Using Environmental Sensors with Mosquitto, Python, and other dangerous components

Steve Cosgrove
Whitireia New Zealand
Wi Neera Drive, Pvt Bag 50910
Porirua
+64 4 2373100 ext 3606
steve.cosgrove@whitireia.ac.nz

Andrew Hornblow
Educational Consultant - Internet of Things
80 King Street, Opunake
+64 6 761 7122
andrew.hornblow@gmail.com

ABSTRACT

An environmental sensor network is proposed in this paper, using low cost sensor and processor requirements. The MQTT protocol is used to transfer data over a range of underlying network infrastructure. An innovative hardware solution is proposed. The Mosquitto broker software is used to receive and pass on messages from the sensors. Python code is used to publish data to the brokers, and to subscribe to interesting data, and process that data as necessary.

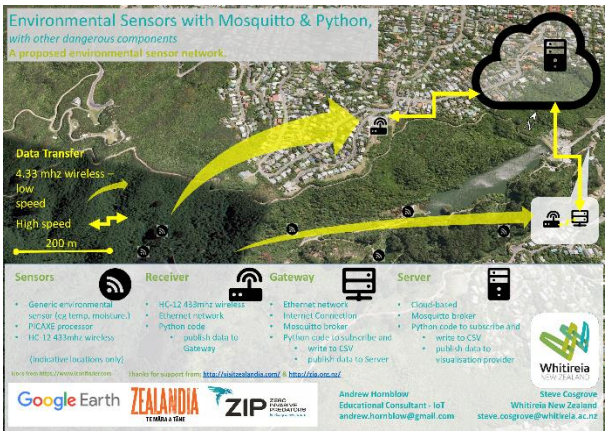
Keywords: IoT MQTT, message broker, sensor, networks, Python, Mosquitto,

1. INTRODUCTION

The Internet of Things (IoT) is a phrase becoming common in the ICT environment. Mainstream media has talked about a world full of sensors for many decades – such as the car KITT in 1982, which appeared to be able to sense anything within a kilometre or so (http://www.imdb.com/title/tt0083437/). A recent commentator looking at the Consumer Electronics Show (CES) in Las Vegas suggests we could soon "… be surrounded by brainy things …".(Baig 2018). A common largescale sensor application is smart on-street parking management (Polycarpou Lambrinos, & Protopapadakis, 2013). This technology is used in Wellington city central business district.

In the environmental monitoring field, manual recording and storing was traditionally the primary data handling strategy (Wilson, Gavin J. and Delahay, Richard J. 2001). During the 21st Century, automated sensors have become more accessible, with lower prices, and enabling technologies meaning they can be easily deployed with minimal resources.

This paper proposes a wireless sensor network infrastructure for use in a bush environment, where there is little supporting infrastructure.



This poster appeared at the 9th annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ2018) and the 31st Annual Conference of the National Advisory Committee on Computing Qualifications, Wellington, New Zealand, on July 11-13, 2018 as part of ITx 2018.

2. THE ENVIRONMENT

This research will be based in an area predominantly made up of New Zealand native bush, with a variety of characteristics. There will be a typical range of bush density, topography, and water between wireless sensors and receivers..

Over a period of up to a year, variable environmental factors will be monitored. Those factors will include characteristics such as rainfall, humidity, or barometric pressure.

Two receivers will be used, and up to six sensor devices. Each device can contain multiple individual sensors. The devices will be in positions between 100 metres and 1,500 metres from the receivers.

3. HARDWARE

This sensor network will be operating in an environment that requires broad resilience, well being a model for keeping cost down.

3.1 Processors

Sensor devices will be based around a PICAXE processor. These processors offer a flexible architecture, including a range of analogue to digital converters (ADCs), I/O ports and serial communication ports (Hackett, 2010).

Raspberry Pi (Pi) computers will be used as Receivers and a Gateway computer, which will be part of the network. These are single board computers, based on ARM processors (RaspberryPI models comparison, n.d.).

A regular enterprise computer will be used as a Server.

3.2 Sensors

A variety of sensors will be used in this project. An example is the DS18B20 digital temperature probe which can be purchased for under $5.00, and can can provide temperature readings in the range of -55°C to +125°C in 9-bit to 12-bit accuracy. Other low-cost sensors include the BMP180 12C Barometric Pressure Sensor, and SI7021 I2C Humidity Sensor (http://www.phoenix-tech.co.nz/nz-en/index.php/shop/sensors).

Research such as Markham, Trigoni, & Ellwood, (2010) suggests that moisture levels in the bush will be a significant predictor of reliability of radio frequency (RF) performance. To measure bush level moisture levels, one of the authors has developed the 'Manukameter'. This is piece of native timber with metal terminal is placed at each end. The PICAXE

processor can measure resistance between the terminals, from which moisture levels can be determined. The device can be calibrated to calibrated using ASNZS1080.1-2012.

4. DATA NETWORKS

Two types of network connections are used. The sensors have very limited resources (including speed, data bus, programming tools), so use very simple infrastructure. While the Pi has fewer resources than an enterprise workstation or server, it does has an implementation of the Python development environment and initial indications are that a Pi can process data received from six sensors in real time, with potential to add more.

4.1 Low Frequency wireless

Communication between sensors and receivers will use the HC-12 serial radio frequency (RF) module operating at 433Mhz (HC-12 Wireless Serial Port Communication Module, 2012). The serial interface on these modules can communicate with the serial port on either a PICAXE processor or a Pi computer.

The data protocol used at this level uses a very simple, low overhead, structure, designed specifically for this situation.

4.2 Ethernet

Connection between the Pi computers will use commodity wired and wireless Ethernet networks. Discussion of these connections is beyond the scope of this poster.

5. MQTT PROTOCOL

The Message Queuing Telemetry Transport (MQTT) is used to move and process data on the Pi computers.

This protocol is simple to use, light weight in terms of data use, open and simple to implement (Banks & Gupta, 2014).

5.1 Mosquitto Broker

MQTT uses a publish/subscribe model. Mosquitto provides services to enable this functionality (Eclipse, 2018). Software can be developed to: i) send (publish) messages to the Mosquitto server, ii) receive messages from the server by subscribing to interesting message topics.

In this project, a Mosquitto MQTT Bridge is used. All messages published to a Gateway are automatically published to the Server.

5.2 Cloud-based Visualisation

The authors have found that the MQTT protocol is particularly suited to real-time visualisation of IoT data. This has benefits in an educational setting. For this project, data will initially be available on the myDevices (2017) platform.

6. SOFTWARE

The hardware and key transport protocol has been described. This project uses a proprietary operating system (OS) included in the PICAXE firmware. The Pi computers use the Raspbian Linux distribution (Raspbian, n.d.).

6.1 PICAXE BASIC

An implementation of the BASIC language, with command particularly chosen for this processor, is used to read sensor hardware, add control bytes, and send to the HC-12 for transmission (Hackett, 2010).

6.2 Python

A low overhead, interpreted language, suited for running on the Pi for low-bandwidth applications (Monk, 2015), Python is used for message processing.

- Receivers run code to read the Pi serial port, and publish incoming data to a Gateway.
- Gateways run Python code that subscribes to messages received, and writes messages to a CSV file for backup.
- Server receives all messages through use of a Mosquitto Bridge. It runs two subscription services: One writes a long-term record of messages to a CSV file; The other service forwards messages to the visualisation cloud provider.

7. CONCLUSION

This paper has introduced plans for using low cost sensor and processor hardware to send sensor data using the MQTT protocol and Mosquitto broker.

At the time of writing, a software model of the deployment has been developed, and various sensor devices are being tried. The next step will be a physical deployment. That will be the basis of further publications of both the hardware performance, and the result of analysis of data collected.

8. REFERENCES

Baig, E. (2018). CES 2018: Get ready for yet more (yes more) smart devices. Retrieved February 13, 2018, from https://www.usatoday.com/story/tech/columnist/baig/2018/01/05/ces-2018-get-ready-yet-more-yes-more-smart-devices/999423001/

Banks, A., & Gupta, R. (Eds.). (2014, October 29). MQTT Version 3.1.1. OASIS Standard. Retrieved from http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

Eclipse. (2018, January 8). Eclipse Mosquitto. Retrieved February 14, 2018, from https://mosquitto.org/

Hackett, R. (2010). PICAXE Microcontroller Projects for the Evil Genius. McGraw Hill Professional.

HC-12 Wireless Serial Port Communication Module. (2012, October). Retrieved from https://www.elecrow.com/download/HC-12.pdf

Markham, A., Trigoni, N., & Ellwood, S. (2010). Effect of rainfall on link quality in an outdoor forest deployment. In 2010 International Conference on Wireless Information Networks and Systems (WINSYS) (pp. 1–6).

Monk, S. (2015). Programming the Raspberry Pi, Second Edition: Getting Started with Python. McGraw Hill Professional.

myDevices. (2017). myDevices - About. Retrieved February 14, 2018, from https://mydevices.com/about/

Polycarpou, E., Lambrinos, L., & Protopapadakis, E. (2013). Smart parking solutions for urban areas. In 2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM) (pp. 1–6). https://doi.org/10.1109/WoWMoM.2013.6583499

*RaspberryPI models comparison. (n.d.). Retrieved February 14, 2018, from http://socialcompare.com/en/comparison/raspberrypi-models-comparison

Raspbian. (n.d.). RaspbianAbout - Raspbian. Retrieved February 14, 2018, from https://www.raspbian.org/RaspbianAbout

Wilson, G. J., & Delahay, R. J. (2001). A review of methods to estimate the abundance of terrestrial carnivores using field signs and observation. Wildlife Research, 28(2), 151. https://doi.org/10.1071/WR00033