Universidad del Azuay

Facultad de Ciencias de la Administración

Escuela de Ingeniería de Sistemas y Telemática

IMPLEMENTACIÓN, CALIBRACIÓN Y EVALUACIÓN DE UN SISTEMA DE MEDICIÓN DE RUIDO CON UN SENSOR DE SONIDO QUE PERMITA LA TRASMISIÓN INALÁMBRICA DE DATOS

Trabajo de graduación previo a la obtención del título de Ingeniero de Sistemas y Telemática

Autor:
David Gerardo Mejía Saca

Director:
Ing. Gabriel Barros Gavilanes PhD.

Cuenca – Ecuador
2018

DEDICATORIA

A Dios.

Por haberme brindando la sabiduría suficiente que me permitieron tomar las decisiones necesarias para cumplir las diferentes metas y sueños propuesto en esta etapa de mi vida.

A mis Padres

Por darme su apoyo y confianza llenándome de valores necesarios, lo que me ayudado a cumplir con todas las metas propuestas.

A mis Amigos

Quienes me han brindado buenos y malos momentos, lo que me ha permito tener la audacia necesaria para sobrevivir y disfrutar cada uno de los ciclos que compartimos juntos.

AGRADECIMIENTO

A la Universidad del Azuay por haberme brindado una beca de estudios.

Al Ingeniero Juan Gabriel Barros por haber confiado en mí y brindarme el apoyo suficiente para desarrollar el tema de tesis.

Al Ingeniero Diego Chacón por inculcar sueños, metas y educarnos en el estudio de nuevas tecnológicas.

Al Instituto IERSE que me brindo el apoyo necesario para realizar las pruebas necesarias que permitieron avalar el sistema propuesto.

RESUMEN:

El ruido es un contaminante que puede producir daños a la salud tanto a nivel psicológico y físico. Las campañas de medición de ruido son costosas, por necesitar sonómetros certificados e ingentes recursos humanos. Actualmente existen sistemas automatizados con trasmisión inalámbrica, para tener mediciones más frecuentes ahorrando recursos. Este artículo explica la implementación de un sistema de medición de ruido de bajo costo, que contiene un sensor de sonido, con recolección y calibración. Los datos son transmitidos en el menor tiempo posible de manera continua a través de una red WSN (IEEE 802.15.4) hasta la pasarela. La información es almacenada en el dispositivo o en la nube. El sistema alcanza una precisión de un dispositivo Clase 3.

Palabras clave: ruido, sistemas automatizados. WSN, IEEE 802.15.4,raspberry,solar,

ABSTRACT

Noise is a contaminant that can cause damage to health at a psychological and physical level. Noise measurement campaigns are expensive because they require certified sound meters and huge human resources. Currently, there are automated systems with wireless transmission to have more frequent measurements and save resources. This article explained the implementation of a low cost noise measurement system that contained a sound sensor with collection and calibration. The data was continuously transmitted to the gateway in the shortest possible time through a WSN network (IEEE 802.15.4). The information was stored on the device or in the cloud. The system reached the precision of a Class 3 device.

UNIVERSIDAD DEL AZUAY
Dpto. Idiomas

Translated by
Ing. Paul Arpi

ÍNDICE

Contenido

1. INTRODUCCIÓN

En la actualidad, el ruido se ha convertido en un contaminante auditivo que causa daños en la salud de los seres humanos (Sevillano et al., 2016). Ejemplos de afectaciones a la salud son la progresiva pérdida de la audición, alteraciones en la presión arterial, ritmo cardiaco, insomnio, cefaleas crónicas, y reducción de la capacidad sexual (Alonso, 2003). Entre los aspectos psicológicos constan el aumento del estrés e irritabilidad. Es por eso que recomienda como límite máximo un nivel sonoro de 80dB (Sanz & García, 2003).

Una campaña de monitoreo implica poseer: los equipos necesarios para realizar las mediciones y registrarlas (hardware) además de los recursos para contratar personas durante el tiempo de duración de la campaña. Asumiendo un salario básico de 400 USD por 160 horas mensuales, obtenemos un valor de 2.5 USD por persona por hora. Una campaña típica (EMOV EP, Inventario de emisiones atmosféricas, 2014, p 21): 2 horas (5 mediciones de 15 minutos), con 15 puntos de muestreo en la ciudad; resulta en 75 USD por día de medición. Uno de los intereses actuales es prolongar los tiempos de las campañas de monitoreo y de ser posible hacerlas continuas. De allí la importancia de las plataformas de medición continúa del ruido.

Para realizar el monitoreo adecuado de este contaminante mediante una plataforma, se debe considerar aspectos como: hardware, costos, recursos humanos, crecimiento de la red (escalabilidad), adaptación del sistema con otras tecnologías (flexibilidad), calidad de los datos (confiabilidad) y margen de error en los datos (precisión) (Alfie, 2017.), además del costo del recurso humano. Los equipos utilizados para medir el sonido son los sonómetros. Estos miden el nivel de presión sonora de un ruido en dB (Moreno, 2012), en tiempo y espacio. Los sonómetros deben cumplir con las normas IEC (International Electrotechniacal Commision) 61672. En la norma IEC 61672-1 se especifican 2 clases. La clase 1 se utiliza en laboratorios o en lugares donde el ambiente acústico pueda ser controlado, mientras clase 2 se utiliza para mediciones generales (Caguano, 2016.).

Sin embargo, la inversión necesaria para implementar toda esta infraestructura, de manera que permita obtener los mejores resultados, es elevada, pudiendo llegar a los 1500 USD por estación de muestreo. Esto ha conllevado a buscar tecnologías más económicas que sustituyan a los dispositivos de monitoreo (Wessels & Basten, 2016). Así nace la

iniciativa de generar estaciones de monitoreo a un costo reducido concentrando la investigación a la implementación de una clase 2 o 3.

La generación de sensores de bajo costo ha creado un interés en el desarrollo de redes WSN (redes de sensores inalámbricas). Entre los protocolos más utilizados están: Bluetooth (sobre IEEE 802.15.1), UWB (sobre IEEE802.15.3), ZigBee (sobre IEEE 802.15.4) y Wi-Fi (sobre IEEE 802.11a) (Sánchez-Rosario et al., 2015). Chang et al., (2015) en parte de su investigación enuncia que el protocolo ZigBee es el más valorado por la industria, ya que siendo de bajo costo e incurriendo en un bajo consumo de energía, puede tener múltiples características en velocidad y distancia

Los estudios acerca de la contaminación auditiva de bajo costo han comenzado a utilizar Smartphones, como los sensores que permiten la obtención del sonido. Estos con los algoritmos de calibración han llegado a obtener resultados satisfactorios que han permitido obtener una idea general de cómo están los niveles de ruido en los lugares en donde fueron tomadas las muestras. Cabe recalcar que estos resultados se encuentran con un error de 3 a 4 dB, dependiendo del método de calibración (Barros, 2015). Sin embargo, esta magnitud de error es todavía demasiado alta, planteando la necesidad de investigar maneras de reducir dicho error.

En este contexto, este trabajo propone diseñar un sistema de medición de ruido, que con técnicas de calibración den resultados satisfactorios para el análisis de la contaminación auditiva. Los datos serán enviados a la nube por medio de una infraestructura de red (WSN) sencilla de levantar.

2. MÉTODO

2.1 Propuesta Tecnológica

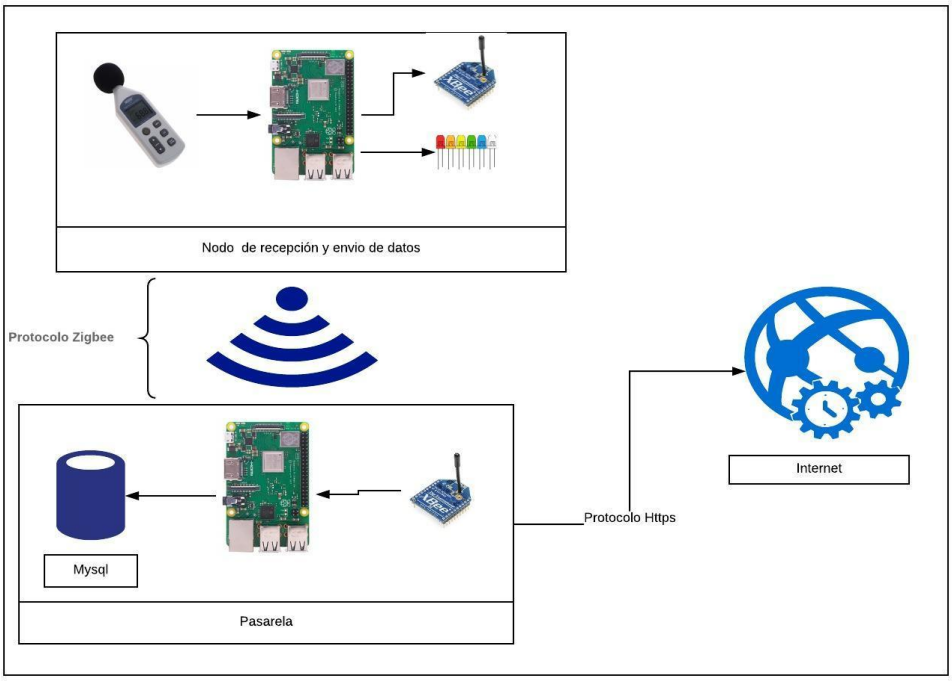
Consiste en la implementación de un prototipo compuesto por un sensor de sonido que permita medir los niveles del ruido, una placa de procesamiento y una antena XBee para el nodo de recepción, mientras que para la pasarela se necesita una placa de procesamiento y una antena XBee, la combinación de estos dos prototipos permite tener un sistema de medición de ruido que permitan obtener datos de calidad deseando obtener al menos un dispositivo con una calidad de medición equivalente a un sensor de clase 3, además de tener la capacidad de transmitir los datos inalámbricamente y su posterior almacenamiento en un servidor web.

2.2 Descripción del sistema para el análisis del ruido

El objetivo del sistema es utilizar tecnologías de bajo costo que permitan analizar los niveles del ruido lo que minimizará el recurso humano al momento de realizar la toma de muestras, por lo tanto, los costos se reducirán. La definición que se escoge para bajo costo tiene relación con el costo total del dispositivo. Es decir, que si el prototipo generado tiene un costo menor a un equipo especializado, la implementación se considera de bajo costo. Además, el sistema de medición podrá alcanzar un tiempo de muestro mayor al conseguido por una persona. El hecho de que la información sea transmitida inalámbricamente genera seguridad en el almacenamiento ya que los datos no son manipulados por terceros.

El sistema está compuesto por un módulo de sonido que nos permitirá obtener los niveles de ruido en el ambiente, un Raspberry Pi 3 (RPi3) el cual contará con un Sistema Operativo (SO) que permitirá procesar, calibrar, almacenar y generar una trama que se enviara mediante un Xbee Digi Mesh (modelo V1) a la pasarela de procesamiento. Ver Figura 1.

Figura 1
Esquema del Funcionamiento general del sistema

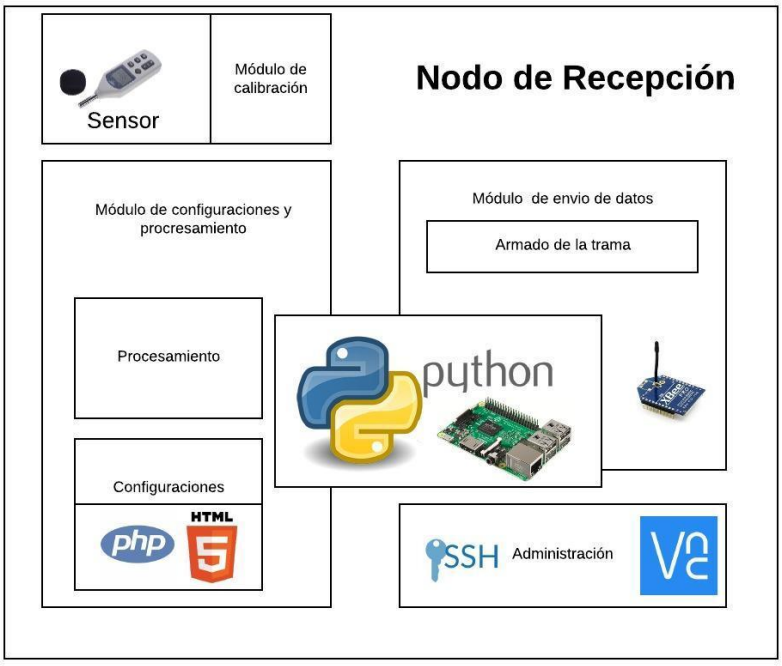


Estructura del sistema, el cual cuenta con un nodo, una pasarela y la comunicación hacia la web.

2.3 Nodo de Recepción

Arquitectura del nodo de recepción. Ver Figura 2.

Figura 2
Nodo de Recepción



Arquitectura del Nodo de Recepción.

2.3.1 Sensores de Sonido

En primera instancia se utilizó un sensor de sonido modelo KY-038 el cual cuenta con las siguientes características definidas por el fabricante (Velleman, 2017). Ver Tabla 1.

Tabla 1
Características sensor KY-038

Parámetros	Característica
Vcc	5V
A0	Señal Analógica
Distancia de inducción	0.5 metros
Gama de frecuencias	100 ~ 10.000 Hz
Sensibilidad mínima	58dB

Características principales del sensor KY-038 que se tienen que tener a consideración para su correcto funcionamiento.

El sensor KY-038 se va a comparar con otros dispositivos de medición de ruido, esto permitirá definir el mejor dispositivo que se utilizará en el nodo de recepción. Ver Figura 3.

Figura 3
Gráfico KY-038

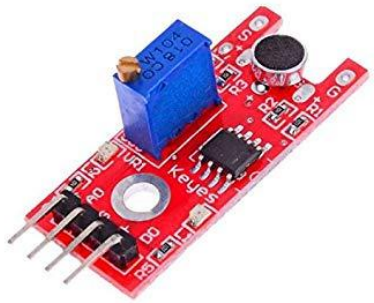


Gráfico del sensor KY-038

En segunda instancia se utilizó un sonómetro Wens modelo 1361el cual cuenta con un sensor de sonido y una placa que permite la amplificación y eliminación del ruido de la señal, esto permite tener muestras más exactas y con un mayor rango de sensibilidad, este cuenta con las siguientes características definidas por el fabricante (Wens, 2018). Ver Tabla 2.

Tabla 2
Características sonómetro 1361

Parámetros	Característica

Vcc	5V
AC	Señal Analógica
DC	Señal Analógica C
Sensibilidad	30-130 dB
Gama de frecuencias	3.5 Hz ~ 8.5khz
Ponderación	a/c

Características principales sonómetro Wens 1361 que se tienen que tener a consideración para su correcto funcionamiento.

El sonómetro Wens 1361 es un dispositivo más complejo por contar con un sensor de sonido, y una placa electrónica que permite obtener datos más exactos, ya que su fabricante indica que permite cumplir la norma IEC651 Tipo 2, además nos brinda la posibilidad de cambiar la frecuencia de ponderación A y C (Pfretzschner, 2002) y contar con los respectivos puertos analógicos. Ver Figura 4.

Figura 4
Gráfico del Wens 1361

Gráfico del sonómetro Wens 1361.

2.3.2 Sistema de Procesamiento

En primera instancia se utilizó una placa Arduino Uno, la misma que permite reconocer y procesar la señal de los diferentes dispositivos. Se utilizó la salida analógica del sensor de sonido KY-038 para una conversión analógica digital por parte de la placa de procesamiento. La programación de estas tareas se puede visualizar en el siguiente diagrama de flujo. Ver Figura 5.

Figura 5
Lectura Sensor-Sonido

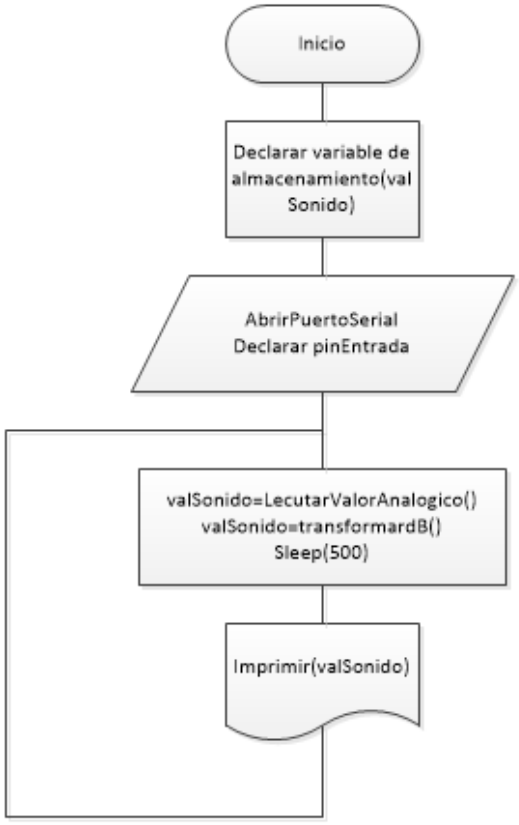


Diagrama de flujo que nos permite identificar el funcionamiento y procesamiento de la señal del sensor de sonido.

La placa Arduino Uno tiene la capacidad de medir tensiones de entrada entre 0 y 5 voltios los mismos que se encuentra ponderados entre 0 y 1023 obteniendo como resultado la fórmula 1 (Solórzano, 2016).

$$r = \frac{5V}{1023} = 4.9mV \quad (1)$$

El sensor de sonido KY-038 envía una señal entre 0 y 1023, según las características del fabricante esta señal tiene que ser convertida a voltios y después a dB, para realizar esta conversión utilizamos la fórmula 2.

$$V = señalSensor * \frac{5}{1023} \quad (2)$$

2.3.3 Validación de sensores y placa de procesamiento

Se realizaron tres pruebas, para identificar los problemas y beneficios de utilizar cada sensor. Ver Tabla 3.

Tabla 3
Pruebas de validación de Sensores

Número	Descripción	Elementos

Prueba1	Comparativa del sensor KY-038, Teléfono P9 Lite con el sonómetro Wens 1361	Sensor Ky-038 Teléfono p9Lite Sonómetro Wens 1361 Arduino Uno Interfaz Serial
Prueba2	Obtención de datos con las salidas analógicas del sonómetro Wens, para validar con los datos presentados en la pantalla LCD del sonómetro.	Sonómetro Wens 1361 Salida analógica DC Arduino Uno Interfaz Serial
Prueba3	Obtención de datos con la salida USB del sonómetro Wens, para validar con los datos presentados en la pantalla LCD del sonómetro.	Sonómetro Wens 1361 Salida USB (Sonómetro) RPi3

Pruebas que se realizaron para validar los datos obtenidos por los sensores que se están comparando.

La prueba 1 consiste en colocar el sensor KY-038, el sonómetro Wens y un teléfono Huawei P9 Lite a una distancia de 1 metro de una fuente de sonido constante durante un periodo ininterrumpido de 10 minutos, con la finalidad de obtener resultados acerca de la diferencia de calidad de los datos. Los mismos que son presentados en la sección de resultados del nodo de recepción.

Una segunda prueba en la que se utilizó únicamente el sonómetro Wens con la finalidad de comparar los resultados obtenidos por la salida analógica, para procesar los datos de la salida analógica se necesitaron fórmulas que permiten transformar los Voltios a dB, esta conversión se realiza en la placa Arduino Uno con la fórmula 3.

$$r = \frac{5V * señalSensor * 100}{1024} \qquad (3)$$

En la tercera prueba se utilizó la salida USB del sonómetro y para el procesamiento de la señal se remplazó la placa Arduino UNO por el RPi3, ya que esta última cuenta con 4 interfaces USB y 40 pines GPIO los mismos que serán utilizados para emitir las señales de error a un par de ledes.

Esta placa es más potente ya que permite la instalación de un SO, en este caso Raspbian y por lo tanto la utilización del lenguaje de programación Python, (Prasad, 2014), esto facilita la instalación y manipulación de los distintos paquetes que permiten el manejo de todas las herramientas que tiene la placa. Ver figura 6.

Figura 6
Placa RPi3
RPi3

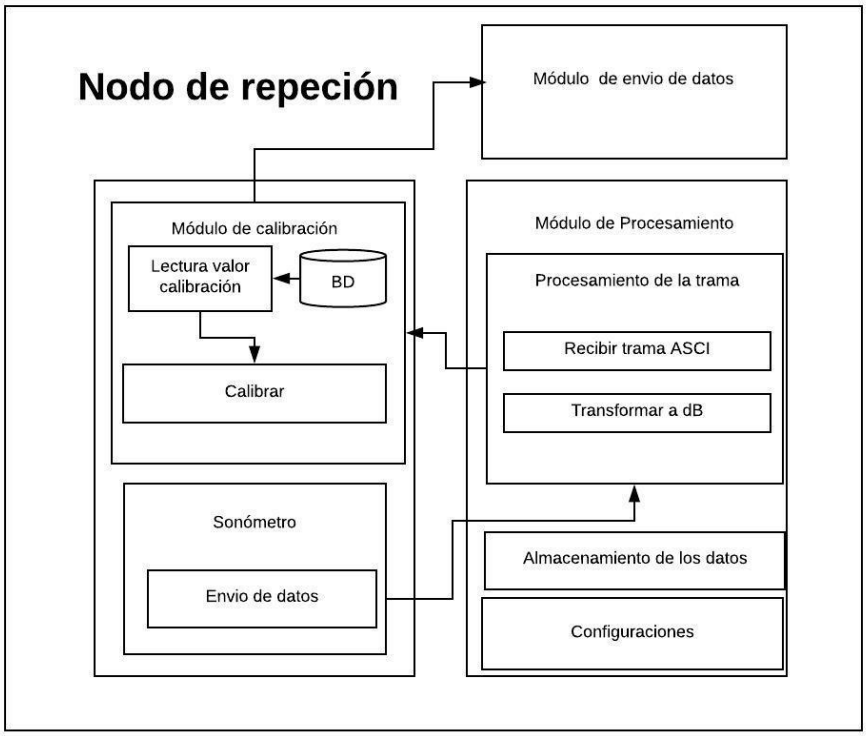
Placa RPi3 modelo b

Para poder utilizar el RPi3 se tiene que comenzar con el proceso de instalación del Sistema Operativo (SO), existe una variedad de SO que soporta la tarjeta, esta elección dependerá de las necesidades y el uso que se la quiera dar.

Para esta propuesta se escogió el SO Raspbian, por ser una distribución de Debían, ya que gracias a su amplia documentación permite una fácil instalación y configuración además de un mejor manejo de paquetes y librerías necesarias para el buen funcionamiento del sistema que se desea implementar. Para poder utilizar la interfaz USB del RPi3 se procedió con la instalación de las librerías necesarias que permitan la manipulación de este puerto, tanto estas configuraciones como la instalación se encuentra de una manera más detallada en el manual Nodo de Recepción (Anexo 1).

La arquitectura que permite la obtención de los datos que son receptados por el sonómetro son presentadas en la figura 7, para esta obtención de estos se realizó con el lenguaje de programación Python ya que permite tener un mejor procesamiento. Ver Figura 7.

Figura 7
Obtención de datos



Arquitectura del funcionamiento del sistema para la obtención de datos

En el RPi3 se implementa un software realizado en Python que se adjunta en el manual nodo de recepción (Anexo 1), este permite realizar la conexión y obtener los datos que son enviados por el USB del sonómetro Wens. Los datos percibidos por el USB se tienen que decodificar para obtener el valor en claro percibido por el sonómetro.

2.3.4 Modulo de configuración y procesamiento

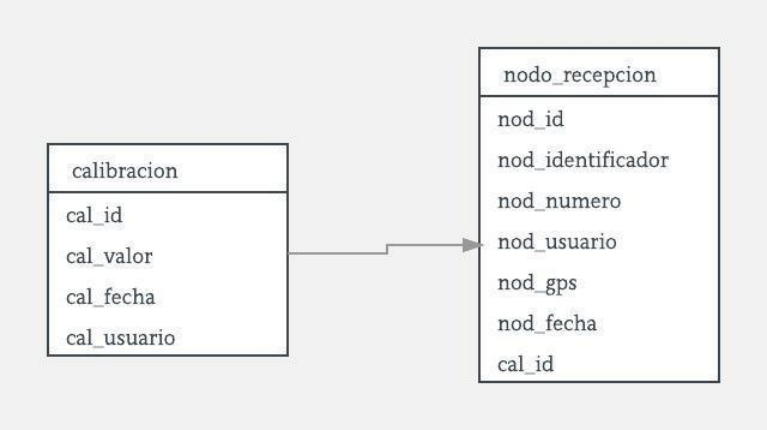
El módulo de configuraciones es un sistema web que permite al usuario gestionar diferentes opciones que tiene el nodo de recepción de datos.

2.3.4.1 Configuración

El servidor web instalado es Apache, este permite ejecutar una página web que fue desarrollada en PHP 7 y HTML5. La codificación en PHP permite tener una página que pueda interactuar con la base de datos.

Para las configuraciones generales del sonómetro se instaló la base de datos MySQL. El diseño de la base de datos está enfocado en una estructura simple que permita almacenar los datos de configuraciones generales. Ver Figura 8.

Figura 8
Estructura de la base de datos



Diseño de la base de datos del nodo.

El diseño de la Base de Datos está compuesto por varios campos necesarios para el buen funcionamiento del sistema. Ver Tabla 4.

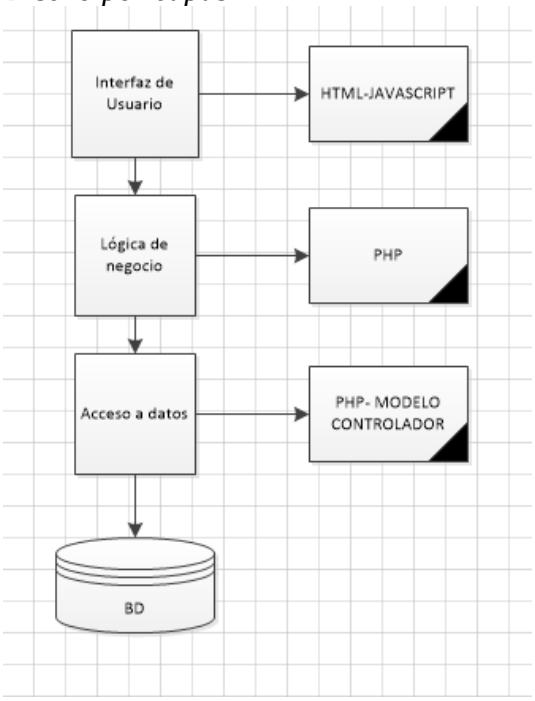
Tabla 4
Base de datos

Campo	Descripción
cal_valor	Es el campo el cual permitirá el almacenamiento el valor de la calibración que necesite el nodo de transmisión.
nod_usuario	Usuario que realiza la toma de muestras
nod_usuario	Usuario que realiza la calibración

Descripción de los campos más importantes del nodo de transmisión.

Para mejorar la experiencia de usuario, este módulo se realizó en base al modelo de capas, el cual permite realizar una plataforma web interactiva con el usuario. Ver Figura 9.

Figura 9
Diseño por capas



Descripción del diseño de capas.

La capa de datos y conexión permite la alteración de los datos, siendo el usuario final quien por medio de las transacciones puede modificarlos.

La capa de negocio esta codificada en el lenguaje de programación PHP, será la encargada de la transaccionalidad que brinda la interfaz.

La capa de presentación es la encargada de tener todas las interfaces que fueron codificadas en el lenguaje de programación HTML5 y JavaScript además de contar con estilos realizados con CSS, como se presenta en el manual del nodo de recepción.

La interfaz Gráfica se realizó de una manera que se acople a la sencillez y facilidad de uso para el usuario final, este proceso se basó según la metodología del autor Jesse James Garrett (Garrett, 2015). De esta forma, el usuario puede acceder a la interfaz de configuraciones en el RPi3 utilizando cualquiera de las interfaces dispuestas en el nodo, por ejemplo Wi-Fi, Ethernet.

2.3.4.2 Procesamiento

Es la lógica general del software que se encarga de realizar y ejecutar todas las tareas del sistema en cuestión. Su proceso de ejecución inicia con métodos de verificación que permiten el reconocimiento de los dispositivos que forman parte del sistema.

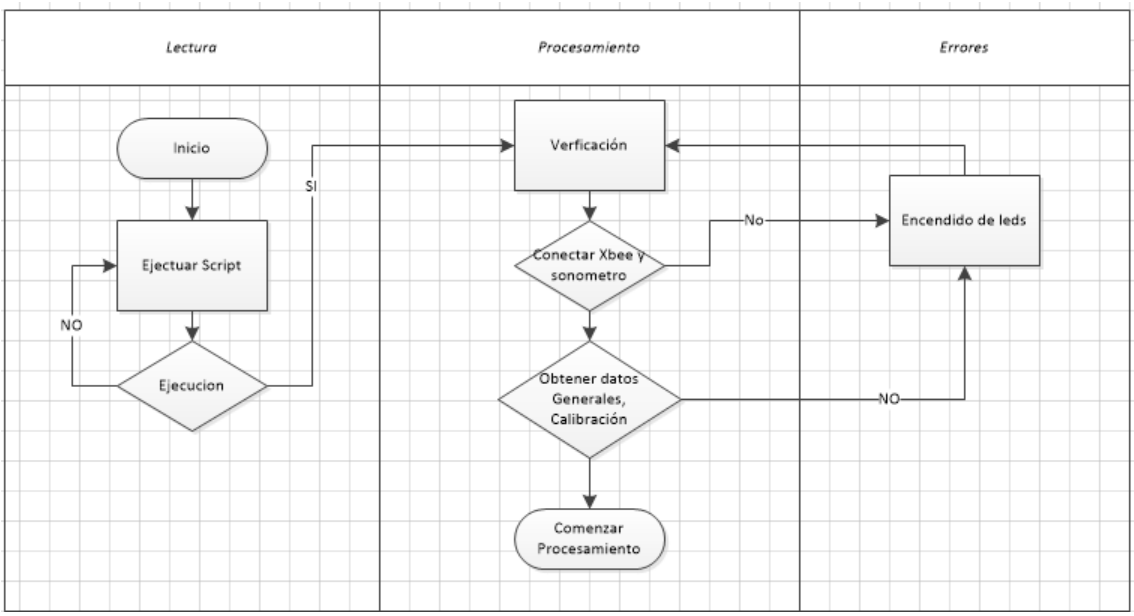
El Proceso de verificación del sistema es el encargado de reconocer que todos los dispositivos necesarios para el funcionamiento estén conectados y en óptimas condiciones, ya que

verifica que esté recibiendo datos y que se pueda enviar los mismos. Si estos dispositivos no se encuentran conectados o no pueden realizar sus procesos el sistema reconoce estos errores y envía señales de error a un par de ledes conectados al RPi3. Esto permite dar a conocer el estado del nodo de recepción al usuario que lo está usando.

Al terminar el proceso de verificación del sistema como; configuración y calibración, se procede con el envío de los datos a la pasarela del sistema, este tema se revisará más a fondo en el módulo de envío de datos en la sección 2.3.6.

Cada de uno de los procesos se pudo realizar ya que la programación del software se realizó con una programación modular que sea autoadaptable a los errores y se pueda auto configurar, esto permite tener un dispositivo de fácil uso dando el valor agregado para que cualquier persona sin demasiados conocimientos en dispositivos tecnológicos pueda operarlo de la mejor manera. Ver Figura 10.

Figura 10
Modelo de programación



Lógica de programación de todo el sistema modular

2.3.5 Módulo de calibración

2.3.5.1 Método de calibración

La calibración se realiza con un desplazamiento de la señal (offset), este desplazamiento es la diferencia de error encontrada al momento de realizar el proceso de calibración.

2.3.5.2 Proceso de calibración

Para la calibración del sonómetro se utilizó el calibrador acústico profesional que cumpla con las normas IEC 61672-1. Este calibrador nos permite encontrar el factor de error que tiene el sonómetro Wens. El calibrador genera una sola frecuencia de verificación a 1Khz y un determinado nivel acústico de 114dB (Technologies, 2012). Esta variación será corregida al momento de procesar los datos para ser enviados. Ver Figura 11.

Figura 11
Calibrador AC-300

Calibrador AC-300

2.3.5.3 Modificación del factor de calibración

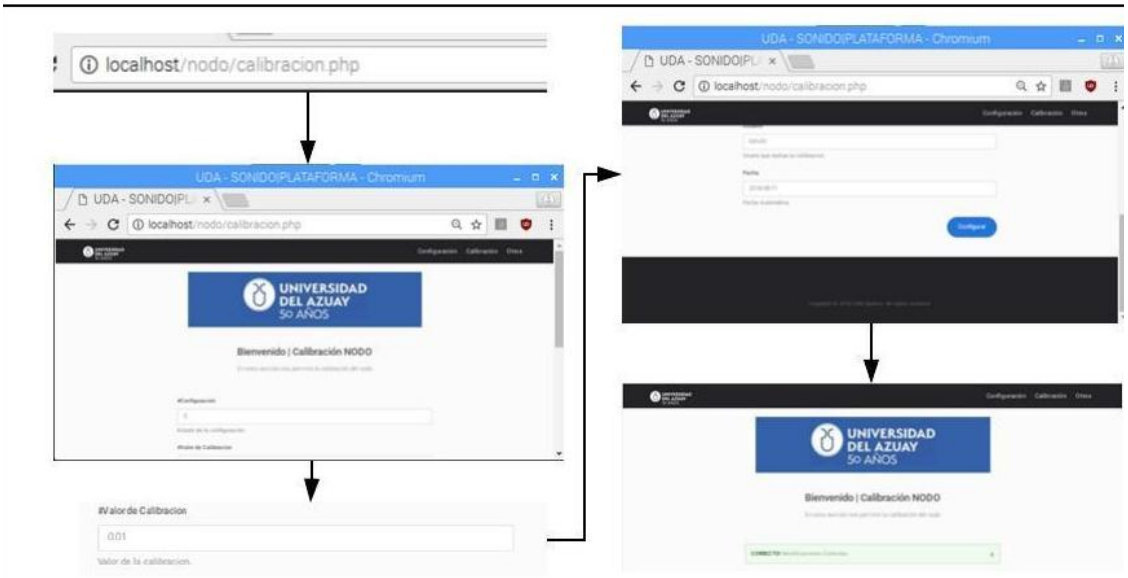
Para cambiar el factor de calibración del sonómetro, se habilitó la siguiente url: http://localhost/nodo la misma que puede ser accedida desde cualquier explorador que se encuentre habilitado en el RPi3, el proceso de ajuste de calibración se detalla continuación:

1. Encender el RPi3.
2. Abrir un Navegador.
3. Ingresar en la url: localhost/nodo.
4. Ingresar el factor de error obtenido en el proceso de calibración.
   Considerar que si el factor de error es menor se deber ingresar con un signo negativo.
5. Guardar la información.
6. Reiniciar el RPi3.

Esto permite almacenar el factor de calibración en la base de datos, al reiniciar el RPi3 estos valores son cargados al software que realiza el procesamiento. Dentro del software se implementó el método de calibración el cual tiene la capacidad de reconocer el nuevo valor e implementarlo en el software generando datos mucho más exactos. Ver Figura 12.

Figura 12

Pasos para el proceso de calibración



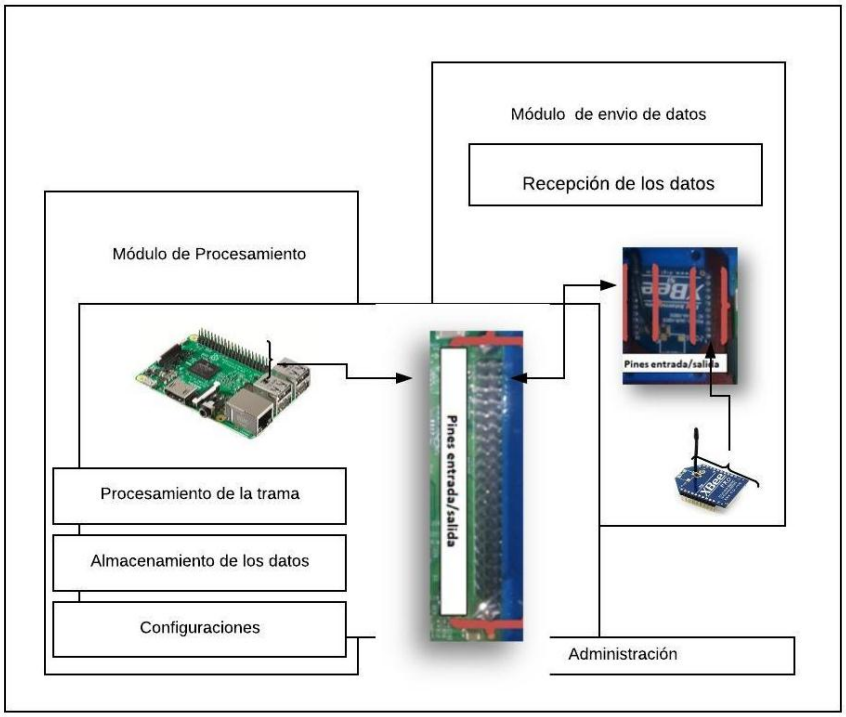
Proceso para configurar el valor de calibración del sonómetro.

2.3.6 Modulo de envío de datos

Para tener una red WSN cada nodo de recepción está incorporado con un módulo XBee S1 modelo Digi Mesh (Mesh, 2018), la topología y características generales se encuentran ampliamente descritas en el manual de la Pasarela (Anexo 2). En esta sección se definirá la configuración y procesos de instalación que permita la comunicación entre la placa RPi3 y el módulo XBee.

La comunicación entre el módulo XBee y la placa RPi3 se la puede realizar de dos maneras diferentes, la primera es una comunicación directa entre los pines de salida-entrada del XBee y los pines GPIO de la tarjeta RPi3. Ver Figura 13.

Figura 13
Xbee-RPi3



Pines de entrada y salida tanto de la placa RPi3 y del XBee.

La segunda es la que se utiliza un módulo adicional que permite la comunicación por medio de una interfaz USB, esto permite tener un mejor control del dispositivo además de poder reconocer los estados que ocurren, ya que cuenta con un micro LED que indica el buen o mal funcionamiento del módulo. Ver Figura 14.

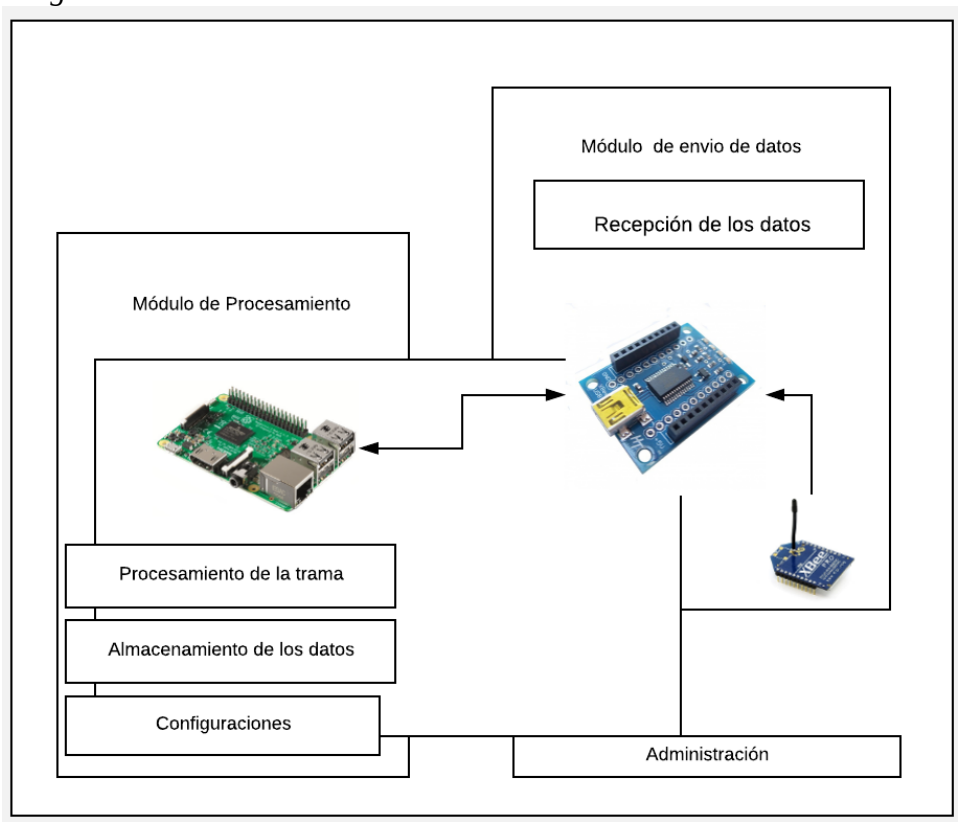
Figura 14
Módulo Explorer

Xbee y su módulo de comunicación USB.

La comunicación entre el USB *explorer* y la placa RPi3 se la realiza mediante una conexión serial, esta se la efectuó con la instalación de los paquetes y librerías necesarias lo que habilitará la comunicación entre el puerto y el USB *explorer*. Ver Figura 15.

Figura 15

Diagrama de comunicación Xbee-RPi3



Comunicación entre el RPi3-USB Explorer- Antena XBee

2.3.6.1 Armado de la trama

La trama está compuesta por tres partes:

La primera a la cual le denominaremos *cabecera*, es la encargada de tener toda la información del nodo de recepción, en esta se encuentra:

- Identificador
- Serie XBee
- Número de nodo

La segunda, consta únicamente con un valor que va desde el 1 hasta el 999. Este valor permite identificar la trama que se está enviando esto permitirá tener un control de las tramas que se llegaron a perder y no fueron almacenadas.

La tercera parte de la trama la cual denominaremos *cuerpo de datos*, es la encargada de tener todos los datos que son obtenidos en el procesamiento. Estos datos

son procesados dependiendo el sensor, por lo que tiene un identificador único, para este caso se tiene dos identificadores que son: GPS (Coordenadas ingresadas por el operador del lugar de la toma de muestras) y MPC (valor obtenido por el sonómetro). Ver Tabla 5.

Tabla 5
Composición de la trama de Datos

Trama Enviada	<=>#058#123456456465#n02#100#GPS:123.123132<-0.121321#MCP:80.599%

Ejemplo de una trama procesada y lista para ser enviada

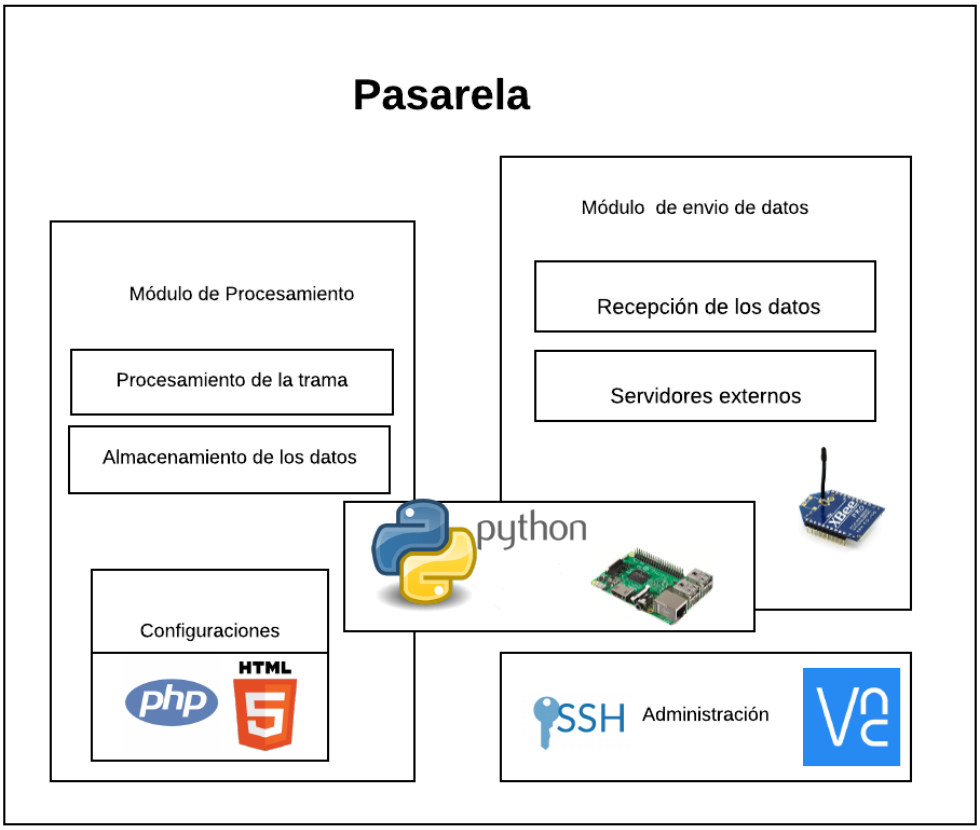
El inicio de la trama es identificado por el símbolo <=> y el final por el símbolo %. Esto permite tener un control sobre la identificación de cada trama recibida, el símbolo # se utiliza como un separador de cada dato que forma parte de la trama. La trama realizada se basó en la trama de los dispositivos Libelium (Libelium, 2018).

2.4 Pasarela

La recolección de los datos es realizada por un sistema al cuál denominaremos *pasarela;* es la encargada de procesar todas las tramas enviadas por cada uno de los nodos que formen parte de la red de sensores (WSN).

El sistema está compuesto por un RPi3 el cual contará con un SO, esto permitirá procesar y almacenar los datos recibidos. Para la comunicación con los nodos se utiliza el módulo XBee de la marca Digi Mesh, además de contar también con el módulo WI-FI el cual permitirá consumir servicios de servidores externos con la finalidad de acoplar el sistema a infraestructuras de terceros. Ver Figura 16

Figura 16
Arquitectura de la pasarela



Arquitectura de la pasarela

La instalación del SO y las configuraciones iniciales se encuentran detalladas en el manual de la Pasarela (Anexo 2).

2.4.1Módulo de procesamiento

Una vez instalado y configurado se procede con la programación en el lenguaje Python, el cual se dividió en 2 módulos.

2.4.1.1 Procesamiento de la trama

Antes de comenzar el procesamiento de los datos la pasarela tiene que pasar por una etapa de comprobación del sistema, este proceso es el encargado de reconocer cuantos nodos se encuentran dentro de la red con la finalidad de determinar un espacio específico en el *buffer*.

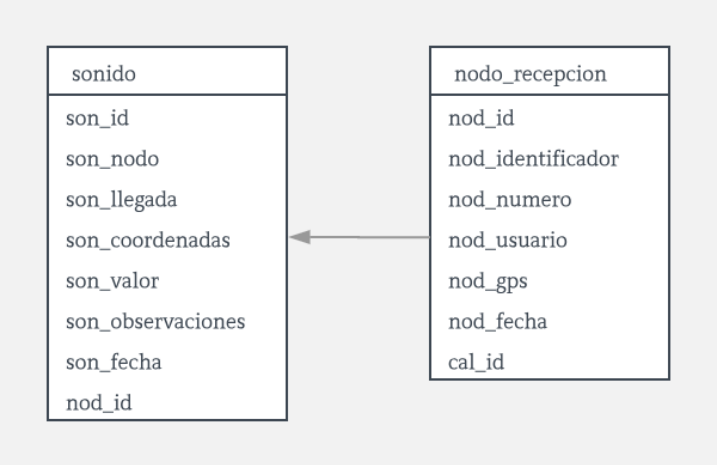
Al finalizar con la etapa de comprobación del sistema la pasarela comienza la etapa de procesamiento.

En esta etapa se obtiene las tramas que se encuentran en el buffer las mismas que pasan a ser descompuestas en datos individuales, esta descomposición se realiza en base a las reglas planteadas en la tabla 5 de la sección 2.3.6.1 de armado de la trama.

2.4.1.2 Almacenamiento de los datos

Para poder realizar el proceso de almacenamiento de los datos se tiene que tener instalado y configurado el servidor y un gestor de BD que para este caso se utilizó Apache y MySQL, este proceso puede ser revisado en el manual de instalación y configuración del sensor de sonido. Ver Figura 17.

Figura 17
BD Pasarela



Estructura de la BD para la pasarela.

Al tener la Base de datos configurada se procede con la implementación del módulo de conexión y los métodos de alteración los mismos que permitirán realizar el almacenamiento de todos los datos obtenidos en la trama recibida.

2.4.2 Módulo de envió de datos

2.4.2.1Recepción de los datos

El software está configurado para tener un espacio de memoria que se auto asigna en la etapa de calibración, esto permite almacenar cada trama que llega de los diferentes nodos que forman parte de la red WSN, el hardware que permite la recepción es el módulo XBee. El RPi3 de la pasarela cuenta con una tarjeta de memoria de 16 Gigabyte (GB), de los cuales 6 GB se le asigna al SO dejando 10 GB de espacio de memoria para el almacenamiento de los datos.

2.4.2.2 Servidores externos

La implementación de servidores externos permite enviar los datos de los sensores a otras infraestructuras, dejando de ser un sistema cerrado a un solo sistema.

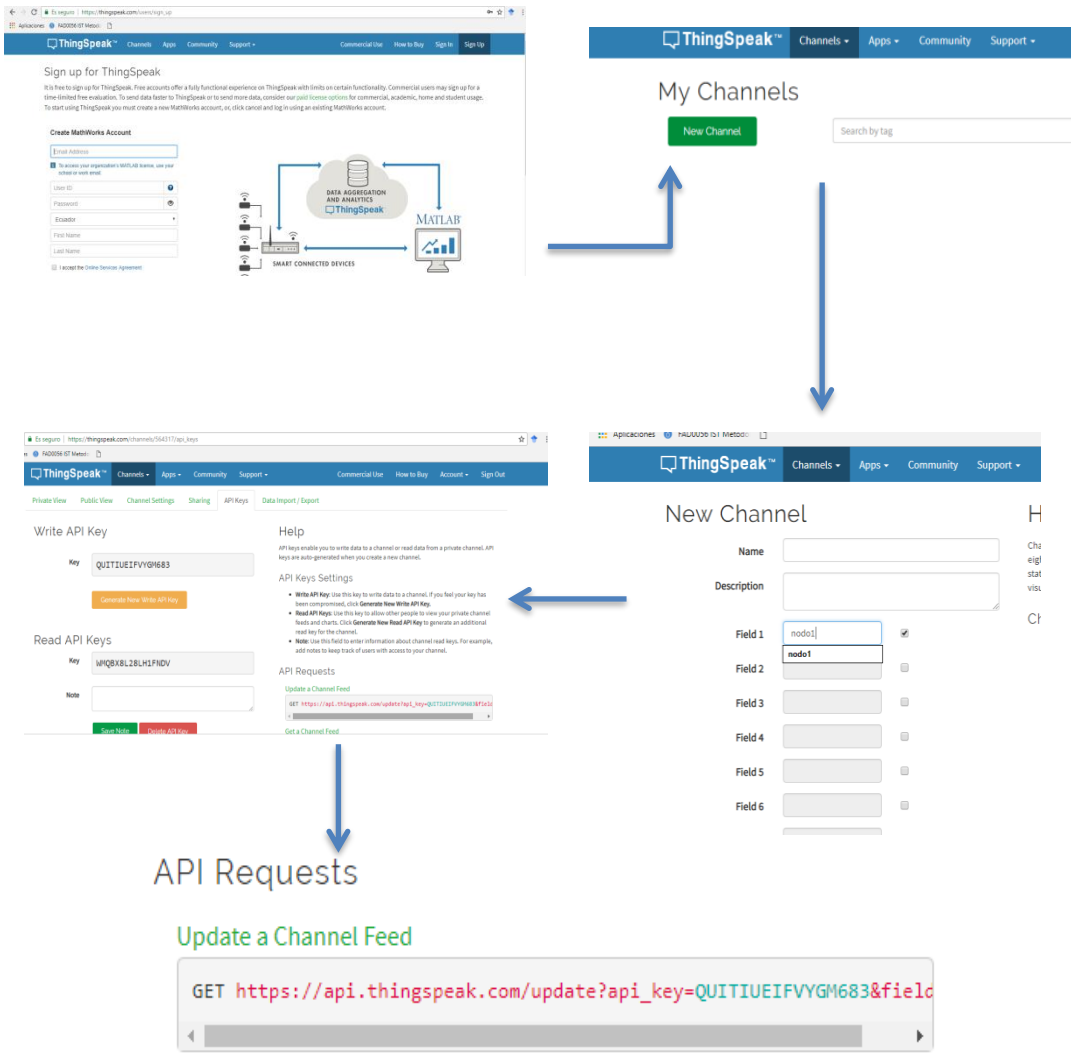
Esta implementación se realiza por medio de servicios los mismos que trabajan bajo el protocolo *Hypertext Transport Protocol Secure* (HTTPS) (Fielding et al., 2014) y

una petición GET. Para este caso se utilizó la infraestructura thingspeak para poder generar el servicio se tiene que seguir los siguientes pasos:

1. Ingresar a thingspeak.com
2. Crear una cuenta
3. Crear un nuevo canal
4. Configurar el canal

En esta configuración se define la cantidad de sensores que se van a utilizar, para este caso únicamente se configuró un solo sensor generando la dirección del servicio: https://api.thingspeak.com/update?api_key=QUITIUEIFVYGM683&field1=valor. Ver Figura 18.

Figura 18
Configuración thingspeak



Procesos para la configuración de un servicio externo en el servidor ThingSpeak.

2.5 Configuración de la red

Para la transmisión de los datos existen dos modalidades: API y AT, el modo API implementa configuraciones extensas y complejas que permiten tener un mejor manejo de tramas (Cacuango, 2017). Utilizar el modo API exige mucha destreza y dominio del tema, lo que alargaría la culminación del prototipo, desfasándose del tiempo de finalización propuesto para el sistema. Por tal motivo se utilizó el modo AT, este modo permite realizar una comunicación en topología estrella, haciendo una comunicación directa entre el nodo de recepción y la pasarela.

EL modo AT permite una comunicación punto a punto sin necesidad de implementar configuraciones extensas para el manejo de las tramas que son enviadas o recibidas.

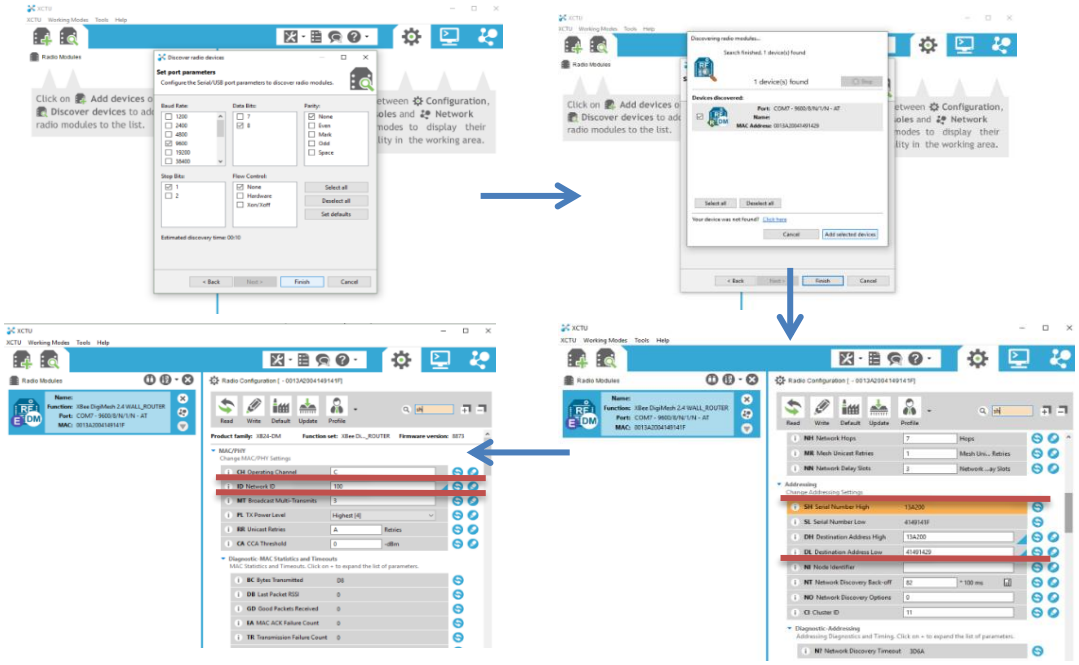
Para este sistema se han configurado dos nodos, uno es un prototipo físico y otro únicamente envía la trama por el programa XCTU (Virtual). Para realizar esta configuración se identifica los XBee que formarán parte de los nodos de recepción, y el XBee que formara parte de la pasarela. Con la finalidad de obtener los identificadores únicos de cada antena- Los identificadores serán configurados en dirección de envió (DH) y en la dirección de recepción (DL). Ver Figura 19.

Figura 19
Reconocimientos de los identificadores de las antenas XBee

Identificadores de las antenas.

Al tener identificado a cada uno de los XBee se realiza la configuración con el programa XCTU, el cual permite modificar y guardar los cambios que se realiza en el Firmware de las antenas XBee, además de modificar las direcciones de las antenas se tiene que modificar el identificador único de la red (PAN ID). Ver Figura 20.

Figura 20
Configuración XBEE



Proceso de configuración de las configuraciones de XBee, DH, DL, PAN ID, parámetros principales para la configuración.

Esto permite que cada módulo XBee que pertenece a un nodo, únicamente pueda comunicarse al módulo XBee que pertenece a la pasarela. Esto con la finalidad de que la trama únicamente pueda ser enviada a la pasarela, mientras que el XBee de la pasarela pueda reconocer a todos los nodos que pertenecen al sistema de medición.

2.6 Administración del RPi3

Para la instalación, configuración y programación de los diferentes paquetes, librerías y software que se implementó en cada uno de los RPi3 se utilizó dos procedimientos.

El primer procedimiento es habilitar el SSH, el mismo que permite tener una conexión remota al RPi3, en la que únicamente nos permite utilizar líneas de comandos para poder interactuar con el mismo

El segundo procedimiento es habilitar el servicio VNC, el mismo que facilita tener una conexión remota que habilita una terminal con la interfaz gráfica del RPi3 permitiendo realizar cualquier procedimiento.

3. RESULTADOS

El sistema de medición de ruido consta de diferentes módulos, los mismos que necesitaron ser avalados y analizados de la siguiente manera:

- Sensor sonido: con celular, sin celular, con calibrador, ambiente controlado, ambiente no contralado.

- Trasmisión de red.

- El consumo de energía y la estimación de tiempo de vida de la batería.

3.1 Sistema con un sensor de sonido

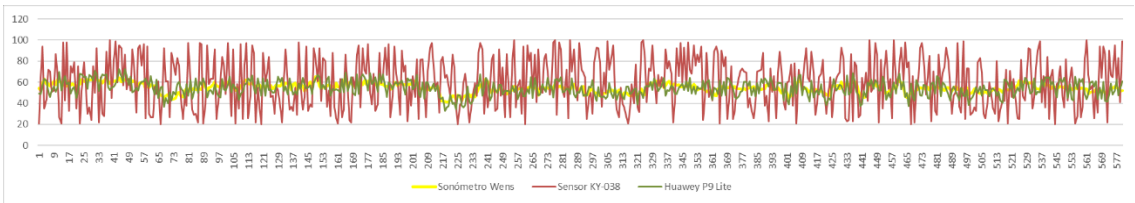
3.1.1 Sensores del nodo de recepción

La primera prueba es una evaluación, entre un sensor de sonido KY-038 un teléfono Huawei p9 Lite, con un sonómetro Wens, este análisis está enfocado en reconocer la calidad de datos que puede proporcionar cada uno de estos dispositivos. Por esta razón, el uso de coeficientes de similitud es necesarios para decidir cuál es el mejor dispositivo. Para esto se utiliza el método que calcula el error de la raíz cuadrada-media (RMSE), el cual indica la distancia media entre los puntos evaluados.

El dispositivo de mejor calidad de datos es aquel que se acerca más al valor de cero, esto se puede clarificar en la forma de la señal que tiene cada dispositivo. Esta prueba se realizó por un periodo de 10 minutos, en la cual los dispositivos en cuestión estuvieron expuestos a iguales condiciones y a un mismo nivel de ruido. Esta prueba obtuvo 577 datos de cada dispositivo Ver Figura 21.

Figura 21

Gráfica de los dispositivos de medición evaluados



Representación de los datos capturados cada segundo por los dispositivos durante un periodo de tiempo de 10 minutos.

Los resultados permitieron calcular su RMSE entre el sonómetro Wens, y lo dispositivos: sensor KY-038 y el teléfono Huawei p9Lite, esto permitió tener una referencia de que dispositivo utilizar en el sistema de medición. Ver Tabla 6.

Tabla 6
Comparativa entre dispositivos de medición de ruido

Descripción	KY-038	Huawei P9 Lite
RMSE	6,06046365dB	1,49193233DB

RMSE del sensor KY-038 y el teléfono Huawei P9 Lite en comparativa con el sonómetro Wens 1361.

Los resultados obtenidos se deben a que el sonómetro Wens cuenta con una circuitería más compleja que permite amplificar la señal y eliminar el ruido no deseado, mientras que el sensor de sonido no cuenta con esta.

3.1.2 Calibración

Esta etapa la realizó con la colaboración del instituto IERSE de la Universidad del Azuay. Para la calibración se utilizó un calibrador acústico clase 1, modelo 3M AC-300, este dispositivo emite un nivel acústico de 114dB a una frecuencia de 1KHz y 500Hz.

Este dispositivo es colocado en el micrófono del sonómetro Wens. El calibrador emite una frecuencia y un nivel acústico constante, este caso se repitió durante 10 veces a una frecuencia de 1Khz y 10 veces a una frecuencia de 500Hz, estos resultados nos permitirán obtener un promedio de la variación entre el calibrador y el sonómetro. Ver Figura 22.

Figura 22
Imagen del proceso de calibración

Forma de colocar el calibrado en el micrófono del sonómetro Wens.

Los resultados obtenidos de las muestras se presentan en la siguiente tabla. Ver Tabla 7

Tabla 7
Muestras obtenidas al realizar el progreso de calibración

Muestra	1000 Hz (114dB)	500 Hz(114dB)
1	116.0	113,6
2	116.3	113,6
3	116.02	113,6
4	116.01	113,6
5	116.06	113,6
6	116.9	113,6
7	116.0	113,6
8	116.0	113,6
9	116.	113,6
10	116.0	113,6
Promedio	116.2	113,6

10 muestras con una frecuencia de 100Hz y 500Hz, obteniendo un valor de calibración de 2dB a una frecuencia de 1000Hz y un valor de 0.6 dB a una frecuencia de 500Hz.

3.1.3 Validación

3.1.3.1 Ambiente Controlado

Para validar los datos del sistema, se realizó una primera prueba en un ambiente controlado. Esta prueba consistía en utilizar una computadora de marca Dell, la misma emitía un sonido constante en periodos de tiempo de un minuto, la intensidad se cambiaba al terminar.

Esta prueba se realizó por dos ocasiones seguidas, ubicando al sistema de medición y al sonómetro 3M (PCE Ibérica, 2017) a una distancia de 50 centímetros del computador, se tiene que considerar que los dispositivos estaban previamente calibrados. Ver Tabla 8.

Tabla 8
Pruebas de validación

	Prueba1								
	1	2	3	4	5	6	7	8	9
Sonómetro3M Calibrado	63,10	74,60	74,90	82,00	84,00	89,00	91,20	94,10	97,60
Sistema de medición calibrado	65,31	74,38	76,59	83,97	85,92	91,01	94,71	97,59	101,00
Sistema de medición sin calibración	63,31	72,38	74,59	81,97	83,92	89,01	92,71	95,59	99,00

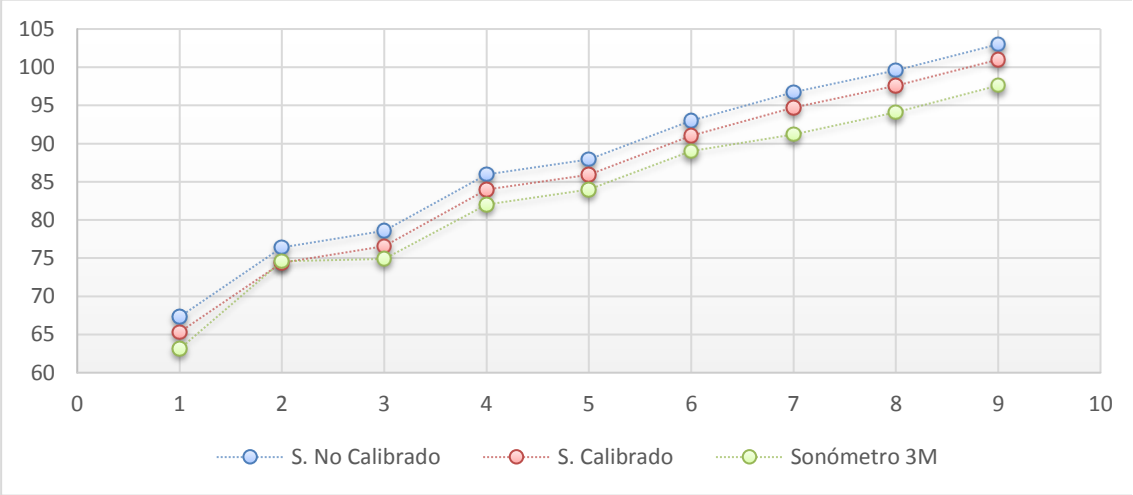
Prueba 2									
	1	2	3	4	5	6	7	8	9
Sonómetro3M Calibrado	67,20	71,80	77,90	82,10	85,80	89,90	91,90	92,10	87,00
sistema de medición calibrado	69,91	74,81	80,56	84,22	85,97	92,52	95,21	95,03	89,36
Sistema de medición sin calibración	67,91	72,81	78,56	82,22	83,97	90,52	93,21	93,03	87,36

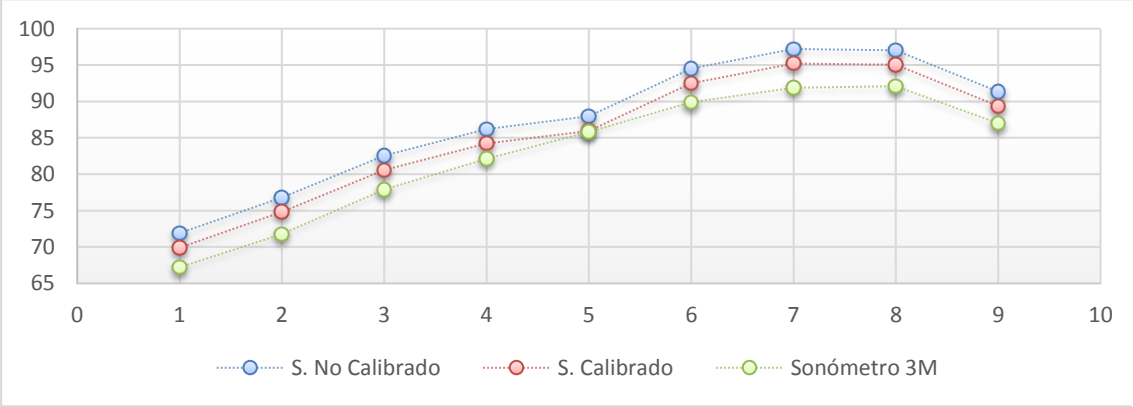
Promedio de resultados obtenidos en dB de cada minuto obtenidos por el sistema de medición y el sonómetro 3M.

Al culminar la primera prueba se obtuvo un promedio de 55 muestras por minuto por parte del sistema de medición, los mismos que fueron promediados para acoplarse a los datos obtenidos por el sonómetro 3M, ya que por su arquitectura no permitía almacenar los datos cada segundo y únicamente se podía obtener el promedio de cada minuto de muestras. Ver Figura 23.

Figura 23

Promedio de los resultados





Puntos de los promedios del sistema calibrado, no calibrado y del sonómetro 3M.

Para la validación, se calculó el error relativo en cada punto, mientras que para la validación de la prueba se utilizó el RMSE. Ver Tabla 9.

Tabla 9
Pruebas de validación

Prueba1										
	1	2	3	4	5	6	7	8	9	RMSE
sistema de medición calibrado	0,067	0,024	0,049	0,048	0,047	0,045	0,060	0,058	0,055	2,48
Sistema de medición sin calibrado	0,035	0,003	0,023	0,024	0,023	0,023	0,039	0,037	0,035	4,36
Prueba 2										
	1	2	3	4	5	6	7	8	9	
sistema de medición calibrado	0,070	0,070	0,060	0,050	0,025	0,051	0,058	0,054	0,050	2,58
Sistema de medición sin calibrado	0,040	0,042	0,034	0,026	0,002	0,029	0,036	0,032	0,027	4,52

Calculo del error relativo en cada minuto además del RMSE.

3.1.3.2 Ambiente No Controlado

Esta prueba se realizó en las instalaciones de la universidad del Azuay en un periodo de 10 minutos, la misma consistía en tener al sistema de medición y al sonómetro 3M a un mismo nivel y expuesto a iguales condiciones, lo que permitió obtener datos que pudieran dar a conocer las variaciones de dB entre ambos. Ver Tabla 10.

Tabla 10
Pruebas de validación

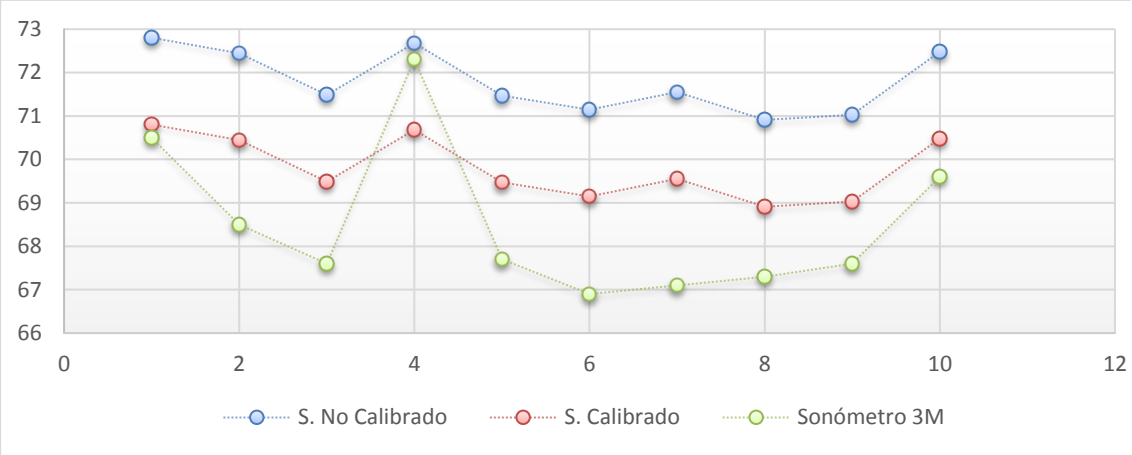
Datos										
	1	2	3	4	5	6	7	8	9	10
Sonómetro3M Calibrado	70,5	68,5	67,6	72,3	67,7	66,9	67,1	67,3	67,6	69,6
sistema de medición calibrado	70,80	70,45	69,49	70,68	69,47	69,15	69,55	68,91	69,03	70,48
Sistema de medición sin calibrado	72,80	72,45	71,49	72,68	71,47	71,15	71,55	70,91	71,03	72,48

Promedio de resultados obtenidos cada minuto por el sistema de medición y el sonómetro 3M.

Al igual que las muestras de la primera prueba, los datos que del sistema se promediaron para acoplarse a los datos del sonómetro 3M, Ver Figura 24.

Figura 24

Comparativa del sistema de medición y el sonómetro 3M



Puntos de los promedios del sistema calibrado, no calibrado y del sonómetro 3M en un ambiente no controlado.

Para cada punto se calculó el error relativo, el cual da una idea de cómo se comportó el sistema durante ese minuto, además de utilizar RMSE para tener una validación de toda la prueba. Ver Tabla 11.

Tabla 11

Pruebas de validación

	Datos									
	1	2	3	4	5	6	7	8	9	RMSE
sistema de medición calibrado	0,004	0,028	0,028	0,022	0,026	0,034	0,037	0,024	0,021	0,73
Sistema de medición sin calibrado	0,033	0,058	0,058	0,005	0,056	0,063	0,066	0,054	0,051	1,68

Calculo del error relativo en cada minuto además del RMSE.

3.2 Procesamiento del nodo de recepción y pasarela

Para la evaluación de una placa que permita el procesamiento, tanto en el nodo de recepción y la pasarela, se tomó a consideración el tiempo de aprendizaje para la correcta manipulación de estas, ya que utilizando el Arduino Uno se prevé culminar con el nodo de recepción en un tiempo de 120 horas mientras que con el RPi3 en 80 horas.

Para el procesamiento tanto del nodo de recepción y la pasarela se necesita tener algunas características a consideración. Se escoge la plataforma RPi3 por facilitar el uso de dispositivos Plug and Play (Barros & Astudillo, 2015), así como la instalación de software de manera sencilla gracias al SO. Ver Tabla 12.

Tabla 12
Comparativa entre la Placa Arduino UNO y la placa RPi3

Característica	Arduino UNO	RPi3
Velocidad Procesador	16Mhz	700Mhz
RAM	2KB	256MB
Voltaje	5V	5V
S. O	NO	Raspbian
Puertos de comunicación	NO	SI
Interfaz USB	NO	SI
Ethernet	NO	SI
Precio	$15	$35

Características que necesita el nodo de recepción y la pasarela, el Arduino Uno no cuenta con algunas, pero se pueden conseguir módulos externos que se acoplen a la placa en cuestión.

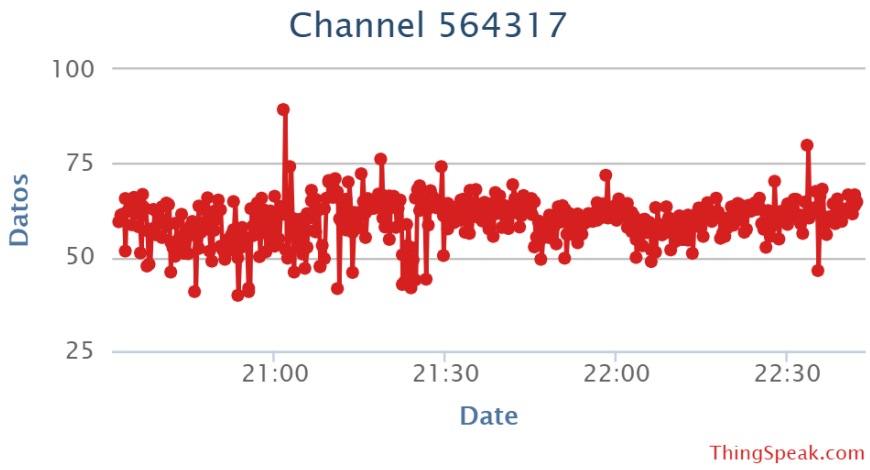
3.2.1 Recepción de datos en la pasarela

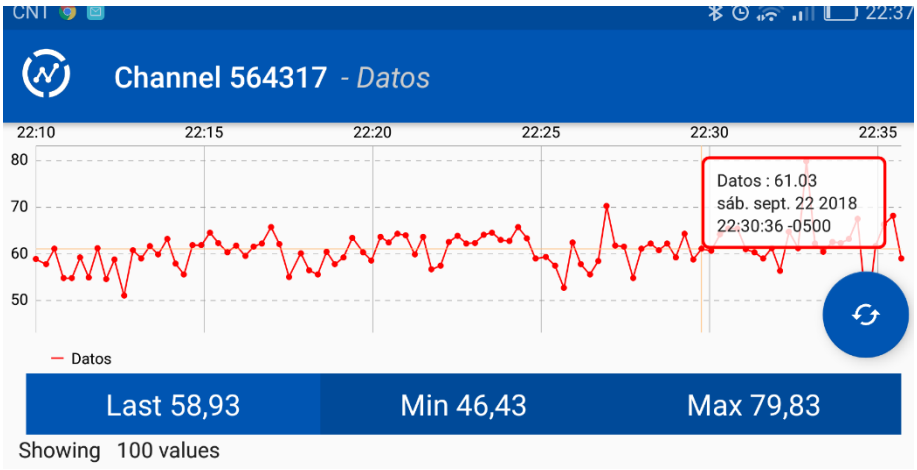
Las tramas que son enviadas por el nodo de recepción tienen un identificador el cual permite reconocer si alguna de estas no fue recibida o procesada. Se realizó una prueba de 2 horas que consistía en tener dos nodos de recepción enviando datos a la pasarela, se tiene que considerar que en esta prueba uno de los nodos fue simulado ya que únicamente se cuenta con un prototipo funcional.

Se calculó el *Packet Delivery Ratio* (PDR) o Tasa de recepción de paquetes, misma que se define como el número de paquetes recibidos (Tuteja et al., 2014) en la pasarela para el total de paquetes trasmitidos desde los nodos. En nuestro caso el PDR es del 100% para nuestro escenario de prueba a una distancia de 1 m entre los nodos.

Además de esto la pasarela cumple con una segunda función la que permite consumir servicios de infraestructuras externas. Para probar este funcionamiento se hizo uso del servicio generado en la plataforma thingspeak, lo que permitió validar el envío de datos a otras infraestructuras. Ver Figura 25.

Figura 25
Gráficas generadas por la plataforma thingspeak





Duración del muestreo de 2 horas continuas, la primera parte de la imagen muestra todos los datos percibidos en la Web, la segunda parte muestra un rango de 100 valores en la APP móvil

3.3 Consumo energético

Para calcular el tiempo de funcionamiento del dispositivo se utiliza la fórmula 5.

$$Horas = \frac{Carga\ Bateria(mAh)}{ConsumomA} \quad (5)$$

La fórmula 5 necesita el consumo de los dispositivos que forman parte del nodo de recepción. Ver Tabla 13.

Tabla 13
Consumo Energético

Dispositivos	Consumo
RPi3	350mA
Xbee Digi Mesh	280mA
Sonómetro Wens	300mA
Total	930mA

Consumo energético de RPi3, XBee Digi Mesh, Sonómetro Wens 1361, datos obtenidos por los fabricantes de cada dispositivo.

Para la prueba de consumo energético se utilizó un *power bank* con una capacidad de 15000mAh según las características del fabricante, por lo tanto el consumo el funcionamiento del nodo de recepción debería ser de 16 horas de funcionamiento constante.

El nodo de recepción utilizo el *power bank* como fuente de alimentación durante un periodo de 3 horas y 36 minutos con un desgaste del 25% del *power bank*, lo que indica que el sistema únicamente tendría un funcionamiento de 14 horas y 40 minutos.

4. DISCUSIÓN

El sistema de medición de ruido está estructurado por dos componentes esenciales: el nodo de recepción y la pasarela.

4.1 Sensor

Al realizar las pruebas de evaluación obtuvimos un error cuadrático medio de 6.060 del sensor KY-038 con respecto a la señal obtenida por el sonómetro Wens, esto demuestra una baja calidad de datos, se atribuye a que este sensor no cuenta con una placa de amplificación y eliminación de ruido (Mora, 2003), por lo que se optó por la utilización del sonómetro Wens para que forme parte del nodo de recepción.

Al culminar los prototipos, el nodo de recepción y la pasarela, se realiza la calibración y validación del sistema de medición con otros sistemas, la etapa de calibración benefició al sistema reduciendo su error en 2dB por el método de offset.

Al tener un sistema calibrado las pruebas en un ambiente controlado generan un error mínimo de 0.05 dB y un error máximo de 3dB, en comparación al sonómetro 3M clase 1, mientras que en un ambiente abierto la diferencia del error se sitúa en un valor mínimo de 0.30dB y 2.45dB. Estos resultados sitúan el sistema de medición y permiten colocar al dispositivo en un tipo clase 3. Los resultados son parecidos a los obtenidos en trabajos similares, 3 a 4 dB dependiendo del método de calibración (Barros, 2015).

4.2 Red

Cada nodo que forma parte del sistema tiene la capacidad de transmitir los datos inalámbricamente por el protocolo ZigBee (sobre IEEE 802.15.4) esto permite tener un monitoreo continuo sin la necesidad de invertir demasiado en recursos humanos, generando una reducción de costos.

4.3 Energía

El estudio del consumo de energía en este tipo de implementaciones es un amplio campo de estudio. Nuestras estimaciones siguieren la factibilidad de implementar campañas con duraciones de hasta 14 a 16 horas. Sin embargo, un protocolo de pruebas exhaustivas es necesario para confirmar esto.

4.4 Flexibilidad

La placa Arduino Uno puede implementar módulos externos que permiten tener una interfaz USB, mientras que la placa RPi3 ya incluye estas características sin necesidad de utilizar módulos externos. La placa Arduino Uno por implementar módulos externos y manejar un lenguaje de programación propio necesita un mayor esfuerzo y tiempo de aprendizaje, esta ampliación de tiempo conlleva a sobrepasar el tiempo de culminación propuesto para el sistema, mientras que el RPi3 por tener un sistema operativo basado en una distribución de Linux reduce este tiempo, evitando sobrepasar el plazo de culminación del sistema.

La selección de la plataforma adecuada para realizar este proyecto no es sencilla, pues existen varias plataformas utilizadas en implementaciones similares. Los criterios de selección dependen también de los objetivos de los autores. Así por ejemplo, si nuestro objetivo es limitar el consumo de energía para permitir un monitoreo prolongado estaríamos tentados a utilizar una plataforma como Libelium (libelium.com), ya que las placas poseen mejoras realizadas a placas Arduino; el costo de un dispositivo de despliegue está entre los 700 y 1200 USD. Si deseáramos una mayor precisión de las medidas de sonido, quizás deberíamos diseñar nuestro propio circuito para tratar la señal de la mejor manera. Sin embargo, es una capacidad que no se posee. Por otra parte, este trabajo tiene una duración determinada de cuatro meses y existen varios temas en los que se puede profundizar. Es decir, para cumplir con el alcance del trabajo, se decidió utilizar una plataforma que posea mejores prestaciones en términos de facilidad de uso y adaptación y con posibilidad de uso “Plug and Play”. Un factor a considerar en la selección de plataforma también incluye la estimación del tiempo necesario para aprender a realizar las tareas necesarias (preparación de trama, comunicación, calibración, etc.) con determinada plataforma.

4.5 Servidor y consumo de datos.

Si bien resulta sencillo y rápido de levantar servicios de tipo thingspeak, quizás en el futuro existan requerimientos para integrar y consumir de manera estandarizada los datos lo que implicaría destinar recursos para implementar los enlaces y las consultas respectivas. Una alternativa consiste en utilizar más tiempo para poder utilizar estándares como por ejemplo OGC-SOS.

5. CONCLUSIONES

Como resultado de este trabajo, se obtiene un sistema de medición de ruido, el mismo que cuenta con un nodo que permite la recepción y envío de datos a la pasarela y esta realiza un procesamiento para el almacenamiento y envío de datos a servidores externos. Este proceso es automático y requiere de mínima intervención humana para su funcionamiento luego de realizar las configuraciones iniciales de calibración y de red.

Los componentes utilizados para la realización tanto del nodo y la pasarela son económicos, esto permitió tener un prototipo con un valor de $100 para el caso de nodo y un valor de $35 para la pasarela generando un costo total del sistema de medición de ruido de $135, lo que se clasificaría como bajo costo.

El sonómetro clase 1 de marca 3M perteneciente a la universidad del Azuay tiene un costo de $1500, esta inversión permite tener datos de calidad, pero no tienen los beneficios tales como transmisión inalámbrica de datos, ni automaticidad en la recolección y almacenamiento ya que es un sistema cerrado y únicamente se pueden obtener los datos de las mediciones con los programas pertenecientes al fabricante.

La implementación de la trama AT facilitó las configuraciones para la transmisión inalámbrica de los datos, esto evita tener un menor control en las tramas de envío ya que no cuenta con un protocolo de control.

Para el caso del nodo de recepción no se consideró el consumo de energía, únicamente se realizó una prueba con una fuente de energía externa la misma que permitió reconocer un consumo relativo que posee.

Las gráficas muestran la naturaleza lineal de la relación entre los dispositivos. Por esta razón se utilizó únicamente el método de calibración de desplazamiento u *offset*.

Sin la calibración estos resultados sobrepasarían en 2dB cada muestra, conllevando a tener un error de 5dB, esto evitaría situar al sistema en un tipo clase 3.

El nodo de recepción ya calibrado aún presenta una diferencia entre 0.22 y 3.5dB, lo que nos da a conocer que se tiene que buscar formas diferentes de calibración para reducir esta diferencia.

Dispositivos de bajo costo pueden hacer la diferencia y generar sistemas de gran utilidad, se necesita una inversión de tiempo y esfuerzo mayor, la cual permita realizar un estudio a fondo en cada sección del sistema, permitiendo encontrar mejores

dispositivos a menores costos, y con mayor rendimiento, además de estudiar otros métodos de calibración que permitan reducir el error de medición. Así como el uso de estándares bien conocidos para permitir una integración “plug and play”, pero esta vez a nivel de las mediciones recolectadas. (Estándares OGC).

El tiempo estimado, según nuestras evaluaciones es de 14 horas y 40 minutos, si esto tiempo se traslada a las campañas realizadas en la que cada muestra se realiza cada hora y por un periodo de 15 minutos, este sistema nos permitiría realizar una campaña de 58 muestras (2 días y 10 horas).

6. TRABAJOS FUTUROS

Estudiar las placas de procesamiento que permitan reducir costos y mejorar rendimiento y consumo de energía.

Mejorar la transmisión de los datos, buscar nuevas tecnologías que permitan realizar diferentes tipos de topologías.

Mejorar el software del sistema de medición de tal manera que busque tener un mayor control de los dispositivos que forman parte de él, permitiendo así apagarlos hasta que comience una nueva etapa de muestras, lo que permitiría ahorrar energía alargando el tiempo de funcionamiento del sistema.

Estudiar a fondo el modo API, el cual permitiría tener un mejor control del envío y recepción de las tramas en la red de sensores.

BIBLIOGRAFÍA

Alfie Cohen, M., & Salinas Castillo, O. (2017). Ruido en la ciudad. Contaminación auditiva y ciudad caminable. *Estudios Demográficos Y Urbanos*, *32*(1), 65–96.

Alonso, A. D. E. (2003). Contaminación acústica y salud Noise pollution and health. *Observatorio Medioambiental*, *6*, 73–95.

Chang, S.-F., Chen, C.-F., Wen, J.-H., Liu, J.-H., Weng, J.-H., & Dong, J.-L. (2015). Application and Development of Zigbee Technology for Smart Grid Environment. *Journal of Power and Energy Engineering*, *3*(4), 356.

G. Barros, F. T. (2015).. *Sound Noise Levels Calibration with Smart-Phones*, 6.

Alcaldía De Cuenca Red De Monitoreo Emov – EP, (2014). Informe de la calidad del aire, 2014. Cuenca– Ecuador

Rana, R., Chou, C. T., Bulusu, N., Kanhere, S., & Hu, W. (2015). Ear-Phone: A context-aware noise mapping using smart phones. *Pervasive and Mobile Computing*, *17*, 1–22.

Sanchez-Rosario, F., Sanchez-Rodriguez, D., Alonso-Hernández, J. B., Travieso-González, C. M., Alonso-González, I., Ley-Bosch, C., Quintana-Suárez, M. A. (2015). A low consumption real time environmental monitoring system for smart cities based on ZigBee wireless sensor network. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2015 International* (pp. 702–707).

Sanz, B. G., & García, F. J. G. (2003). *La contaminación acústica en nuestras ciudades*. Fundación“ La Caixa.”

Sevillano, X., Socoró, J. C., Al’\ias, F., Bellucci, P., Peruzzi, L., Radaelli, S., … others. (2016). DYNAMAP--Development of low cost sensors networks for real time noise mapping. Noise Mapping, 3(1).

Wessels, P. W., & Basten, T. G. H. (2016). Design aspects of acoustic sensor networks for environmental noise monitoring. *Applied Acoustics*, *110*, 227–234.

Velleman. (2017). Compatible microphone sound sensor module. Retrieved from https://www.robotshop.com/media/files/pdf/sound-sensor-module-arduino-datasheet.pdf

Wens. (2018). Product Sount Meter WS1361. Retrieved from http://www.wensn.com/html_products/WS1361-17.html

Pfretzschner, J. (2002). Interoperación nacional sobre verificación de sonómetros.

Hernández Juárez, J. R. (2009). Diseño de un sonómetro.

Solórzano Lescano, S. L. (2016). Sistema de medición de contaminación auditiva, empleando una red de sensores inalámbricos y sensores Ban. Quito: Universidad de las Américas, 2016.

Cacuango, C., & Vinicio, M. (2017). Implementación del modo API en una red de sensores WSN para la medición de contaminación auditiva. Quito: Universidad de las Américas, 2017.

Technologies, 3MQuest. (2012). 3M División de Salud Ocupacional y Seguridad Ambiental Calibrador 3MTM AcoustiCal AC-300. Retrieved from https://multimedia.3m.com/mws/media/888931O/catalogo-3m-calibrador-acoustical-ac-300.pdf

Mesh, D. (2018). XBee/XBee-PRO DigiMesh 2.4. Retrieved from https://www.digi.com/resources/documentation/digidocs/pdfs/90000991.pdf

Libelium. (2018). Waspmote Technical Guide. Retrieved from http://www.libelium.com/downloads/documentation/waspmote_technical_guide.pdf

Fielding, R., & Reschke, J. (2014). Hypertext transfer protocol (HTTP/1.1): Message

syntax and routing.

Barros, G., & Astudillo S., D. (2015). Off-the-Shelf Platform for remote monitoring of vital signs. Maskana, 6(Supl.), 21-27. Retrieved from https://publicaciones.ucuenca.edu.ec/ojs/index.php/maskana/article/view/694

Tuteja, A., Gujral, R., & Thalia, S. (2010). Comparative performance analysis of DSDV, AODV and DSR routing protocols in MANET using NS2. In Advances in Computer Engineering (ACE), 2010 International Conference on (pp. 330–333). IEEE.

PCE Ibérica S.L. (n.d.). Series SoundPro SE/DL. 2017. Retrieved from https://www.pce-iberica.es/manuales/manual-soundpro-se-dl.pdf

Anexo 1 (Manual Nodo de Recepción)

Resumen
Reporte técnico del nodo de recepción. Se empleó una red de sensores acústicos inalámbricos (WSN) para evaluar la transición y calidad de los datos que son percibidos y enviados, se realizó con dos nodos receptores compuestos por sonómetro Wens 1361 para la recepción de la información, la misma que será procesada por un RPi3 para poder ser enviada por un Xbee DigiMesh Serie1, el cual permite su transmisión con el protocolo 802.15.4. Un nodo coordinador compuesto por un Xbee Serie1 que permite la recepción de la información para ser enviada a un arduino mega el cual nos permite el procesamiento de la misma.

Introducción
Red de sensores

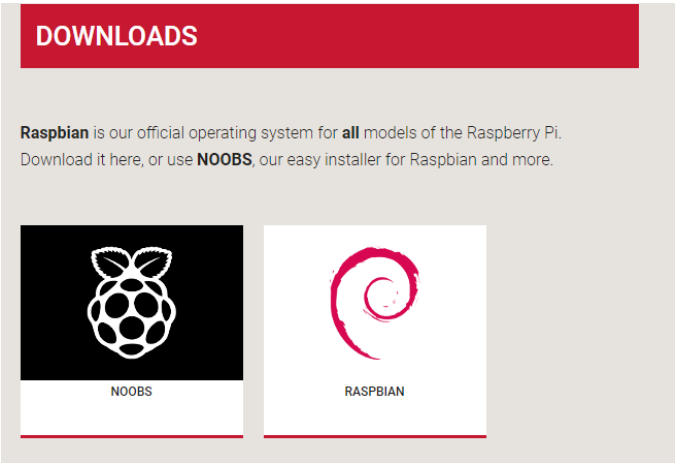
- Xbee serie 1
- Topología

Sistema de ruido
Sensor de sonido
Procesamiento
Arduino
RPi3
Materiales y métodos

- 2 Xbee Digi Mesh Serie 1
- 2 KY-038
- 2 Arduino UNO
- 1 Arduino Mega
- 1RPi3
- Juegos de cables

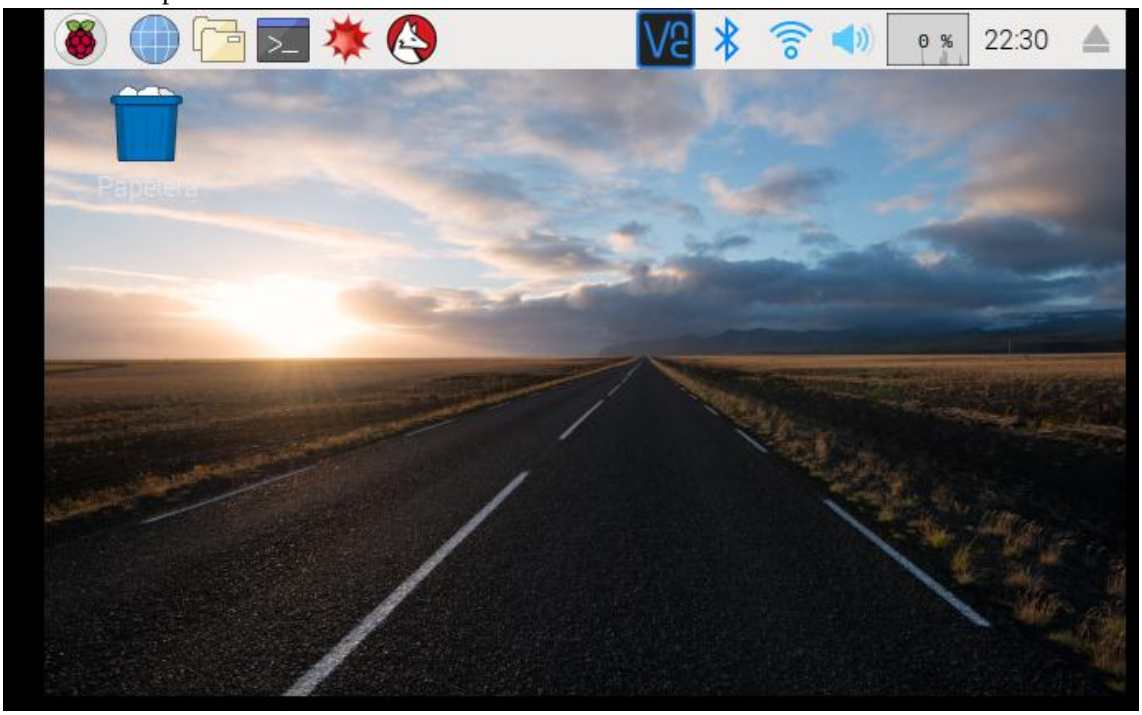
Experimentación
Instalación del Raspberry Pi3
El proceso de instalación del RPI3 se detalla a continuación.

1. Ingresar a la pagina https://www.raspberrypi.org/downloads/
2. Seleccionar el S.O Raspbian



3. Crear una micro SD Booteable con el SO descargado, para este proceso se utilizó el programa Win32DiskImage

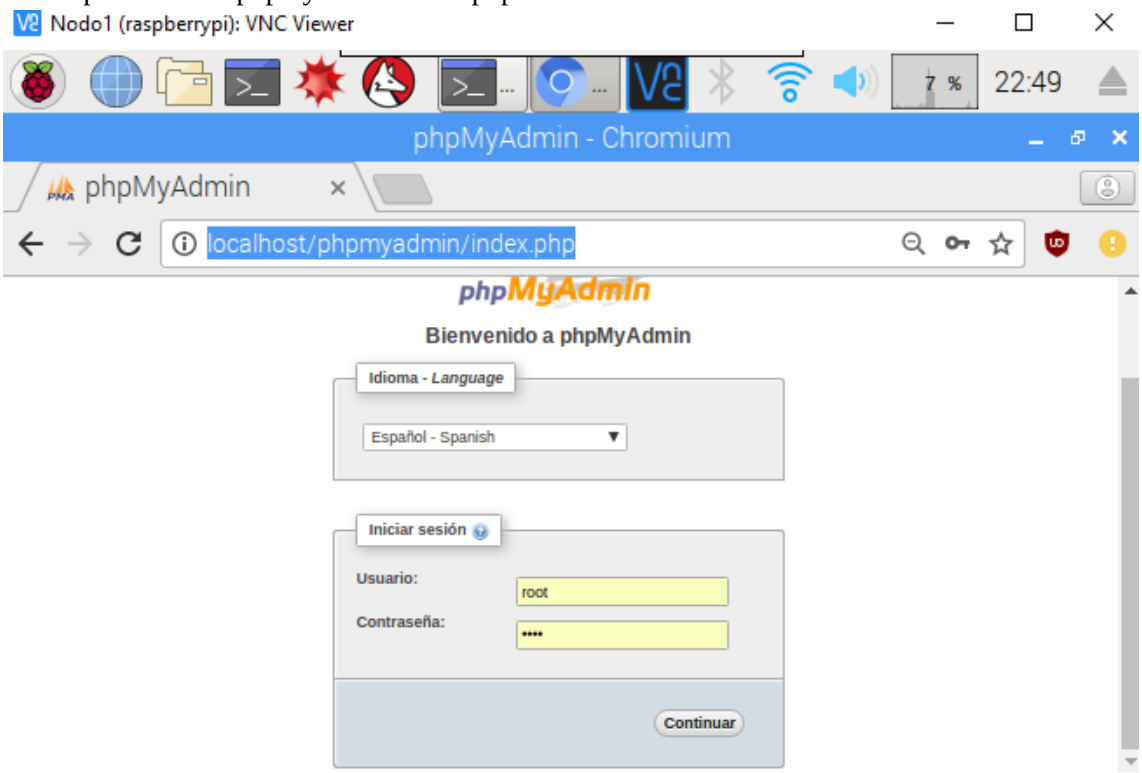
4. AL insertar la tarjeta en l aplaca RPi3 se conecta con una fuente de alimentación, lo que iniciara el dispositivo.



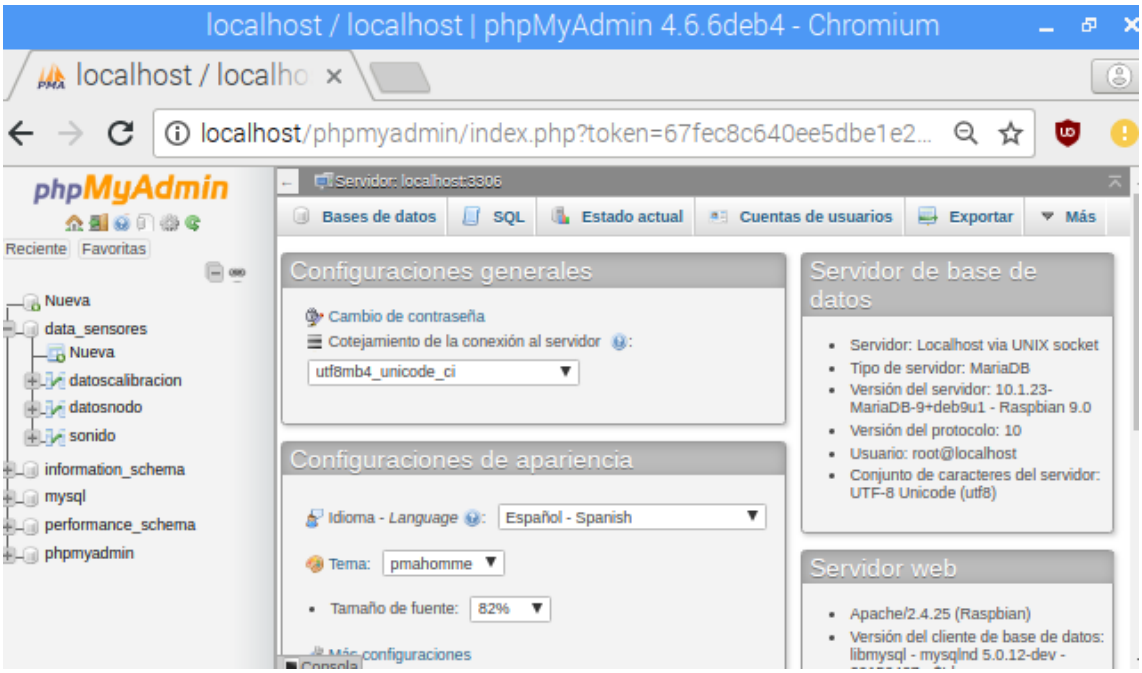
5. Instalación de Paquetes para la configuración del Nodo

```
1  sudo apt-get install python-mysql.connector
2  sudo apt-get install python-mysqldb
3  sudo apt-get install python-mysql-client
4  sudo apt-get install python-mysqldb
5  sudo apt-get install pymysql
6  pip install PyMySQL
7  sudo apt-get install python3-pymysql
8  sudo cat /dev/ttyACM0
9  sudo cat /dev/ttyS0
10 cd ..
11 cd /home/pi/Downloads/
12 cd ..
13 cd pi/Documents/
14 mv nodo ../../../var/www/html/
15 sudo mv nodo ../../../var/www/html/
```

6. AL tener instalado los paquetes, podemos configurar el phpmyadmin con la siguiente url: http://localhost/phpmyadmin/index.php



7. Ingresar con la contraseña e usuario root.



8. Ejecutar el Scritp a continuación

```
-- phpMyAdmin SQL Dump
-- version 4.6.6deb4
-- https://www.phpmyadmin.net/
--
-- Servidor: localhost:3306
-- Tiempo de generaciÃ³n: 25-09-2018 a las 00:12:01
-- VersiÃ³n del servidor: 10.1.23-MariaDB-9+deb9u1
-- VersiÃ³n de PHP: 7.0.30-0+deb9u1

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101                                                  SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101                                                  SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101                                                  SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de datos: `data_sensores`
--
CREATE DATABASE IF NOT EXISTS `data_sensores` DEFAULT
CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
USE `data_sensores`;

-- --------------------------------------------------------
--


--
-- Estructura de tabla para la tabla `datoscalibracion`
--
```

```
CREATE TABLE `datoscalibracion` (
  `id` int(11) NOT NULL,
  `cal_valor` float NOT NULL,
  `cal_fecha` date NOT NULL,
  `cal_usuario` varchar(200) NOT NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

-- --------------------------------------------------------
--

--
-- Estructura de tabla para la tabla `datosnodo`
--

CREATE TABLE `datosnodo` (
  `id` int(11) NOT NULL,
  `nodo_identificador` varchar(200) NOT NULL,
  `nodo_numero` int(11) NOT NULL,
  `nodo_usuario` varchar(200) NOT NULL,
  `nodo_gps` varchar(100) NOT NULL,
  `nodo_fecha` date NOT NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

-- --------------------------------------------------------
--

--
-- Estructura de tabla para la tabla `sonido`
--

CREATE TABLE `sonido` (
  `son_id` int(11) NOT NULL,
  `son_nodo` varchar(10) NOT NULL,
  `son_codigo` varchar(50) NOT NULL,
  `son_llegada` int(11) NOT NULL,
  `son_coordenadas` varchar(150) NOT NULL,
  `son_valor` varchar(10) NOT NULL,
  `son_observaciones` varchar(200) NOT NULL,
  `son_fecha` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Ãndices para tablas volcadas
--

--
-- Indices de la tabla `datoscalibracion`
--
ALTER TABLE `datoscalibracion`
  ADD PRIMARY KEY (`id`);

--
-- Indices de la tabla `datosnodo`
--
ALTER TABLE `datosnodo`
```

```
  ADD PRIMARY KEY (`id`);

--
-- Indices de la tabla `sonido`
--
ALTER TABLE `sonido`
  ADD PRIMARY KEY (`son_id`);

--
-- AUTO_INCREMENT de las tablas volcadas
--

--
-- AUTO_INCREMENT de la tabla `datoscalibracion`
--
ALTER TABLE `datoscalibracion`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT de la tabla `datosnodo`
--
ALTER TABLE `datosnodo`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT de la tabla `sonido`
--
ALTER TABLE `sonido`
  MODIFY `son_id` int(11) NOT NULL AUTO_INCREMENT;--
-- Base de datos: `phpmyadmin`
--
CREATE DATABASE IF NOT EXISTS `phpmyadmin` DEFAULT
CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
USE `phpmyadmin`;

-- --------------------------------------------------------
--
```

9 Programación del software en Python

Script Principal

```
#Importar librerias necesarias para la codificacion
#Fin importancion de librerias
import time
import RPi.GPIO as Salidas
from               funciones_comunes               import
conexionSonometro,conexionSerial,lecturaSonometro,envioDa
tosAT,contadorTrama,armarTrama,estadoError
from                 conexionBdNodo                import
obtenerDatosNodo,obtenerDatosCalibracion,insertaDatos
numeroNodo='n01'
serieSonometro='123132132123'

ledSonometro = 7
ledXbee = 12
Salidas.setmode(Salidas.BOARD)
Salidas.setup(ledSonometro,Salidas.OUT)
Salidas.setup(ledXbee,Salidas.OUT)
```

```
for cal in range(1, 3):
    estadoError(Salidas,ledXbee)
    estadoError(Salidas,ledSonometro)

try:
    estadoConexion=True
    #Inicio de las conexiones del sonometro y del xbee---
--------
    while estadoConexion:
        time.sleep(3)
        try:
            sonometro=conexionSonometro()
            xbee=conexionSerial()
            cabecera=obtenerDatosNodo()
            valCalibracion=obtenerDatosCalibracion()
            valGps="#GPS:11232132;151561561"

            if(not bool(sonometro)):
                estadoConexion = True
                print('Error-LecturaSonometro')
                estadoError(Salidas,ledSonometro)
            else:
                estadoConexion = False

            if(not bool(xbee)):
                estadoConexion = True
                print('Error-Xbee')
                estadoError(Salidas,ledXbee)
            else:
                if(not estadoConexion):
                    estadoConexion = False

            if(not bool(cabecera)):
                estadoConexion = True
                print('Error-CabeceraBd')
            else:
                if(not estadoConexion):
                    estadoConexion = False

            if(not bool(valCalibracion)):
                estadoConexion = True
                print('Error-DatoCalibracion')
            else:
                if(not estadoConexion):
                    estadoConexion = False

        except:
            print('ErrorEscritura-Lectura-Serial-Usb')
    while True:
        try:
            time.sleep(1)
            Salidas.output(ledSonometro,True)
            Salidas.output(ledXbee,True)
            contarTrama=contadorTrama()
            dB=lecturaSonometro(sonometro,valCalibracion)
```

```
tramaAT=armarTrama(cabecera,contarTrama,valGps,dB)
            envioDatosAT(xbee,tramaAT)
            if(True):

insertaDatos(str(serieSonometro),str(numeroNodo),int(cont
arTrama.replace("#","")),str(valGps),str(dB.replace("#MCP
:","")))
            #print(dB)
            print(tramaAT)

            if(not bool(dB)):
                sonometro=conexionSonometro()
                print('ErrorGeneral-Sonometro')
                estadoError(Salidas,ledSonometro)
            #print(tramaAT)
        except:
            print('ErrorGeneral-Xbee-Sonometro')
            estadoError(Salidas,ledXbee)
            xbee=conexionSerial()
except:
    print('Comunicate con la universidad del AZUAY')
```

Funciones

```
import sys
import usb.core
import time
import serial

sumar=True
valor=0

def conexionSonometro():
    try:

dev=usb.core.find(idVendor=0x16c0,idProduct=0x5dc)
        assert dev is not None
        resultado = dev
    except:
        resultado = False
    return resultado

def conexionSerial():
    try:
        try:
            ser = serial.Serial('/dev/ttyUSB0',9600)
            resultado=ser
        except:
            ser = serial.Serial('/dev/ttyUSB1',9600)
            resultado=ser
    except:
         resultado = False
    return resultado
```

```
def lecturaSonometro(sonometro,calibracion):
    try:
        ret = sonometro.ctrl_transfer(0xC0 , 4, 0, 0, 200)
        dB = (ret[0]+((ret[1]&3)*256))*0.1+30
        dB = dB + calibracion
        dB = "{0:.2f}".format(dB)
        dB = str(dB).zfill(6)
        dB = "#MCP:"+dB
    except:
        dB=False
    return dB

def envioDatosAT(xbee,tramaAt):
    try:
        xbee.write(str.encode(str(tramaAt)))
    except:
        xbee.write(str.encode(str(000.00)))

def contadorTrama():
    try:
        global sumar
        global valor

        if(sumar):
            valor=valor+1
        else:
            valor=valor-1
        if(valor == 999):
            sumar=False
        elif(valor == 0):
            sumar=True

        resulValor="#"+str(valor).zfill(3)
    except:
       resulValor=-100

    return resulValor

def
armarTrama(datosCabecera,datosIdentificador,datosGps,dato
sSonometro):
    try:

tramaAt=str(datosCabecera)+str(datosIdentificador)+str(da
tosGps)+str(datosSonometro)+"%"
        tramaAt="<=>#"+str(len(tramaAt)).zfill(3)+tramaAt
    except:
        tramaAt=False
    return tramaAt

def estadoError(salida,numero):
    try:
        salida.output(numero,True)
        time.sleep(1)
        salida.output(numero,False)
```

```
    except:
        tramaAt=False

#d=armarTrama(1,2,6)
#print(d)
```

Conexión

```
import pymysql.cursors
import datetime
connection = pymysql.connect(host='localhost',
                        port=3306,
                        user='root',
                        passwd='root',
                        db='data_sensores',
                        autocommit=True)

def obtenerDatosNodo():
    try:
        with connection.cursor() as cursor:
            sql     =     "SELECT     nodo_numero     as
id,nodo_identificador as num from datosnodo ORDER by ID
DESC Limit 1"
            cursor.execute(sql)
            resultado=cursor.fetchone()

cabArmada="#"+str(resultado[0])+"#"+str(resultado[1])
            cabecera=cabArmada
    except:
        cabecera = False

    return cabecera

def obtenerDatosCalibracion():
    try:
        with connection.cursor() as cursor:
            sql   =   "SELECT   cal_valor   as   valor   from
datoscalibracion ORDER by ID DESC Limit 1"
            cursor.execute(sql)
            resultado=cursor.fetchone()
            calibracion=float(resultado[0])
    except:
        calibracion = False

    return calibracion

def
insertaDatos(son_codigo,son_nodo,son_llegada,son_coordena
das,son_valor):
        try:
            fechaActual= datetime.datetime.now()

            with connection.cursor() as cursor:
```

```
                sql         =         "INSERT         INTO
sonido(son_codigo,son_nodo,son_llegada,son_coordenadas,so
n_valor,son_observaciones,son_fecha)                VALUES
(%s,%s,%s,%s,%s,%s,%s)"
                cursor.execute(sql,(
                    son_codigo,
                    son_nodo,
                    son_llegada,
                    son_coordenadas,
                    son_valor,
                    'Ninguna',
                    fechaActual))
        finally:
            pass



#insertaDatos('','',1,'','')
#d=obtenerDatosNodo()
#print(d)
```

10 Programación del servidor web

Código PHP-HTML(Índex)

```
<?php require 'head.php'?>
  <br>
<center><img src="imagenes/portada.png"></center>
<?php             require             'controlador.php';
$valores=obtenerDatos();?>
  <section class="p-b-0">
    <div class="container">
      <div class="heading">
        <h2>Bienvenido | Configuraciones NODO</h2>
        <p>En esta secciÃ³n nos permite la configuracion
del nodo ademas de establecer el parametro de calibraciÃ³n
</p>
      </div>
    </div>
  </section>

  <section class="p-t-10">
    <div class="container">
      <div class="row">
        <div class="col-lg-7 mx-auto">

          <?php if(isset($_GET['core'])){?>
             <div   class="alert   alert-success   alert-
dismissible" role="alert">
              <button  type="button"  class="close"  data-
dismiss="alert" aria-label="Close">
              <span aria-hidden="true">&times;</span>
              </button>
              <b>CORRECTO!</b> Modificaciones Correctas.
            </div>
            <?php } ?>
```

```
          <form          id="form1"          method="POST"
action="procesos.php">
            <input   type="hidden"   class="form-control"
id="configuracion" name="configuracion">
            <div class="form-group">
              <label
for="idConfiguracion">#ConfiguraciÃ³n</label>
              <input   type="text"   class="form-control"
id="idConfiguracion"  name="idConfiguracion"  value='<?php
echo $valores[0]["id"]?>'>
              <small   class="form-text">Estado   de   la
configuraciÃ³n.</small>
            </div>
             <div class="form-group">
              <label
for="idConfiguracion">#Identificador</label>
              <input   type="text"   class="form-control"
id="idIdentificador"  name="idIdentificador"  value='<?php
echo $valores[0]["nodo_identificador"]?>'>
              <small   class="form-text">Estado   de   la
configuraciÃ³n.</small>
            </div>
            <div class="form-group">
              <label for="message">NÃºmero</label>
              <input   type="text"   class="form-control"
id="idNumero"       name="idNumero"    value='<?php    echo
$valores[0]["nodo_numero"]?>'>
              <small     class="form-text">NÃºmero     de
nodo.</small>
            </div>
             <div class="form-group">
              <label for="message">Usuario</label>
               <input   type="text"   class="form-control"
id="idUsuario"    name="idUsuario"    value='<?php    echo
$valores[0]["nodo_usuario"]?>'>
              <small     class="form-text">Nombre     de
usuario.</small>
            </div>
              <div class="form-group">
              <label     for="message">Coordenadas     de
muestras</label>
               <input   type="text"   class="form-control"
id="idGps"     name="idGps"          value='<?php     echo
$valores[0]["nodo_gps"]?>'>
              <small class="form-text">Si existe Indicar
coordenadas.</small>
            </div>
               <div class="form-group">
              <label for="message">Fecha</label>
                <input  type="text"  class="form-control"
id="idFecha"    name="idFecha"        value='<?php    echo
$valores[0]["nodo_fecha"]?>'>
              <small              class="form-text">Fecha
AutomÃ¡tica</small>
            </div>
```

```
            <button type="submit" class="btn btn-primary
btn-lg   btn-rounded   btn-effect   btn-shadow   float-
right">Configurar</button>
          </form>

        </div>
      </div>
    </div>
  </section>


  <!-- /main -->
<?php require 'footer.php'; ?>
```

Código PHP-HTML(Controlador)

```
<?php
  $servername = "localhost";
     $username = "root";
     $password = "root";
     $dbname = "data_sensores";

        function obtenerDatos(){
    global $servername,$username,$password,$dbname;
    $conn   =   mysqli_connect($servername,   $username,
$password, $dbname);
    if($conn->errno){die("Error  conexion  "  .  $conn-
>error);}
    $sql="select * from datosnodo ORDER by ID DESC Limit
1";
    //$sql="SELECT    *    FROM    `rev_imagen`    where
rev_imagen_codigo=400";
    $res=$conn->query($sql);
     $datos=array();
    while ($fila=$res->fetch_assoc()){
        $datos[]=$fila;
    }

    return $datos;
    $conn->close();
        }

       function
configurarNodo($nodo_identificador,$nodo_numero,$nodo_usu
ario,$nodo_gps){
      global $servername,$username,$password,$dbname;
      $conn   =   mysqli_connect($servername,   $username,
$password, $dbname);
      if($conn->errno){die("Error  conexion  "  .  $conn-
>error);}
       $sql="INSERT INTO datosnodo (nodo_identificador,
nodo_numero, nodo_usuario, nodo_gps, nodo_fecha) VALUES
('$nodo_identificador',$nodo_numero,'$nodo_usuario','$nod
o_gps',SYSDATE())";
```

```
      //$sql="SELECT    *    FROM    `rev_imagen`    where
rev_imagen_codigo=400";
      $res=$conn->query($sql);
      $conn->close();
        }
          function obtenerDatosCalibracion(){
    global $servername,$username,$password,$dbname;
    $conn    =    mysqli_connect($servername,    $username,
$password, $dbname);
    if($conn->errno){die("Error   conexion   "   .   $conn-
>error);}
    $sql="select * from datoscalibracion ORDER by ID DESC
Limit 1";
    //$sql="SELECT    *    FROM    `rev_imagen`    where
rev_imagen_codigo=400";
    $res=$conn->query($sql);
     $datos=array();
    while ($fila=$res->fetch_assoc()){
        $datos[]=$fila;
    }

    return $datos;
    $conn->close();
        }

       function
configurarNodoCalibracion($nodo_valor,$nodo_usuario){
      global $servername,$username,$password,$dbname;
      $conn    =    mysqli_connect($servername,    $username,
$password, $dbname);
      if($conn->errno){die("Error   conexion   "   .   $conn-
>error);}
       $sql="INSERT         INTO         datoscalibracion
(cal_valor,cal_usuario,cal_fecha)                    VALUES
('$nodo_valor','$nodo_usuario',SYSDATE())";
      $res=$conn->query($sql);
      $conn->close();
        }
```

Manual Pasarela (Anexo2)

Resumen

Reporte técnico para la implementación de una pasarela que permita la recepción de los datos, además del procesamiento, almacenamiento e utilización de servicios externos para la comunicación con infraestructuras externas.
Para el prototipo se necesitó una palca Raspberry Pi 3 (RPi3) y una antena Xbee Digi Mesh, el software utilizado se realizó con el lenguaje de programación Python.
Red de sensores

- Xbee serie 1
- Configuración de la red

Procesamiento

RPi3

Materiales y métodos

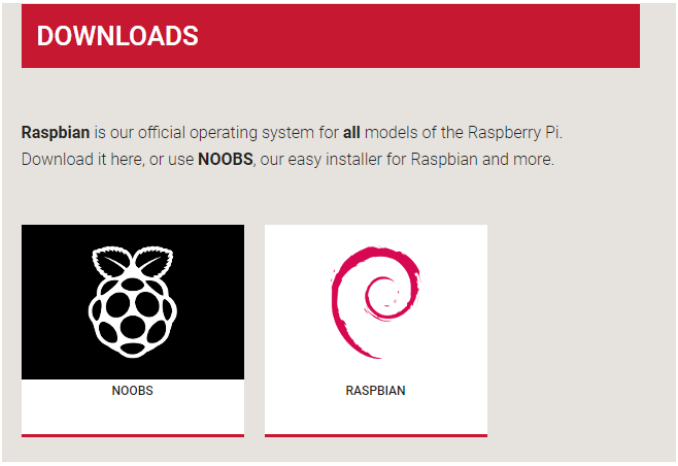
- 2 Xbee Digi Mesh Serie 1
- RPi3

Experimentación

Instalación del Raspberry Pi3

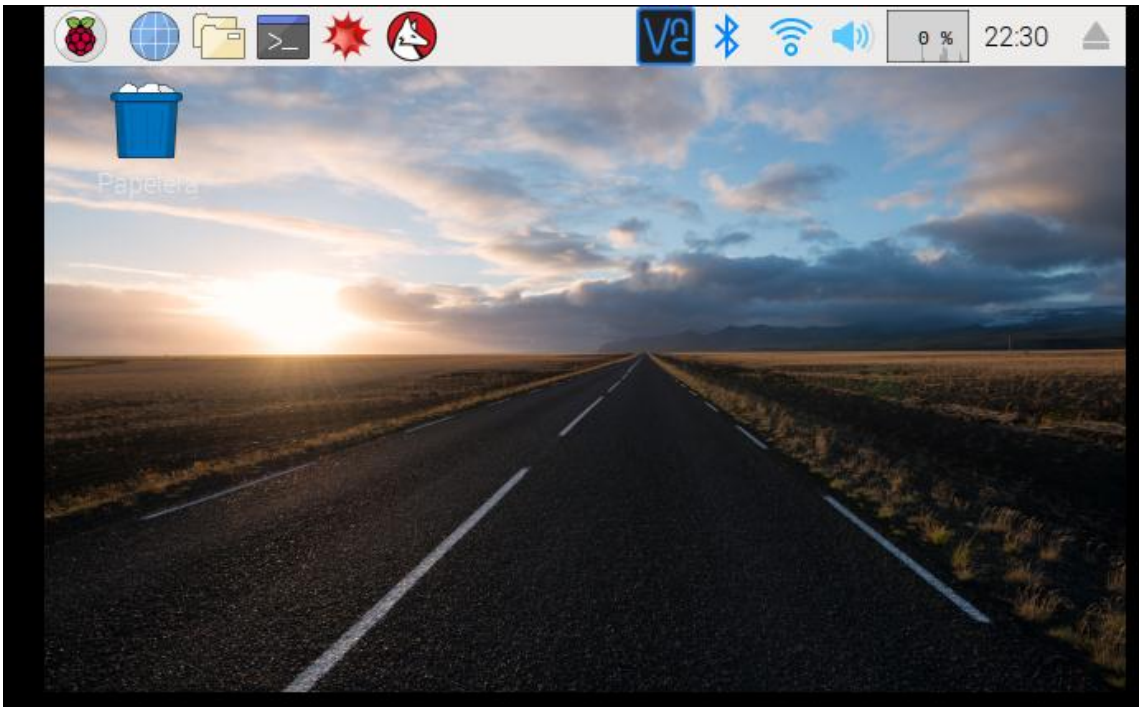
El proceso de instalación del RPI3 se detalla a continuación.

1. Ingresar a la pagina https://www.raspberrypi.org/downloads/
2. Seleccionar el S.O Raspbian



3. Crear una micro SD Booteable con el SO descargado, para este proceso se utilizó el programa Win32DiskImage

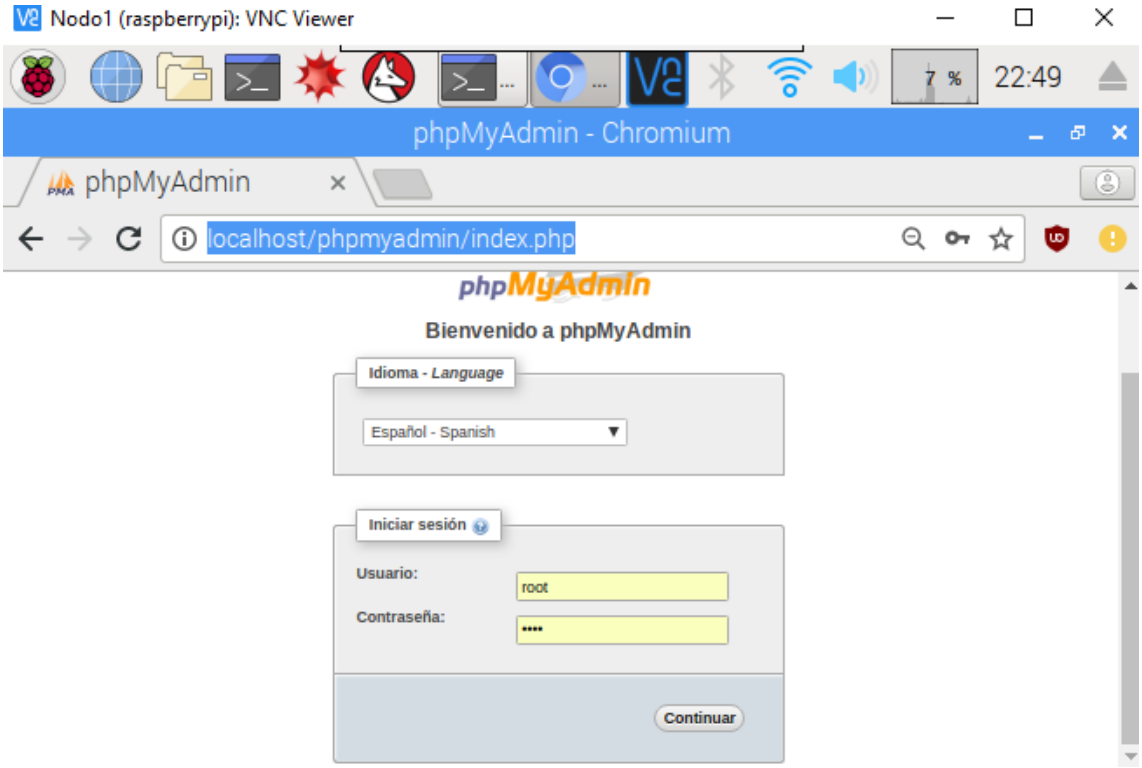
4. AL insertar la tarjeta en l aplaca RPi3 se conecta con una fuente de alimentación, lo que iniciara el dispositivo.



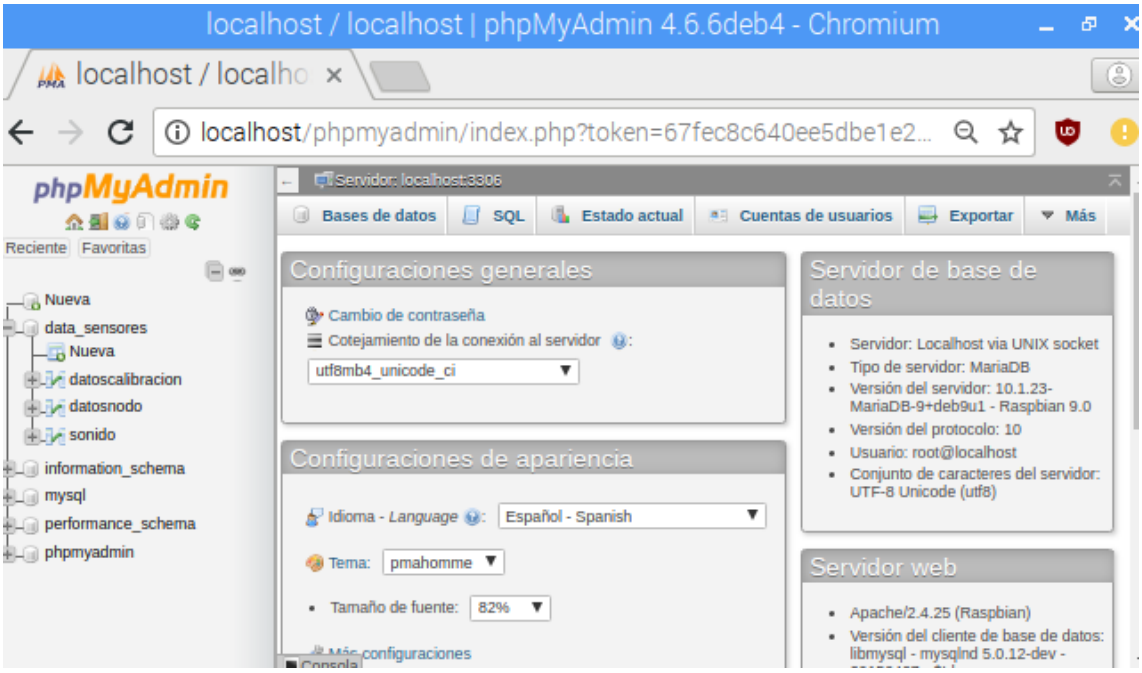
5. Instalación de Paquetes para la configuración del Nodo

```
 1  sudo apt-get install python-mysql.connector
 2  sudo apt-get install python-mysqldb
 3  sudo apt-get install python-mysql-client
 4  sudo apt-get install python-mysqldb
 5  sudo apt-get install pymysql
 6  pip install PyMySQL
 7  sudo apt-get install python3-pymysql
 8  sudo cat /dev/ttyACM0
 9  sudo cat /dev/ttyS0
10  cd ..
11  cd /home/pi/Downloads/
55  cd ..
56  cd pi/Documents/
57  mv nodo ../../../var/www/html/
58  sudo mv nodo ../../../var/www/html/
```

6. AL tener instalado los paquetes, podemos configurar el phpmyadmin con la siguiente url: http://localhost/phpmyadmin/index.php



7. Ingresar con la contraseña e usuario root.



8. Ejecutar el Scritp a continuación

```
-- phpMyAdmin SQL Dump
-- version 4.6.6deb4
-- https://www.phpmyadmin.net/
--
-- Servidor: localhost:3306
-- Tiempo de generaciÃ³n: 25-09-2018 a las 00:12:01
-- VersiÃ³n del servidor: 10.1.23-MariaDB-9+deb9u1
-- VersiÃ³n de PHP: 7.0.30-0+deb9u1

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
```

```
SET time_zone = "+00:00";


/*!40101                                                  SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101                                                  SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101                                                  SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de datos: `data_sensores`
--
CREATE DATABASE IF NOT EXISTS `data_sensores` DEFAULT
CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
USE `data_sensores`;

-- ------------------------------------------------------
--


--
-- Estructura de tabla para la tabla `datoscalibracion`

--
-- Estructura de tabla para la tabla `datosnodo`
--

CREATE TABLE `datosnodo` (
  `id` int(11) NOT NULL,
  `nodo_identificador` varchar(200) NOT NULL,
  `nodo_numero` int(11) NOT NULL,
  `nodo_usuario` varchar(200) NOT NULL,
  `nodo_gps` varchar(100) NOT NULL,
  `nodo_fecha` date NOT NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

-- ------------------------------------------------------
--


--
-- Estructura de tabla para la tabla `sonido`
--

CREATE TABLE `sonido` (
  `son_id` int(11) NOT NULL,
  `son_nodo` varchar(10) NOT NULL,
  `son_codigo` varchar(50) NOT NULL,
  `son_llegada` int(11) NOT NULL,
  `son_coordenadas` varchar(150) NOT NULL,
  `son_valor` varchar(10) NOT NULL,
  `son_observaciones` varchar(200) NOT NULL,
  `son_fecha` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--
ALTER TABLE `sonido`
  ADD PRIMARY KEY (`son_id`);

--
-- AUTO_INCREMENT de las tablas volcadas
--
-- AUTO_INCREMENT de la tabla `datosnodo`
--
ALTER TABLE `datosnodo`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
--
-- AUTO_INCREMENT de la tabla `sonido`
--
ALTER TABLE `sonido`
  MODIFY `son_id` int(11) NOT NULL AUTO_INCREMENT;--
-- Base de datos: `phpmyadmin`
--
CREATE DATABASE IF NOT EXISTS `phpmyadmin` DEFAULT
CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
USE `phpmyadmin`;
```

11 Programación del software en Python

Script Principal

```
import serial
import threading
from funcionesBd import insertaDatos
from envioDatos import envioDatos
ser = serial.Serial('/dev/ttyUSB0',9600)
numNodos = 1
cadenaTotal = 7
#Autocalibracion del buufer serial para el ingreso de los
datos
data = ser.read(1000)
data = str(data)
datosValidar = data.split("<=>#")
for unDatosValidar in datosValidar:
    if("#" in unDatosValidar):
        cadenaValida = unDatosValidar.split("#")
datosValidar.pop(0)
datosValidar.pop(len(datosValidar)-1)
datosValidar = set(datosValidar)
cadenaRecibidaDatos = 0
for unDato in datosValidar:
    valor = unDato.split("#")
    cadenaRecibidaDatos    =    cadenaRecibidaDatos    +
int(valor[0])+7
#print(cadenaRecibidaDatos)

data = ser.read(cadenaRecibidaDatos)
data = str(data)
val=data.find("%")
data = ser.read(cadenaRecibidaDatos)
```

```
data = ser.read(val-1)
data = str(data)
#Estructuracion de los datos para poder ser almacendados
y enviados
while True:
    gps = False
    data = ser.read(cadenaRecibidaDatos)
    data = str(data)
    cadenaDatos=data.split("<=>#")
    for unaCadena in cadenaDatos:
        if(len(unaCadena)>6):
            if("GPS" in unaCadena.upper()):
               gps = True
            unaCadenaLimpia = unaCadena.split("#")
            if 6 == len(unaCadenaLimpia):
                #hilo1Bd                                    =
threading.Thread(target=insertaDatos,
args=(unaCadenaLimpia,gps,))
                #hilo1Bd.start()
                insertaDatos(unaCadenaLimpia,gps)
                hilo1Http                                   =
threading.Thread(target=envioDatos,
args=(unaCadenaLimpia,True,))
                hilo1Http.start()
                #envioDatos(unaCadenaLimpia,True)
            else:
                print("error")

#primera lectura con el valor del buffer Calibrado
```

Envio Datos

```
import requests

def envioDatos(datos,gps):
        try:
            valorNodo
=datos[2][(datos[2].find('n')+1):len(datos[2])]
            valorSonido                                     =
datos[5][(datos[5].find(':')+1):datos[5].find('%')]
            cambio=int(valorNodo)
            nodo = 'field'+str(cambio)+'='+valorSonido

cadena='https://api.thingspeak.com/update?api_key=B3D46ZY
BXBJVBPJB&'+nodo
            r = requests.get(cadena)
            #print(cadena)
        finally:
            pass
```

Conexión

```
import pymysql.cursors
connection = pymysql.connect(host='localhost',
                        port=3306,
                        user='root',
                        passwd='root',
                        db='data_sensores',
                        autocommit=True)

def insertaDatos(unaCadenaLimpia,gps):
    #print (unaCadenaLimpia)
    if (gps):
        try:
            with connection.cursor() as cursor:
                sql         =         "INSERT         INTO
sonido(son_codigo,son_nodo,son_llegada,son_coordenadas,so
n_valor,son_observaciones) VALUES (%s,%s,%s,%s,%s,%s)"
                cursor.execute(sql,(
                    unaCadenaLimpia[1],
                    unaCadenaLimpia[2],
                    unaCadenaLimpia[3],
                    unaCadenaLimpia[4],
                    unaCadenaLimpia[5],
                    'Ninguna'))
        finally:
            pass
    else:
        try:
            with connection.cursor() as cursor:
                sql         =         "INSERT         INTO
sonido(son_codigo,son_nodo,son_llegada,son_valor,son_obse
rvaciones) VALUES (%s,%s,%s,%s,%s,%s)"
                cursor.execute(sql,(
                    unaCadenaLimpia[1],
                    unaCadenaLimpia[2],
                    unaCadenaLimpia[3],
                    unaCadenaLimpia[4],
                    'Ninguna'))
        finally:
            pass
```

12 Programación del servidor web

Código PHP-HTML(Índex)

```
<?php require 'head.php'?>
  <br>
<center><img src="imagenes/portada.png"></center>
<?php              require              'controlador.php';
$valores=obtenerDatos();?>
  <section class="p-b-0">
    <div class="container">
      <div class="heading">
        <h2>Bienvenido | Configuraciones NODO</h2>
```

```
        <p>En esta secciÃ³n nos permite la configuracion
del nodo ademas de establecer el parametro de calibraciÃ³n
</p>
      </div>
    </div>
  </section>

  <section class="p-t-10">
    <div class="container">
      <div class="row">
        <div class="col-lg-7 mx-auto">

          <?php if(isset($_GET['core'])){?>
             <div   class="alert   alert-success   alert-
dismissible" role="alert">
              <button  type="button"  class="close"  data-
dismiss="alert" aria-label="Close">
              <span aria-hidden="true">&times;</span>
              </button>
              <b>CORRECTO!</b> Modificaciones Correctas.
            </div>
            <?php } ?>
          <form          id="form1"          method="POST"
action="procesos.php">
            <input   type="hidden"   class="form-control"
id="configuracion" name="configuracion">
            <div class="form-group">
              <label
for="idConfiguracion">#ConfiguraciÃ³n</label>
              <input   type="text"   class="form-control"
id="idConfiguracion"  name="idConfiguracion"  value='<?php
echo $valores[0]["id"]?>'>
              <small   class="form-text">Estado   de   la
configuraciÃ³n.</small>
            </div>
             <div class="form-group">
              <label
for="idConfiguracion">#Identificador</label>
              <input   type="text"   class="form-control"
id="idIdentificador"  name="idIdentificador"  value='<?php
echo $valores[0]["nodo_identificador"]?>'>
              <small   class="form-text">Estado   de   la
configuraciÃ³n.</small>
            </div>
            <div class="form-group">
              <label for="message">NÃºmero</label>
              <input   type="text"   class="form-control"
id="idNumero"       name="idNumero"    value='<?php   echo
$valores[0]["nodo_numero"]?>'>
              <small     class="form-text">NÃºmero     de
nodo.</small>
            </div>
             <div class="form-group">
              <label for="message">Usuario</label>
```

```
                <input   type="text"   class="form-control"
id="idUsuario"    name="idUsuario"    value='<?php    echo
$valores[0]["nodo_usuario"]?>'>
                <small     class="form-text">Nombre      de
usuario.</small>
            </div>
              <div class="form-group">
              <label     for="message">Coordenadas     de
muestras</label>
                <input   type="text"   class="form-control"
id="idGps"     name="idGps"          value='<?php     echo
$valores[0]["nodo_gps"]?>'>
              <small class="form-text">Si existe Indicar
coordenadas.</small>
            </div>
               <div class="form-group">
              <label for="message">Fecha</label>
                <input  type="text"  class="form-control"
id="idFecha"    name="idFecha"        value='<?php    echo
$valores[0]["nodo_fecha"]?>'>
              <small              class="form-text">Fecha
AutomÃ¡tica</small>
            </div>
            <button  type="submit"  class="btn  btn-primary
btn-lg    btn-rounded    btn-effect    btn-shadow    float-
right">Configurar</button>
          </form>

        </div>
      </div>
    </div>
  </section>


  <!-- /main -->
<?php require 'footer.php'; ?>
```

Código PHP-HTML(Controlador)

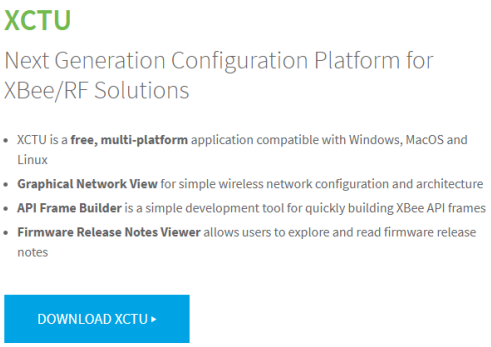
```
<?php
  $servername = "localhost";
      $username = "root";
      $password = "root";
      $dbname = "data_sensores";

         function obtenerDatos(){
    global $servername,$username,$password,$dbname;
    $conn    =    mysqli_connect($servername,    $username,
$password, $dbname);
    if($conn->errno){die("Error   conexion   "   .   $conn-
>error);}
    $sql="select * from datosnodo ORDER by ID DESC Limit
1";
    //$sql="SELECT    *    FROM    `rev_imagen`    where
rev_imagen_codigo=400";
```

```
    $res=$conn->query($sql);
     $datos=array();
    while ($fila=$res->fetch_assoc()){
        $datos[]=$fila;
    }

    return $datos;
    $conn->close();
        }

       function
configurarNodo($nodo_identificador,$nodo_numero,$nodo_usu
ario,$nodo_gps){
      global $servername,$username,$password,$dbname;
      $conn   =   mysqli_connect($servername,   $username,
$password, $dbname);
      if($conn->errno){die("Error  conexion  "  .  $conn-
>error);}
       $sql="INSERT  INTO  datosnodo  (nodo_identificador,
nodo_numero, nodo_usuario, nodo_gps, nodo_fecha) VALUES
('$nodo_identificador',$nodo_numero,'$nodo_usuario','$nod
o_gps',SYSDATE())";
      //$sql="SELECT    *    FROM    `rev_imagen`    where
rev_imagen_codigo=400";
      $res=$conn->query($sql);
      $conn->close();
        }
          function obtenerDatosCalibracion(){
    global $servername,$username,$password,$dbname;
    $conn    =    mysqli_connect($servername,    $username,
$password, $dbname);
    if($conn->errno){die("Error  conexion  "  .  $conn-
>error);}
    $sql="select * from datoscalibracion ORDER by ID DESC
Limit 1";
    //$sql="SELECT    *    FROM    `rev_imagen`    where
rev_imagen_codigo=400";
    $res=$conn->query($sql);
     $datos=array();
    while ($fila=$res->fetch_assoc()){
        $datos[]=$fila;
    }

    return $datos;
    $conn->close();
        }

       function
configurarNodoCalibracion($nodo_valor,$nodo_usuario){
      global $servername,$username,$password,$dbname;
      $conn   =   mysqli_connect($servername,   $username,
$password, $dbname);
      if($conn->errno){die("Error  conexion  "  .  $conn-
>error);}
```

```
        $sql="INSERT          INTO          datoscalibracion
(cal_valor,cal_usuario,cal_fecha)                    VALUES
('$nodo_valor','$nodo_usuario',SYSDATE())";
      $res=$conn->query($sql);
      $conn->close();
        }
```

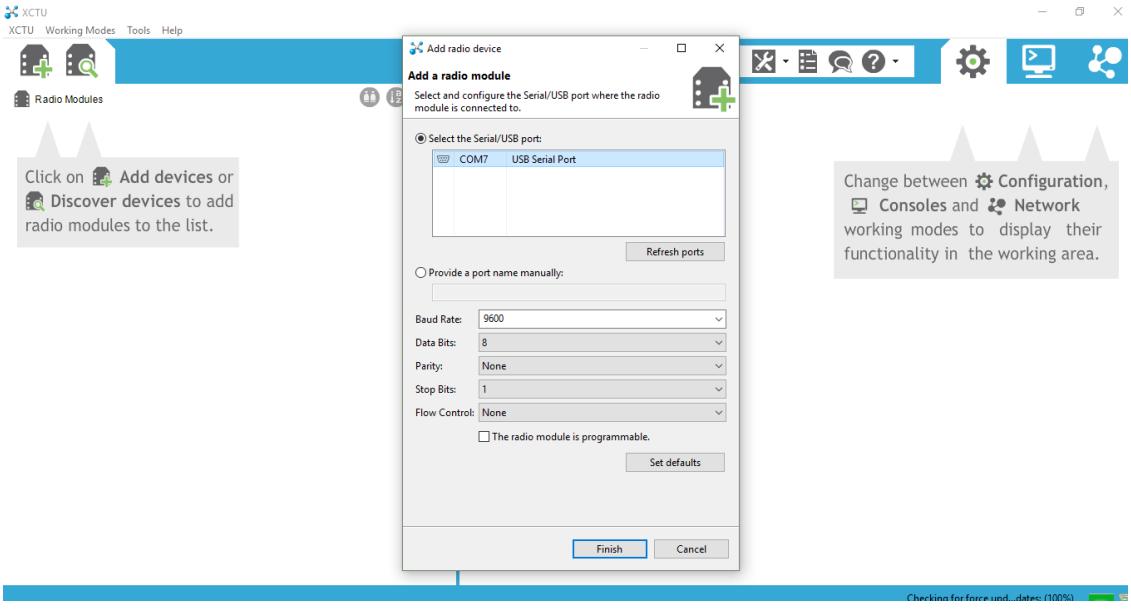
13 Descargar el programa XCTU

Para descargar el programa ingresar al siguiente LINK: https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu

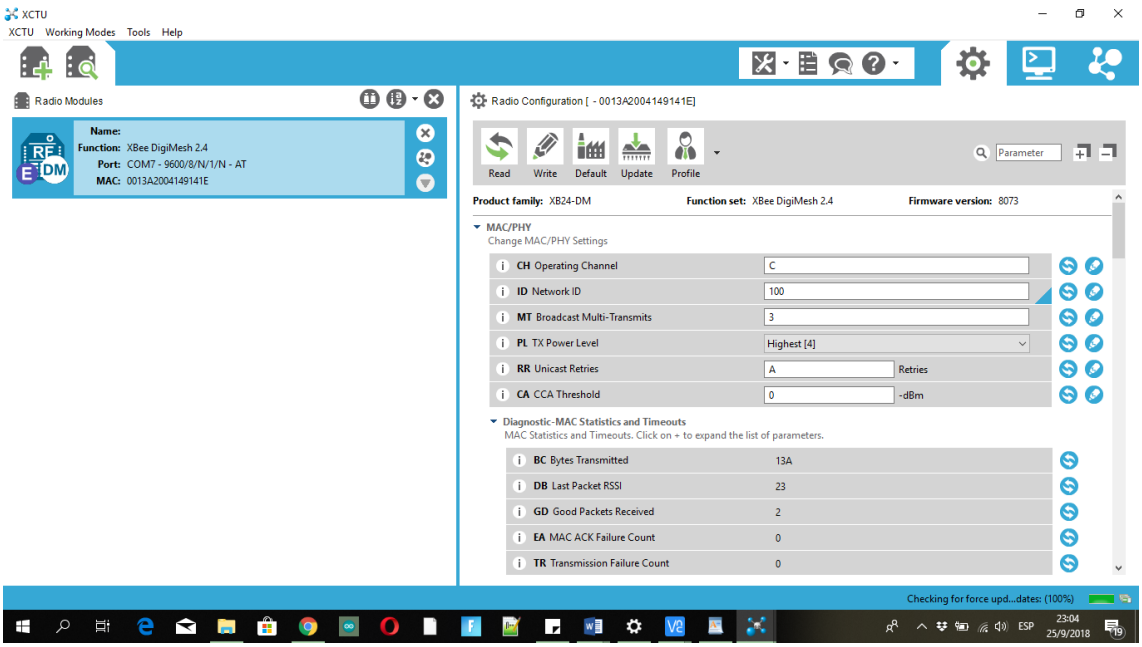


14 Conectar módulo XBee

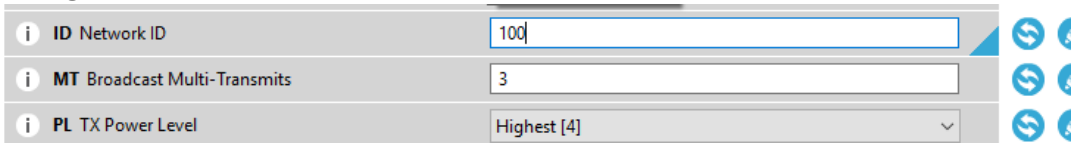
15 Iniciar el programa XCTU



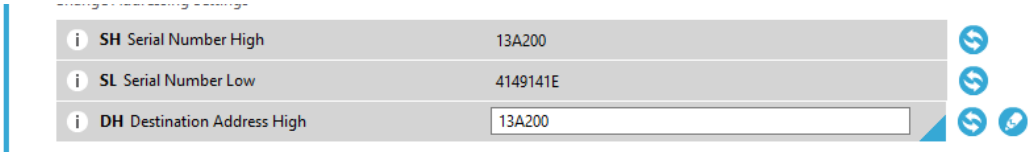
16 Configuración del Firmware



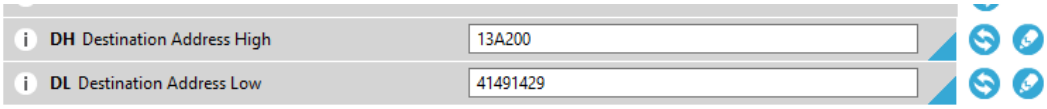
17 Configuración del ID de la red



18 Configuración SH de la red



19 Configuración del DL



Doctora María Elena Ramírez Aguilar, Secretaria de la Facultad de Ciencias de la Administración de la Universidad del Azuay

C E R T I F I C A:

Que, el Consejo de Facultad en sesión del 11 de mayo de 2018, conoció y aprobó la solicitud para realización del trabajo de titulación, presentada por :

Estudiante: David Gerardo Mejía Saca (cód. 68045)
Tema: **"Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la transmisión inalámbrica de datos"**
Previo a la obtención del título de Ingeniero de Sistemas y Telemática
Director: Ing. Gabriel Barros
Tribunal: Ing. Diego Chacón
Ing. Oswaldo Merchán

Plazo de presentación del trabajo de titulación, con la calificación del Director: seis meses a partir de la fecha de aprobación, esto es hasta el 11 de noviembre de 2018.

E INFORMA:

Que en aplicación de la Disposición General Cuarta del Reglamento de Régimen Académico vigente, en caso de que el estudiante no culmine y apruebe el trabajo de titulación luego de dos períodos académicos contados a partir de la fecha de culminación de estudios, deberá realizar la actualización de conocimientos previa a su titulación.

Cuenca, 14 de mayo de 2018

UNIVERSIDAD DEL AZUAY | Ciencias de la Administración Facultad
Dra. María Elena Ramírez Aguilar
Secretaria - Abogada

CONVOCATORIA

Por disposición de la Junta Académica de la escuela de Ingeniería de Sistemas y Telemática se convoca a los Miembros del Tribunal Examinador, a la sustentación del Protocolo del Trabajo de Titulación: **"Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la transmisión inalámbrica de datos con autonomía energética",** presentado por el estudiante David Gerardo Mejía Saca con código 68045, previa a la obtención del título de Ingeniero de Sistemas y Telemática, para el día **Miércoles, 09 de mayo de 2018 a las 08:40.**

Tomar en cuenta que posterior a la sustentación del Diseño del Trabajo de Titulación, por ningún concepto se puede realizar modificaciones ni cambios en los documentos; únicamente, en caso de diseño aprobado con modificación, el Director adjuntará al esquema un oficio indicando que se procede con los cambios sugeridos.

Cuenca, 07 de mayo de 2018

Dra. María Elena Ramírez Aguilar
Secretaria de la Facultad

Ing. Gabriel Barros .....

Ing. Diego Chacón .....

Ing. Oswaldo Merchán .....

ESCUELA DE INGENIERÍA DE SISTEMAS Y TELEMATICA

FECHA: **07 de MAYO DE 2018.**

Estudiante: David Gerardo Mejía Saca

Oficio Nro. 017-2018-DIST-UDA

Cuenca, 2 de mayo de 2018

Señor Ingeniero
Oswaldo Merchán Manzano
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN
Presente.-

De nuestras consideraciones:

La Junta Académica de la Escuela de Ingeniería de Sistemas y Telemática, reunida el día 2 de mayo del 2018, recibió el proyecto de tesis titulado "Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la transmisión inalámbrica de datos con autonomía energética", presentado por David Gerardo Mejía Saca estudiante de la Escuela de Ingeniería de Sistemas y Telemática, y revisado por el Ing. Gabriel Barros Ph.D. previo a la obtención del título de Ingeniero de Sistemas y Telemática.

Por lo expuesto, y de conformidad con el Reglamento de Graduación de la Facultad, recomendamos como director y responsable de aplicar cualquier modificación al diseño del trabajo de graduación posterior al Ing. Gabriel Barros Ph.D., como co-director y miembro de tribubal al Ing. Diego Chacón, y como miembro de tribunal al Ing. Oswaldo Merchán.

Atentamente,

Ing. Marcos Orellana Cordero
Coordinador Escuela de Ingeniería de Sistemas y Telemática
Universidad del Azuay

ACTA
SUSTENTACIÓN DE PROTOCOLO/DENUNCIA DEL TRABAJO DE TITULACIÓN

Fecha de sustentación: **Miércoles, 09 de mayo de 2018 a las 08:40**

1.1. Nombre del estudiante: David Gerardo Mejía Saca
1.2. Código: 68045
1.3. Director sugerido: Ing. Gabriel Barros
1.4. Codirector (opcional): _____
1.4.1. Tribunal: Ing. Diego Chacón e Ing. Oswaldo Merchán
1.4.2. Título propuesto: **"Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la transmisión inalámbrica de datos con autonomía energética"**
1.4.3. Aceptado sin modificaciones :

1.4.4. Aceptado con las siguientes modificaciones:

Cambian en el título y retiran la frase "con autonomía energética"

1.4.5. No aceptado
1.4.6. Justificación:

Tribunal

Ing. Gabriel Barros

Ing. Diego Chacón

Ing. Oswaldo Merchán

Sr. David Gerardo Mejía Saca

Dra. María Elena Ramírez Aguilar
Secretaria de la Facultad

RÚBRICA PARA LA EVALUACIÓN DEL PROTOCOLO DE TRABAJO DE TITULACIÓN
(Tribunal)

1.1. Nombre del estudiante: David Gerardo Mejía Saca
1.2. Código : 68045
1.3. Director sugerido:
Codirector (opcional): Ing. Gabriel Barros
1.3.1. Título propuesto: **"Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la transmisión inalámbrica de datos con autonomía energética"**
1.4. Revisores tribunal: Ing. Diego Chacón e Ing. Oswaldo Merchán
1.5. Recomendaciones generales de la revisión:

	Cumple	No cumple
Problemática y/o pregunta de investigación	✓	
1. ¿Presenta una descripción precisa y clara?	✓	
2. ¿Tiene relevancia profesional y social?		
Objetivo general		
3. ¿Concuerda con el problema formulado?	✓	
4. ¿Se encuentra redactado en tiempo verbal infinitivo?	✓	
Objetivos específicos		
5. ¿Permiten cumplir con el objetivo general?	✓	
6. ¿Son comprobables cualitativa o cuantitativamente?	✓	
Metodología		
7. ¿Se encuentran disponibles los datos y materiales mencionados?	✓	
8. ¿Las actividades se presentan siguiendo una secuencia lógica?	✓	
9. ¿Las actividades permitirán la consecución de los objetivos específicos planteados?	✓	
10. ¿Las técnicas planteadas están de acuerdo con el tipo de investigación?	✓	
Resultados esperados		
11. ¿Son relevantes para resolver o contribuir con el problema formulado?	✓	
12. ¿Concuerdan con los objetivos específicos?	✓	
13. ¿Se detalla la forma de presentación de los resultados?	✓	
14. ¿Los resultados esperados son consecuencia, en todos los casos, de las actividades mencionadas?	✓	

Ing. Gabriel Barros

Ing. Diego Chacón

Ing. Oswaldo Merchán

Cuenca, 10 de mayo de 2018

Ingeniero
Oswaldo Merchán Manzano
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN
UNIVERSIDAD DEL AZUAY

De mi consideración:

Por medio de la presente me permito dirigirme a Ud. con el propósito de comunicarle que, luego de la sustentación del tema de tesis: "Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la trasmisión inalámbrica de datos.", certifico que se han realzado los cambios solicitados por el tribunal.

Por la favorable acogida que se sirva dar a la presente, anticipo mis agradecimientos.

Atentamente

Ing. Gabriel Barros.

Director Sugerido

Cuenca, 4 de mayo de 2018

Ingeniero
Oswaldo Merchán Manzano
DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN
UNIVERSIDAD DEL AZUAY

De mi consideración:

Por medio de la presente me permito dirigirme a Ud. con el propósito de comunicarle que, una vez revisado el diseño de tesis previa a la obtención del título de Ingeniero de Sistemas y Telemática del estudiante: David Gerardo Mejía Saca, cuyo tema es: "Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la trasmisión inalámbrica de datos con autonomía energética", recomendar la aprobación del mismo.

Por la favorable acogida que se sirva dar a la presente, anticipo mis agradecimientos.

Atentamente

Ing. Gabriel Barros.

Director Sugerido

UNIVERSIDAD DEL AZUAY

DOCTORA MARÍA ELENA RAMÍREZ AGUILAR, SECRETARIA DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACIÓN DE LA UNIVERSIDAD DEL AZUAY

C E R T I F I C A:

Que, el señor **MEJIA SACA DAVID GERARDO** con código **68045,** alumno de la carrera de **INGENIERIA DE SISTEMAS Y TELEMATICA,** tiene aprobado el **92,27%** de créditos de su malla curricular.

Cuenca, 02 de mayo de 2018

Dra. María Elena Ramírez Aguilar
SECRETARIA DE LA FACULTAD
DE CIENCIAS DE LA ADMINISTRACIÓN

UNIVERSIDAD DEL AZUAY
FACULTAD DE ADMINISTRACION
SECRETARIA

Derecho No. **001-001-000171877**
mjmr.-

No 0842147

UNIVERSIDAD DEL AZUAY

Cuenca 4 de mayo de 2018

Ingeniero,

Oswaldo Merchán Manzano

DECANO DE LA FACULTAD DE CIENCIAS DE LA ADMINISTRACION

UNIVERSIDAD DEL AZUAY

De mi consideración,

Estimado Señor Decano, yo **David Gerardo Mejía Saca** con C.I. **0106625437**, código estudiantil **68045** estudiante de la carrera de Ingeniería Sistemas y Telemática, solicitamos muy comedidamente a usted y por su intermedio al Consejo de Facultad, la aprobación del protocolo de trabajo de titulación con el tema "Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la trasmisión inalámbrica de datos con autonomía energética." previo a la obtención del titulo de Ingeniero en Sistemas y Telemática para lo cual adjuntamos la documentación respectiva.

Por la favorable acogida que brinde a la presente, anticipamos nuestros agradecimientos.

Atentamente

David Gerardo Mejía Saca

Estudiante de la Carrera de Sistemas y Telemática

UNIVERSIDAD DEL AZUAY

UNIVERSIDAD DEL AZUAY
INGENIERÍA EN SISTEMAS Y TELEMÁTICA
DISEÑO DE TESIS

1. DATOS GENERALES

1.1 Nombre del estudiante: David Gerardo Mejía Saca

1.1.1 Código:68045

1.1.2 Contacto:

Teléfono: 0996008330

Convencional: 072806920

Correo electrónico: davidgerardo15@hotmail.com

1.3Director sugerido: Ing. Gabriel Barros

1.2.1 Contacto:

Convencional: 072817320

Correo electrónico: gbarrosg@uazuay.edu.ec

1.3 Co-director sugerido: Ing. Diego Chacón.

Convencional: 072869001

Correo electrónico: dchacon@uazuay.edu.ec

1.4 Asesor metodológico:

1.5 Tribunal designado:

1.6 Aprobación:

1.7 Línea de Investigación de la carrera:

1.7.1 Código UNESCO: 1203 Informática de Computadores.

1.7.2 Tipo de trabajo: Investigación

1.8 Área de estudio: Sistemas informáticos y redes de comunicaciones.

1.9 Título propuesto: Implementación, calibración y evaluación de un sistema de medición de ruido con un sensor de sonido que permita la trasmisión inalámbrica de datos.

1.10 Estado del proyecto: El estudio de la contaminación auditiva se realiza con sonómetros que cumplen las normativas IEC 61672, los mismos que tienen costos de hasta los $5000; en la actualidad las redes de sensores inalámbricos(WSN) permiten las transmisión de los datos de sensores económicos lo que permite tener un sistema de bajo costo. El proyecto se fundamenta en la implementación, calibración y evaluación de un sistema que permita obtener los niveles de ruido con tecnología de bajo costo. Los resultados que se obtengan de esta evaluación, determinarán si se puede utilizar dispositivos económicos para el estudio y análisis de la contaminación auditiva.

2. CONTENIDO

2.1 Motivación de la investigación:

Los análisis de la contaminación auditiva en la ciudad de Cuenca son presentados cada año, los mismos que no permiten conocer de forma real el estado o los niveles de ruido que la ciudad o ciudadanía soporta a diferentes horas y días, por tal motivo la importancia de la presente investigación se fundamenta en la implementación, de un sistema que mida los niveles de ruido de manera continua, generando datos en periodos reales y cortos de tiempo que permitan evidenciar los niveles de ruido.

2.2 Problemática:

La sociedad vive una etapa muy acelerada, la cual hace que no se le dé la importancia necesaria a contaminantes a los cuales se ha acostumbrado, siendo este el caso el ruido, el cual causa una contaminación auditiva que produce daños psicológicos y físicos, existen investigaciones acerca de este tema que toman como base un período mínimo de un año pero para

realizarlos se necesita sonómetros certificados y el recurso humano que pueda realizar la toma de muestras, el problema con el recurso humano es que hacen que las campañas de medición sean cortas en duración y en espacios muy largos de tiempo, mientras que un sistema automatizado permitirá tener mediciones mucho más frecuentes en períodos mucho más cortos sin necesidad de muchos recursos.

2.3 Pregunta de investigación:

¿Puede un sistema desarrollado con equipos existentes en el mercado, obtener datos con la calidad al menos igual a la de un sonómetro tipo III en el contexto de una red de sensores inalámbrica que mide el ruido y reportarlos a un servidor remoto?

2.4 Resumen:

El tema a desarrollar consiste en la implementación de un sistema que contiene un sensor de sonido, el cual permita la recolección y calibración de los datos obtenidos, siendo estos transmitidos en el menor tiempo posible desde el nodo sensor de manera continua, por medio de una red WSN que utiliza la tecnología basada en el estándar IEEE 802.15.4, hasta llegar a una pasarela que envié esta información a la nube.

Esta infraestructura será realizada con dispositivos de bajo costo, lo cual permitirá comparar los datos resultantes de hardware y software para que puedan ser comparados con resultados de investigaciones que se han obtenido dentro de la ciudad de Cuenca, los cuales determinarán la factibilidad del uso de este nuevo sistema, permitiendo tomar decisiones relevantes acerca del estudio de la contaminación auditiva.

2.5 Indagación Exploratoria:

En la actualidad, el ruido se ha convertido en un contaminante auditivo que está comenzando a causar daños en la salud de los seres humanos (Sevillano et al., 2016). Ejemplos de

afectaciones a la salud son la progresiva pérdida de la audición, alteraciones en la presión arterial, ritmo cardiaco, también insomnio, cefaleas crónicas, y reducción de la capacidad sexual (Alonso, 2003). Entre los aspectos psicológicos constan el aumento del estrés e irritabilidad. Es por eso que recomienda como límite máximo un nivel sonoro de 80dB (Sanz & García, 2003).

Esta problemática viene dado por el acelerado crecimiento y desarrollo descontrolado de las ciudades, pues el ruido es todo sonido indeseable que afecta o perjudica a las personas (Alfie Cohen & Salinas Castillo, 2017). En este sentido, ha sido documentado que para la población habitante de un centro urbano el ruido generado por el tráfico vehicular es la principal causa de molestia. Dada esta evidencia, instituciones como la Organización Mundial de la Salud (OMS) han definido a la contaminación auditiva como el tercer problema ambiental de mayor relevancia en el mundo. (Pacheco, 2009)

Para medir el nivel de contaminación del ruido hay que considerar 3 magnitudes que se encuentran relacionadas. La intensidad, que es el nivel este se encuentra ligado a la cantidad de energía empleada para generarlo, esta se mide en decibelios (dB), luego la frecuencia de exposición y por último la duración (Alonso, 2003).

El 70% de las emisiones sonoras provienen de los vehículos motorizados (Rana, Chou, Bulusu, Kanhere, & Hu, 2015), es por eso que es fundamental conocer que en la ciudad de Cuenca la empresa EMOV EP reveló que en el año 2014 el parque automotor del cantón Cuenca ascendió a 131 488 unidades (EMOV EP, Inventario de emisiones atmosféricas, 2014, p 21).

Para realizar el monitoreo adecuado de este contaminante, debemos considerar estos aspectos: hardware, costos, crecimiento de la red (escalabilidad), adaptación del sistema con otras tecnologías (flexibilidad), calidad de los datos (confiabilidad) y margen de error en los datos (precisión). Pero la inversión que se necesita para implementar toda la infraestructura que permita

obtener los mejores resultados es elevada, esto ha conllevado a buscar tecnologías más económicas que sustituyan a estos dispositivos (Wessels & Basten, 2016) .Es por eso que nace la iniciativa de generar estaciones de monitoreo a un costo reducido. Estos dispositivos pueden ser ubicados en diferentes zonas permitiendo realizar un monitoreo del ruido.

Los equipos utilizados para medir el sonido son los sonómetros, estos miden el nivel de presión sonora de un ruido en dB, en tiempo y espacio. Los sonómetros que se utilizan cumplen con las normas IEC (International Electrotechniacal Commision) 61672. En la norma IEC 61672-1 especifica 2 clases, clase1 utilizada en laboratorios o en lugares donde el ambiente acústico pueda ser controlado, clase 2 para mediciones generales (Caguano, 2016.), concentrando la investigación a la implementación de un clase 2.

Sevillano et al (2016) presenta una investigación en la que divide en dos grupos diferentes a los sensores:

-Basada en una computadora integrada

-Basada en un microcontrolador

Los sistemas basados en computadores integrados son posibles gracias al crecimiento exponencial de la informática, las cuales se pueden encontrar en tamaños muy reducidos. Estos vienen equipados con placas de sonido, modem GPRS, 3G, 4G o WI-FI y un software que permita el análisis y la calibración de los datos entrantes. Por otro lado, los Sistemas basados en un microcontrolador tienen un mínimo consumo de energía ya que no cuentan con una serie de placas para la transmisión de información; varios sistemas de control que utilizan este tipo de tecnología cumplen con la clase II IEC 61672.

La generación de sensores de bajo costo ha creado un interés en el desarrollo de redes WSN (redes de sensores inalámbricas), los mismos que son mucho más económicos y que permiten la

transmisión de los datos en tiempo real sin la necesidad de utilizar cableado para conectar los nodos. Entre los protocolos más utilizados están: Bluetooth (sobre IEEE 802.15.1); UWB (sobre IEEE802.15.3), ZigBee (sobre IEEE 802.15.4) y Wi-Fi (sobre IEEE 802.11a / b / g)(Sánchez-Rosario et al., 2015).

Chang et al., (2015) en parte de su investigación enuncia que el protocolo ZigBee es el más valorado por la industria, ya que siendo de bajo costo y tolerando un bajo consumo de energía, puede tener múltiples características en velocidad y distancia para la creación de diferentes tipos redes tanto en transmisión como en topología, gracias a su fácil adaptación a pasarelas que tienen arquitecturas diferentes.

Los estudios acerca de la contaminación auditiva de bajo costo han comenzado a utilizar Smartphone, como los sensores que permiten la obtención del sonido, estos con los algoritmos de calibración han llegado a obtener resultados satisfactorios que han permitido obtener una idea general de cómo está los niveles de ruido en los lugares en donde fueron tomadas las muestras, cabe recalcar que estos resultados se encuentran con un error de 3 a 4 dB , dependiendo del método de calibración(G. Barros, 2015).

Sin embargo esta magnitud de error es demasiado alta, es por eso que este trabajo propone diseñar un sistema el cual buscará tener una autonomía de energía además de contar con un sensor de sonido, el cual con técnicas de calibración den resultados mucho más satisfactorios, los mismos que serán enviados a la nube por medio de una infraestructura de red sencilla de levantar.

Objetivo general:

Implementar, calibrar y evaluar un sistema de medición de ruido con un sensor de sonido que permita la trasmisión inalámbrica de datos con autonomía energética.

2.6 Objetivos específicos:

UNIVERSIDAD DEL AZUAY

1 Realizar la revisión de la literatura y proyectos similares para identificar, conceptualizar términos y herramientas utilizadas en la investigación del tema propuesto.

2 Implementar y calibrar un sistema para el análisis del ruido mediante un sensor de sonido

3 Implementar una red inalámbrica que permita la transmisión de datos.

4 Analizar y Comparar los resultados con otros estudios realizados en: costo, precisión, tiempo total de campaña, consumo de energía.

2.8 Metodología:

El estudio corresponde a un tipo a una investigación de tipo descriptivo-analítico, ya que se realiza dependiendo de los análisis y resultados que se obtengan de la implementación del sistema.

La recolección de los datos y su respectiva calibración se basa en una medición numérica y algoritmos de calibración el cual permitirá encontrar patrones correlativos que permitan dar un mejor enfoque a los resultados obtenidos.

La transmisión de datos se basa en una medición numérica de segundos de transmisión, tanto los cuales permitirán medir la ocurrencia y obtener una serie de detalles los cuales podrán ser comparados con otros tipos de investigaciones o tecnológicas.

2.9 Alcances y resultados esperados:

Al término del proyecto se pretende implementar un sistema de medición de ruido que busque tener un monitoreo continuo, el cual permita la recepción, calibración, y transmisión de los datos hacia el servidor, examinando la forma de tener autonomía con respecto a la energía utilizada, estos resultados determinarán la factibilidad del uso de este sistema, permitiendo tomar decisiones relevantes acerca del estudio de la contaminación auditiva.

2.10 Supuestos y riesgos:

Riesgos	Probabilidad	Alternativas de solución
Escasez de los elementos electrónicos para armar el diseño del sensor	Media	Tener varias formas de armar el sensor con sus repuestos.
que los equipos disponibles no permitan la generación de un dispositivo de clase III (al menos)	Baja	Gestionar el uso de nuevos dispositivos
que el consumo de energía de los dispositivos haga impráctico su uso en el escenario objetivo	Baja	Usar baterías más grandes
problemas con la radio, ya sea por no ser posible configurar las topologías definidas en el estándar (coordinador) o por otro tipo de configuración	Baja	Realizar la prueba con Wi-Fi

2.11 Esquema tentativo:

Introducción

Capítulo 1: Fundamentación teórica

Capítulo 2: Desarrollo del sistema que permita el análisis del ruido

Capítulo 3: Configuración de la red WSN

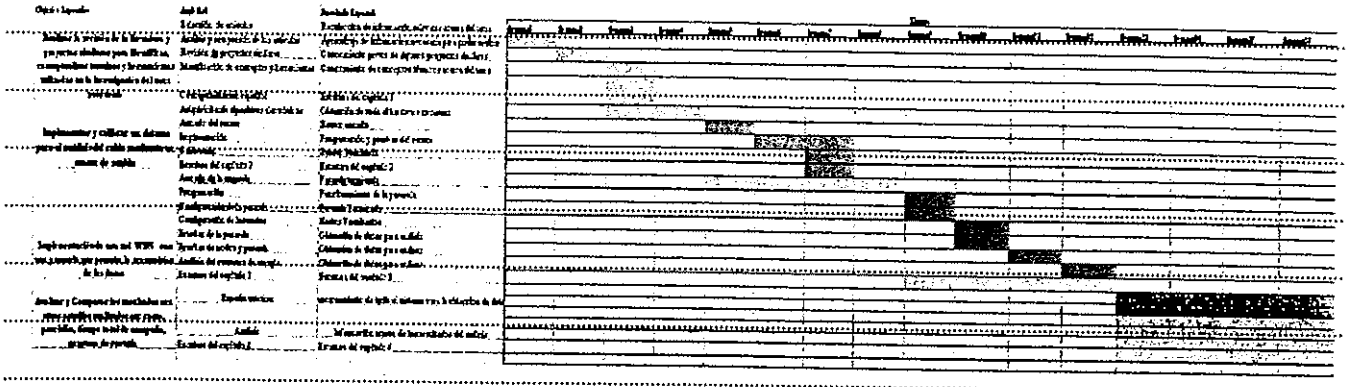
Capítulo 4: Análisis de resultados

Conclusiones y trabajos futuros.

UNIVERSIDAD DEL AZUAY

Bibliografía.

2.12 Cronograma:



2.13Referencias:

Alfie Cohen, M., & Salinas Castillo, O. (2017). Ruido en la ciudad. Contaminación auditiva y ciudad caminable. *Estudios Demográficos Y Urbanos*, *32*(1), 65–96.

Alonso, A. D. E. (2003). Contaminación acústica y salud Noise pollution and health. *Observatorio Medioambiental*, *6*, 73–95.

Chang, S.-F., Chen, C.-F., Wen, J.-H., Liu, J.-H., Weng, J.-H., & Dong, J.-L. (2015). Application and Development of Zigbee Technology for Smart Grid Environment. *Journal of Power and Energy Engineering*, *3*(4), 356.

G. Barros, F. T. (2015). *Sound Noise Levels Calibration with Smart-Phones*, 6.

Alcaldía De Cuenca Red De Monitoreo Emov – EP, (2014). Informe de la calidad del aire, 2014. Cuenca– Ecuador

Rana, R., Chou, C. T., Bulusu, N., Kanhere, S., & Hu, W. (2015). Ear-Phone: A context-aware noise mapping using smart phones. *Pervasive and Mobile Computing*, *17*, 1–22.

Sanchez-Rosario, F., Sanchez-Rodriguez, D., Alonso-Hernández, J. B., Travieso-González, C. M., Alonso-González, I., Ley-Bosch, C., Quintana-Suárez, M. A. (2015). A low

consumption real time environmental monitoring system for smart cities based on ZigBee wireless sensor network. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2015 International* (pp. 702–707).

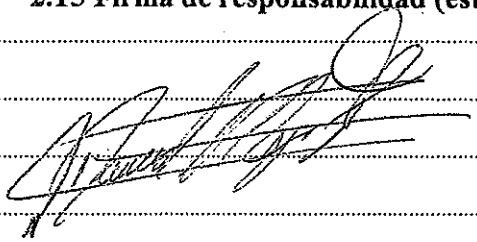
Sanz, B. G., & García, F. J. G. (2003). *La contaminación acústica en nuestras ciudades*. Fundación" La Caixa."

Sevillano, X., Socoró, J. C., Al'\ias, F., Bellucci, P., Peruzzi, L., Radaelli, S., … others. (2016). DYNAMAP--Development of low cost sensors networks for real time noise mapping. Noise Mapping, 3(1).

Wessels, P. W., & Basten, T. G. H. (2016). Design aspects of acoustic sensor networks for environmental noise monitoring. *Applied Acoustics*, *110*, 227–234.

2.14 Anexos: para casos en los que se requiera respaldar el proyecto.

2.15 Firma de responsabilidad (estudiante)



David Mejia

2.16 Firma de responsabilidad (director sugerido)

Ing. Gabriel Barros.

UNIVERSIDAD DEL AZUAY

2.16 Firma de responsabilidad (co-director)

Ing. Diego Chacón.

2.17 Firma de responsabilidad (Asesor Metodológico)

Ing. Diego Chacón.

2.18 Fecha de entrega: 10/05/2018