# Sensor node for monitoring environmental factors in industrial installations

Lauro Manoel Lima da Gama
Universidade do Estado do Amazonas
Manaus, Amazonas
Email: lauro.gama@gmail.com

Msc, Almir Kimura Junior
Universidade do Estado do Amazonas
Manaus, Amazonas
Email: akimurajr@gmail.com

***Abstract*—This paper proposes a sensor node for measuring environmental factors on industry surroundings. Industrial environment is regulated in varying degrees across the world by many enviromental and safety policies. The adherence of these policies makes necessary a continuous andd distributed measurement of different factors like temperature, noise levels and the presence of contaminants. The analysis of this data may reveal violations in work conditions and enviromental damage. The proposal of a low cost, easy interface and customizable sensor module to be used in network environment will help its use.**

## I. Introduction

The dynamic nature of industries present a challenge to wired networks and the pressures of competivity and cost efficiency apply many constraints to development.

Wireless sensor networks have advantages over wired networks because the sensor nodes are may be installed without changes in existent infrastructure by a relative small price[1].

July 1, 2015

### A. Sensor network

In recent years, wireless sensor networks (WSNs) have gained worldwide attention for use in different applications. Sensor nodes are spatially distributed across a large area of interest to sense, measure, and gather information and transmit the data to the user. The nodes are typically equipped with radio transceivers, micro-controllers, and batteries. These sensor nodes are in general equipped with one or more sensors (e.g. mechanical, thermal, biological)[2]. They are small in size, inexpensive, and could be deployed in large numbers. They can be used in applications such as military target tracking and surveillance, natural disaster relief, biomedical health monitoring , and industrial automation.[3] Each of these nodes can sense, measure, and gather information from the environment and, based on some local decision process, they can transmit the sensed data to the user [2].

### B. Design and implementation

*1) Network topology:* The target for the proposed network are indoor buildings with air conditioning and artificial lightning. These types of building normally range to up to hundred of meters of built area with many partitions between the areas. The nodes dont need to communicate between themselves, only with the server. Short-range wireless technologies such as IEEE 802.15.4 in mesh network configuration are widely considered to be cost-effective solution for use in industrial settings[3]. These characteristics make the star network topology the most adequate to be implemented.

In this topology each sensor node connects directly with the central hub, sending and receiving information. This communication is made over WIFI where each node connects with the hub and receives an IP address. Each node has an internal identification in hash format that is sent with each message. The message payload from each sensor may change according with the hardware so the message format must be flexible.

*2) Sensor Node:* The sensor nodes are raspberry pi 2 model B [4] running a Raspbian[5] distribution composed of a microcontroller receiving signals from sensors. Each signal must be converted to an appropriated scale and packaged in a message to the Central Hub. The following subjects are being monitored: Temperature, humidity, noise, luminance, and flamable gas presence.

The Sensor node must be fault resilient and able to restart in case of failure. The sensor node has a program called *”client”* that is responsible for the data collection and transmission.

The client persists data from the sensors in a SQLite[6] database and transmits this information to the central hub.

A Lipo battery is used to provide energy backup to the node in case of power shortage.

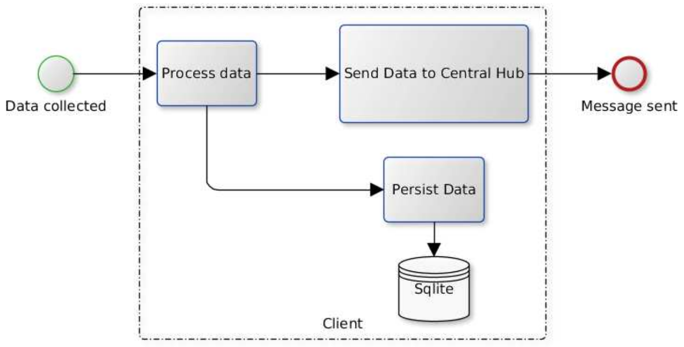

Fig. 1. Sensor Node Diagram

*3) Central Hub:* The central hub must be able to process a high volume of messages coming from the sensor nodes and to answer requisitions for information from network users.

The central hub was implemented also using raspberry pi 2 model B running a computer program based on Tornado[7], a RabbitMQ broker and a webserver based on the framework flask[8].

Tornado is a Python web framework and asynchronous networking library. By using non-blocking network I/O, Tornado can scale to tens of thousands of open connections, making it ideal for long polling, WebSockets, and other applications that require a long-lived connection to each user.[7]

The server is responsible for listening for messages posted on the RabbitMQ broker and collect the relevant data and persist them in its own SQLite database. This database has information from all nodes composing the network.

The RabbitMQ broker is responsible for managing communications between nodes and the central hub.

The webserver is an application with an REST API[9] responsible for providing users of the system with an method to visualize the data collected by the sensor network. It uses a lightweight server structure ideal for running in embedded systems.

The services implemented in the central hub may be divided between data collection and communication.

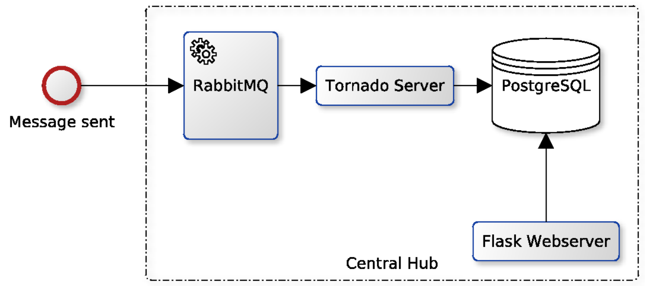

Fig. 2. Central Hub Diagram

*4) Message Protocol:* The messages are sent from the sensor node to the central Hub using the MQTT protocol.

MQTT stands for MQ Telemetry Transport. It is a publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimise network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles also turn out to make the protocol ideal of the emerging machine-to-machine (M2M) or Internet of Things world of connected devices, and for mobile applications where bandwidth and battery power are at a premium[10].

MQTT is a publish/subscribe messaging system that allows clients to publish messages without concerning themselves about their eventual destination; messages are sent to an MQTT broker where they may be retained.

The messages' payloads are just a sequence of bytes, up to 256MB, with no requirements placed on the format of those payloads and with the MQTT protocol usually adding a fixed header of two bytes to most messages.

Other clients can subscribe to these messages and get updated by the broker when new messages arrive. To allow for the variety of possible situations where MQTT can be put to use, it lets clients and brokers set a ”Quality of Service” on a per-message basis from ”fire and forget” to ”confirmed delivery”. MQTT also has a very light API, with all of five protocol methods, making it easy to learn and recall, but there's also support for SSL-encrypted connections and username/password authentication for clients to brokers.

The broker utilized in the implementation was RabbitMQ through the python package Pika. Pika is a pure-Python implementation of the AMQP 0-9-1 protocol that tries to stay fairly independent of the underlying network support library[11].

*5) Message Format:* The message payload is formatted using in the Json format. JSON (JavaScript Object Notation) is a lightweight data-interchange format[12].

```json
{
   "timestamp": "2015-07-16 15:40:49.218438",
   "data": {
      "ilumination": 210,
      "noise": 123,
      "carbon_monoxide": 170,
      "temperature": 51,
      "humidity": 127
   },
   "id": 307
}
```

## II. Conclusion

In this article an Architecture for Wireless sensor networks for monitoring Environmental factors is presented. The model takes into account the need for constant data analysis and transmission of high volumes of messages in a heavy electronic noise environment. The presented architecture is able to be scaled to a high number of nodes.

## Acknowledgment

The authors would like to thank their families for their love, care, kindness, patience, and unfailing trust.

## References

[1] Pavithra L and A Angeline, “Survey on Industrial Wireless Sensor Networks,” pp. 3166–3171, 2015.
[2] R. F. d. S. Ribeiro, “Urubu : energy scavenging in wireless sensor networks,” Ph.D. dissertation, Polytechnic Institute of Porto School of Engineering, 2010.
[3] M. Cheffena, “Industrial wireless sensor networks : channel modeling and performance evaluation,” pp. 12–14, 2012. [Online]. Available: http://jwcn.eurasipjournals.com/content/pdf/1687-1499-2012-297.pdf
[4] R. P. Foundation. (2015) What is raspberry pi? [Online]. Available: https://www.raspberrypi.org/help/what-is-a-raspberry-pi/
[5] Raspbian. (2015) Raspbian.org. [Online]. Available: https://www.raspbian.org/
[6] . (2015) Sqlite. [Online]. Available: https://www.sqlite.org/
[7] ——. (2015) Tornado. [Online]. Available: http://www.tornadoweb.org/en/stable/
[8] (2015) A python microframework. [Online]. Available: http://flask.pocoo.org/
[9] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” *Building*, vol. 54, p. 162, 2000. [Online]. Available: http://www.ics.uci.edu/ fielding/pubs/dissertation/top.htm
[10] (2015) What is mqtt. [Online]. Available: http://mqqt.org

[11] (2015) Introduction to pika. [Online]. Available: https://pika.readthedocs.org
[12] (2015) Definition of json. [Online]. Available: http://www.json.org