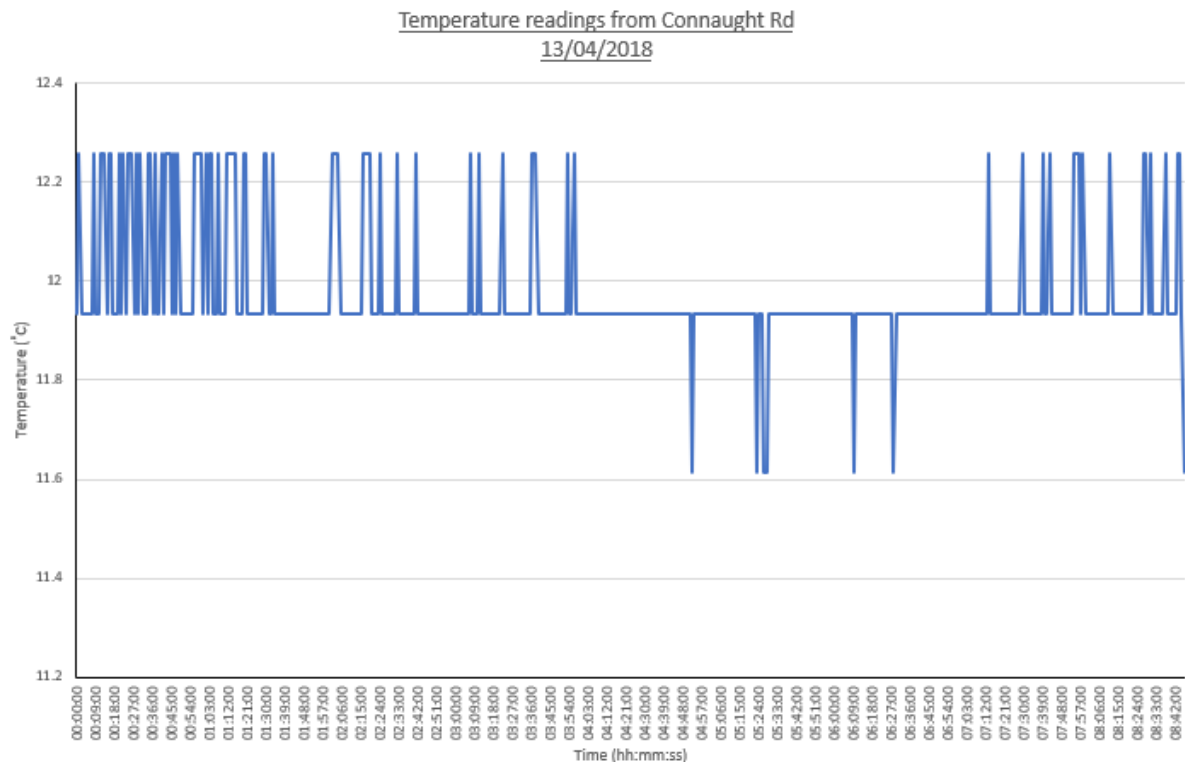


Figure 15 – Results Day 3

These readings show that the temperature over the three calendar days remained quite consistent with staying between 11.9°C and 12.6°C for most of the test. However, on the 11/04/2018 at around 1300 there is a peak to 14.2°C which matches the time of the only bright sunlight through the testing. Which suggests that the temperature of the house is directly affected by sunlight outside. Readings of this nature could be useful in unoccupied properties for housing companies such as United Welsh. With earlier discussion with a representative of United Welsh as mentioned the maintenance team were interested in the use of Wireless Sensor Technology in order to monitor the state of unoccupied houses and the temperature reading could be very useful in obtaining the risk of mould growth as:

*“The amount of moisture in one cubic metre of air varies as a function of the air temperature. Warm air can absorb more moisture than cold air. At a room temperature of 20° C and a relative humidity of 60 % one cubic metre of room air holds 10 g of water. If the room temperature is reduced to 8° C the air in the room can only absorb half the amount of moisture. This means that 50 % of the moisture escapes and precipitates as condensation water, mostly on the cooler outside walls. The risk of mould growth therefore increases at these points.” [25]*

This suggestion of how temperature can affect the risk of mould growth shows how the use of temperature sensors within unoccupied properties can be of benefit to the operating company. I am quite confident that if deployed on a large scale of unoccupied

properties it would be possible to identify the most at risk properties from the data received. Although for more confidence in results and less error it could be advantageous to add more sensors to this such as humidity and a sensor for the wall moisture.

The second and more major test of the usefulness and operational capabilities of the system was to be when the house became occupied. This was to see how a wireless sensor network in a home scenario could benefit both the housing company but also the tenants of the building. Using the Waspotes already programmed, there would be a temperature sensing Waspote on both the ground floor and the second floor. There would also be a Waspote sensing PIR on the ground floor with the sensor facing the door. This test designed to test temperature differentials around the house when the front door was opened. This data, if proved it made a difference, could be used to work out if tenants on the ground floor were overtime paying more for utilities with heat escape through the front door but also with heat from the ground floor rising to higher floors. This information would be acquired by looking at the temperature differences between floors and comparing that to when the boiler had been turned on for heating in the individually controlled flats. Which could be identified by looking at the data from the smart meters. The programming for the Waspotes was in place for this testing however the disputes between United Welsh and the building contractors meant that the building work that was due to take place in February was still not complete in the timescale of this project. Therefore, this test could not take place. Tests were conducted with all these sensors to show they could work and the physics behind the idea of the temperature differentials suggests that this data could prove useful.

Even without the second test, the first test shows that the Wireless sensor network could be setup and run within a home environment and meaningful data was obtained. This was only with a very limited selection of sensors, only one in the first test. Further sensors would have obtained further results.



## 7 Future Work

So far, a wireless sensor network has been setup and testing an unoccupied property and yielded promising results. The next step will be to implement the second test that was mentioned in an occupied multi-flat house. All the programming and infrastructure is in place for this test it is just waiting on occupation of the testbed on Connaught Road. Installation of these sensors could be done in less than an hour.

Regarding the first test it would be advantageous to add humidity sensors and a moisture sensor for the walls. From this data it should be possible to work out which figures of each sensor leave properties most at risk and should be given preventative maintenance. Once this is the case a Graphical User Interface in the form of a webpage could be created so that the maintenance team at United Welsh can just log-on and see a ranking list of properties showing the most likely to fall into disrepair.

With a successful test of an occupied building more sensors could be tested within the home. An example talked about with a representative from United Welsh would be the installation of a carbon monoxide detector at the boiler. This would be advantageous as the Libelium Smart Gases Carbon monoxide detector has the resolution of 30-1000 parts per million. [26] One of the most popular in-home carbon monoxide alarms doesn't register until 50 parts per million. [27] A person does not begin to feel affects until 100 parts per million. The use of this sensor in a home could aid the maintenance team to identify defective boilers before they become dangerous, making the job easier and more cost effective for the company but also safer for the tenants.

Once a selection of useful sensors has been identified it would be the natural step to look at a larger scale test over multiple properties. Could be done with a Meshlium device at each home using Zigbee for the Wasmotes and sending data to a central server over the internet. Or alternatively it could be done by having a single Meshlium in a central location and could possibly use LoRaWAN and the communication technology which would exceed the Zigbee maximum distance of around 100m up to 5km in an urban area. Successful deployment would warrant a graphical user interface to allow both tenants and the operating housing company to access the data in a user-friendly way. This information would then be able to aid people in how they use things like heating in their homes as well as help the housing company work out bills in relation to split payments for single properties.

## 8 Conclusions

The main aim of this project was to prove that a wireless sensor network can be deployed successfully in rented accommodation and achieve meaningful results. This aim was partially achieved with provision in place for it to be achieved in full. A wireless sensor network was successfully deployed into unoccupied rented accommodation. Meaningful data was gathered from the unoccupied house which is in itself a success. However, this data was only gained over two full days and a greater amount of data over a longer period would have yielded more confident data, but this was not possible due to two days being the longest amount of time being available for this test. Also, the data being successfully obtained has led to idea that by adding more sensors to the unoccupied house you would gain much more actionable data regarding building management.

Regarding deploying a wireless sensor network in occupied housing, no conclusions can be definitively drawn as it is only theoretical at present. Although this aspect of the project is not a failure as all these sensors were programmed and tested with the equipment that would be used in the testbed at another site. Therefore, these sensors could be deployed to obtain the required data at any time in the future. Based on the results on the unoccupied property I can suggest that the results of the occupied test would be successful but without experimental data to provide evidence this is just speculation.

The project in its entirety I would describe as a success, although not all the aims were achieved valuable insight was found. There were no great failures in the project just time restrictions based on testing being done in real-world conditions where circumstances were beyond the control of project. The data obtained, and the insight gained from the process of implementation has allowed recommendations to be drawn for future work regarding wireless sensor networks within rented accommodation, with benefits for both the tenants and housing association. These future works I am sure would provide further promising insight into the use of wireless sensor networks within the home.

## 9 Reflection on Learning

There were four main objectives to this project, firstly, to become familiar with wireless sensor networks and the communication technologies used. Through background research and from learning throughout the implementation process I believe this objective was met and that it provided benefit to the overall process of the project. I have gained a much deeper understanding of the wireless sensor networks as well and a broad understanding of the main wireless technologies used in IoT with a much deeper understanding of Zigbee 802.15.4.

The second objective was to understand how Wasmotes worked and gain and understanding of the C++ required. I believe this has been one of the hardest parts of the project having never programmed with sensors nor in C++. However, through the use of Libelium's excellent documentation, by the end of this project I would describe myself as proficient in programming Wasmotes in C++. Although one of the hardest parts of the project it was enjoyable to do.

The third objective to this project was to create a wireless sensor network with an external database. I managed to complete this successfully. This part of the project took a lot of time but was a lot of fun and it was incredibly satisfying when all these different areas of the project I had been spending so much time on came together to form a single functioning system. There was difficulty in setting up the Meshlimum initially due to how difficult it was to find the documentation for the one in the Cardiff Computer Science department. This was due to Libelium only making the most recent version of documentation easily available and the module in use for this project was several versions behind. Also, I had used MySQL before but had never administered a server. This was an interesting part of the project and I'm sure will be useful knowledge in the future.

The final part of this project was to draw conclusions of the effectiveness of the project. I believe I did this to the best of my abilities with the knowledge at hand as I was unable to perform all the testing I had hoped for. That has been a good learning point for me though. Throughout university all testing you need to do is under lab conditions where you have all the control. This gave me insight into real world testing where you might have your own schedule, but it is reliant on other people and other companies to be able to perform what you need to do. However, I feel as if I managed to work around these issues reasonable well, for example with the delay before the first test rather than waiting around

for the test to take place I prepared for the second test in case I was able to perform it close to the first one so as not to waste time.

I chose to this project as the area of IoT and wireless sensor networks incorporates so many different technologies and software. I was interested in the area of computer science my knowledge didn't expand much past that of what was in the syllabus during my education at Cardiff University. This seemed like the perfect opportunity to explore an area I believed I would enjoy learning more about. In that sense I was correct and have thoroughly enjoyed spending time researching and playing with this area. To such an extent I have started to attend meet-ups of Cardiff's IoT businesses and enthusiasts, this has only further added to my interest in this area and possible future careers.

In reflection on this project there are certain things I did that in hindsight I should have approached differently. When it came to use the Libelium kit I believe I should have done more research on wireless sensor networks generally, as I dove into the Libelium sensor kit very fast and then coming back to looking at the area generally was very difficult as I became a bit focussed on the Libelium way of doing things. Also, in the coding stage whenever I had a coding issue I become to fixated on the issue and rather than taking a step back to rethink my approach became bogged down and what turned out to be a simple problem often from rushing into something. In future I'd take my time and plan all the code out a lot more before implementing it. I think a lot more time would have been saved in this way. Also, in hindsight it would have been more advantageous when it came to the second test of the network in an environment with people to have set it up somewhere else such as my home. Although the data would not have been applicable to the original multiple occupancy model it could have still yielded some interesting results and would have given more information. But I held out too long for the dispute between United Welsh and the contractors, however I don't feel that it has hindered the project that much as it is still something that can be performed with United Welsh. The final aspect of the project I disagree with my method in part with is the writing of the final report. I chose to write up the report once the project had been completed so as to be able to write it with a consistent tone. I don't disagree with this approach completely as I find I tend to write reports better when doing it as a single project however I feel that I would have benefitted from better note taking throughout the whole of the project. This would have made bringing the report together much easier.

In conclusion I have thoroughly enjoyed undertaking this project and I feel I have not only given to the project but have taken a lot away from it. The skills and methodologies of my work I have commented on in hindsight I am sure will be useful in my career. I Also hope the project I have undertaken will have some benefit to United Welsh's endeavour to use IoT devices to better manage their properties as well as giving benefit to the tenants directly.

# Bibliography

- [1] K. Rose, S. Eldridge and L. Chapin, "The Internet of Things: An Overview," The Internet Society (ISOC), 2015.
- [2] C. I. P. S. Mitch Maiman, "Product Design & Development," 3 December 2015. [Online]. Available: <https://www.pddnet.com/blog/2015/03/4-key-elements-iot>. [Accessed 16 April 2018].
- [3] D. S. Yinbiao and e. al., "Internet of Things," IEC, Geneva, 2014.
- [4] Undersea Warfare, "Undersea Warfare," Winter 2005. [Online]. Available: [http://www.public.navy.mil/subfor/underseawarfaremagazine/Issues/Archives/issue\\_25/sosus.htm](http://www.public.navy.mil/subfor/underseawarfaremagazine/Issues/Archives/issue_25/sosus.htm). [Accessed 5 May 2018].
- [5] Silicon Labs, "The Evolution of Wireless Sensor Networks," 2013. [Online]. Available: <https://www.silabs.com/documents/public/white-papers/evolution-of-wireless-sensor-networks.pdf>. [Accessed 24 3 2018].
- [6] E. Elahi and A. Gschwender, ZigBee Wireless Sensor and Control Network, Upper Saddle River: Prentice Hall, 2009.
- [7] National Instruments, "National Instruments," 24 August 2016. [Online]. Available: <http://www.ni.com/white-paper/7142/en/>. [Accessed 17 April 2016].
- [8] C. Faulkner, "Tech Radar," Future Plc, 9 May 2017. [Online]. Available: <https://www.techradar.com/news/what-is-nfc>. [Accessed 24 April 2018].
- [9] Bluetooth SIG, "Bluetooth," 2018. [Online]. Available: <https://www.bluetooth.com/bluetooth-technology/radio-versions>. [Accessed 24 April 2018].
- [10] RS Components , "Design Spark," RSA, 21 May 2012. [Online]. Available: <https://www.rs-online.com/designspark/eleven-internet-of-things-iot-protocols-you-need-to-know-about>. [Accessed 24 April 2018].
- [11] The Things Network, "LoRaWAN Overview," 2018. [Online]. Available: <https://www.thethingsnetwork.org/docs/lorawan/>. [Accessed 5 3 2018].
- [12] Sigfox, "Sigfox Technical Overview," 2018. [Online]. Available: <https://www.sigfox.com/en/sigfox-iot-technology-overview>. [Accessed 4 May 2018].
- [13] IGI Global, "IGI Global," [Online]. Available: <https://www.igi-global.com/dictionary/basic-concepts-wireless-sensor-networks/26487>. [Accessed 29 April 2018].
- [14] Libelium, "Libelium," [Online]. Available: <https://www.libelium.com/>.
- [15] Microchip, "Microchip," 2018. [Online]. Available: [www.microchip.com/wwwproducts/en/ATmega1281](http://www.microchip.com/wwwproducts/en/ATmega1281). [Accessed 30 April 2018].
- [16] Libelium , "Wasp mote Datasheet v7.2," October 2017. [Online]. Available: [www.libelium.com/development/waspote/documentation/](http://www.libelium.com/development/waspote/documentation/). [Accessed 16 March 2018].

- [17] Libelium, “Waspote IDE v07 User Guide,” February 2007. [Online]. Available: [www.libelium.com/development/waspote/sdk\\_applications](http://www.libelium.com/development/waspote/sdk_applications). [Accessed March 2018].
- [18] Libelium, “Waspote Programming Guide v7.0,” February 2017. [Online]. Available: [www.libelium.com/development/waspote/documentation/programming-guide/](http://www.libelium.com/development/waspote/documentation/programming-guide/). [Accessed March 2018].
- [19] D. Gascon and M. Calahorra, “[Ev\_5] - Socket 5 Reading for Gases v20,” Libelium Comunicaciones Distribuidas S.L., 2015.
- [20] Libelium, “Meshlium Datasheet V4.0,” February 2013. [Online]. Available: [http://www.libelium.com/uploads/2013/02/meshlium-datasheet\\_eng.pdf](http://www.libelium.com/uploads/2013/02/meshlium-datasheet_eng.pdf). [Accessed March 2018].
- [21] Digi International Inc., “Digi,” 2018. [Online]. Available: <https://www.digi.com/>. [Accessed 7 March 2018].
- [22] Digi, “XCTU User Guide,” 12 July 2017. [Online]. Available: [https://www.digi.com/resources/documentation/digidocs/90001458-13/default.htm#concept/c\\_90001458-13\\_start.htm%3FTocPath%3D\\_\\_\\_\\_\\_1](https://www.digi.com/resources/documentation/digidocs/90001458-13/default.htm#concept/c_90001458-13_start.htm%3FTocPath%3D_____1). [Accessed 10 March 2018].
- [23] Oracle Corporation, “MySQL,” 2018. [Online]. Available: <https://dev.mysql.com>. [Accessed 12 March 2018].
- [24] Libelium, “Wireless Sensor Networks,” November 2014. [Online]. Available: <http://www.libelium.com/development/waspote/documentation/>. [Accessed March 2018].
- [25] Inoutic, “Humidity and mould growth in living areas,” 2018. [Online]. Available: <http://www.inoutic.de/en/tips-on-window-purchase/ventilation/humidity-and-mould/luftfeuchtigkeit-schimmelpilzbildung.html>. [Accessed 04 May 2018].
- [26] Libelium, “Smart Gases 3.0 Technical Guide v7.1,” February 2017. [Online]. Available: <http://www.libelium.com/development/waspote/documentation/gases-board-technical-guide/>. [Accessed 04 May 2018].
- [27] Life Saver, “Lifesaver Battery Carbon Monoxide Alarms,” 9 February 2015. [Online]. Available: [http://documents.4rgos.it/v1/static/7001617\\_R\\_D001A](http://documents.4rgos.it/v1/static/7001617_R_D001A). [Accessed 4 May 2018].

# Appendix

## Note from United Welsh

*United Welsh are a Housing Association based across 11 different local authorities in South Wales providing just under 6000 homes. We approached Cardiff University computer science department to help us look at how IOT sensors can be used in our homes to help us gain data and insight into how we can better maintain and repair our properties for the benefit of our tenants. We used one of our properties on Connaught Road in the Roath area of Cardiff as a test bed for the sensors. The property is currently empty and awaiting new tenants so this offered the chance to use sensors on the property as a baseline and also to consider what we can learn from the effects over time in the condition of the property.*

Ms L Shute  
United Welsh,  
13 Beddau Way,  
Caerphilly,  
CF83 2AX