ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"IMPLEMENTACIÓN DE UN DISPOSITIVO QUE PERMITA ENRUTAR LOS DATOS ADQUIRIDOS DE UNA RED ZIGBEE HACIA LA NUBE UTILIZANDO LA MINICOMPUTADORA RASPBERRY PI Y SOFTWARE DE CÓDIGO ABIERTO"

INFORME DE PROYECTO DE GRADUACIÓN

PREVIO A LA OBTENCIÓN DEL TÍTULO DE:

LICENCIATURA EN REDES Y SISTEMAS OPERATIVOS

PRESENTADO POR:

EDGAR GUALBERTO ARELLANO PEDRAZZOLI

STALIN FROILAN TOMALA MIRANDA

GUAYAQUIL – ECUADOR

2015

AGRADECIMIENTO

A Dios, y en especial a mi madre Andrea Pedrazzoli por todo el apoyo y aliento que me ha dato para completar esta meta y darme la oportunidad de poder estudiar, a nuestro director de tesis el Ing. Ronald Raúl Criollo Bonilla y a las personas que colaboraron de una u otra forma para la realización de este trabajo

Edgar Arellano Pedrazzoli

AGRADECIMIENTO

Antes que todo y en primera instancia agradezco a mi Dios por darme esta gran oportunidad de vida, segundo a mis padres Luz Miranda Montero y Emilio Tomala, por brindarme todo ese apoyo y confianza, al sr. Miguel Maposita Cueva y Sra. por esa ayuda incondicional, a mi querida esposa y compañera de vida Margarita Maposita, por siempre estar a mi lado en todo momento, para finalmente agradecerle principalmente al Ing. Ronald Criollo por creer en nosotros, por fomentarnos la creencia de no rendirnos, siempre luchar hasta el final y por su constante ayuda.

Stalin Tomala M

DEDICATORIA

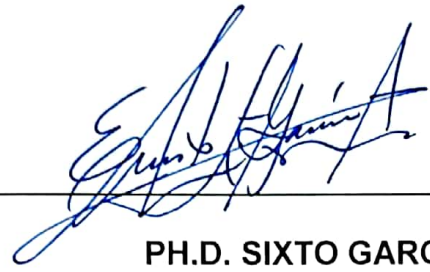
Este trabajo realizado con esfuerzo está dedicado a mi familia, a mi madre y hermanos por apoyarme en todo y a Priscila Matamoros por alentarme a seguir adelante y en especial a mi padre Edgar Arellano Moncayo por ser un ejemplo a seguir.

Edgar Arellano Pedrazzoli

Va dedicado para todas esas personas que siempre fueron optimistas y creyeron en mí, María del Carmen, Enrique, amigos, familiares, pero sobre todo va dedicado especialmente a Margarita Maposita.

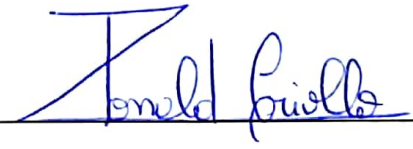
Stalin Tomala Miranda

TRIBUNAL DE SUSTENTACIÓN



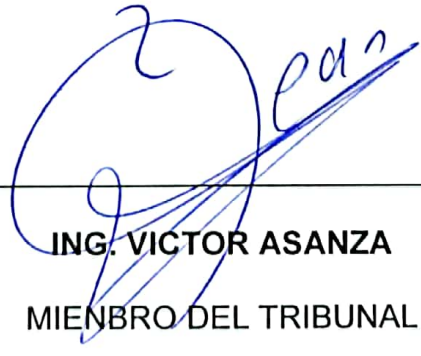
PH.D. SIXTO GARCÍA A.

SUBDECANO SUBROGANTE DE LA FIEC



ING. RONALD CRIOLLO

DIRECTOR DEL PROYECTO DE GRADUACIÓN



ING. VICTOR ASANZA

MIENBRO DEL TRIBUNAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de este informe de proyecto de Grado, nos corresponden exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

Edgar Arellano Pedrazolli

Stalin Tomala Miranda

RESUMEN

Hoy en día estamos atravesando por cambios tecnológicos de gran impacto, entre ellos podemos mencionar el Internet de las cosas (IoT) que busca que la mayoría de los dispositivos puedan estar conectados al Internet. Este concepto ha ido creciendo a un ritmo acelerado de la mano con el uso de las redes de sensores inalámbricas, lo que ha generado la necesidad de dispositivos que permitan recolectar datos de una red inalámbrica para posteriormente enrutarlos hacia la nube.

El proyecto comprende la implementación de un dispositivo que nos permita recolectar los datos provenientes de sensores en una red Zigbee, para posteriormente enrutarlos hacia la nube. Las tecnologías a utilizar serán de código abierto tales como el mini computador Raspberry PI y dispositivos Xbee que están basados en el estándar de comunicación IEEE 802.15.4.

ÍNDICE GENERAL

Pág.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ABREVIATURAS

Ghz	GigaHertz
GPU	Unidad de procesamiento gráfico
HTML	HyperText Markup Language
IEEE	Instituto de Ingeniería Eléctrica y Electrónica
Kbps	KiloBytes Per Second
MHz	MegaHertz
PHP	Personal Home Page
RAM	Memoria de acceso aleatoria
WEB	World Wide Web
SGBD	Sistema de Gestión de Base de Datos
CMD	command prompt

INTRODUCCIÓN

El crecimiento constante de la necesidad tecnológica conlleva que nuevas tendencias vayan hacia un mundo que no esté limitado por cables, siendo su rumbo dirigido hacia un medio inalámbrico en el cual podamos disfrutar de todos los beneficios que se encuentra implícitos en el mismo concepto.

En el presente proyecto se propone la implementación de un dispositivo que permita enrutar los datos adquiridos de una red Zigbee hacia la nube utilizando la minicomputadora Raspberry Pi y software de código abierto.

El primer capítulo da una especificación general de los problemas que presentan actualmente las redes de sensores inalámbricas, y de qué manera nuestro proyecto los solucionará.

El segundo capítulo detalla cada una de las tecnologías necesarias para la elaboración de nuestra solución tecnológica.

En el tercer capítulo se explica el proceso de implementación de nuestra solución tecnológica.

Por último, en el cuarto capítulo se detalla las distintas pruebas realizadas a nuestra solución tecnológica.

CAPÍTULO 1

1. Antecedentes y Justificación

Actualmente el mundo se encuentra en un constante avance tecnológico, donde la implementación de redes sensoriales [1] tiene una función primordial para satisfacer las necesidades de nuestra constante evolución, permitiéndonos dar grandes pasos en la investigación, desarrollo, economía y educación. Normalmente estos sensores usan cables o medios inalámbricos para la transmisión de información hacia un dispositivo que permite visualizar los datos recolectados. Al momento de diseñar e implementar una red de sensores, se crean los siguientes problemas que pretendemos solventar con nuestro dispositivo:

- Costos elevados en hardware y software para control de mallas sensoriales centralizadas.
- Dificultad en la interacción entre tecnologías inalámbricas.
- integración y administración de datos de forma centralizada.
- Lecturas erróneas afectadas directamente por el tiempo.
- Dificultad de visualizar variaciones de las lecturas en el tiempo oportuno.
- Dificultad en integrar servicios de computación en la nube.
- Dificultad en el envío, recepción y almacenamiento de la información para su posterior uso.

1.1. Identificación del problema

Las tecnologías inalámbricas con el paso del tiempo han adoptado un campo bastante amplio en lo que respecta a la mejora continua de las comunicaciones en general, generando grandes cambios e innovaciones tecnológicas que han permitido crear nuevos dispositivos electrónicos cada vez más pequeños y potentes a un costo relativamente bajo, siendo capaces de detectar y medir cualquier magnitud de una forma sencilla y con gran precisión. Estos avances tecnológicos han permitido el crecimiento en la investigación y desarrollo de las

redes de sensores inalámbricas, que permiten comunicar una serie de dispositivos para que trabajen entre sí de forma eficiente. En este tipo de redes, los sensores tienen un papel fundamental como herramientas de recolección de datos, la cual al final es procesada en información útil que es utilizada en diferentes proyectos o aplicaciones.

En una red de sensores podemos encontrar algunos problemas tales como:

- Costo elevado en hardware y software para el control centralizado de los datos.
- Lecturas erróneas afectadas directamente por el tiempo.
- Dificultad de visualizar variaciones de las lecturas en el tiempo oportuno.
- Falta de integración con servicios en la nube.

1.2. Justificación

Las redes de sensores inalámbricas son unas de las tecnologías más investigadas en la actualidad [2], las cuales tienen la capacidad de obtener datos considerables de nuestro entorno en tiempo real. Todo esto hace imprescindible la oportunidad de poder explotar esta información adquirida, en el desarrollo de sistemas complejos tales como: prevención de desastres, control del clima, entre otros. Debido a esto, es muy importante administrar de forma adecuada estos datos, con lo que proponemos como mejor opción nuestro proyecto que consiste

en la implementación de un dispositivo que permitirá enrutar los datos hacia la nube. Nuestra solución contará además con una aplicación web que permitirá gestionar los datos de la red Zigbee.

Actualmente en el mercado no hay un dispositivo similar que utilice hardware y software abierto junto con los servicios en la nube, como la que proporcionará este proyecto, provocando que esta idea sea un reto de emprendimiento e integración tecnológica.

1.3. Objetivos

La implementación de nuestra solución tecnológica pretende alcanzar los siguientes objetivos:

Objetivo General

Establecer la factibilidad de la implementación de un dispositivo que permita enrutar los datos adquiridos de una red Zigbee hacia la nube utilizando la minicomputadora Raspberry Pi y software de código abierto.

Objetivos Específicos

- Analizar los requerimientos tecnológicos de la solución.

- Diseñar la solución utilizando la minicomputadora Raspberry Pi, software de código abierto y servicios en la nube.
- Implementar la solución.
- Realizar diferentes pruebas a la solución.

1.4. Metodología

El proceso para el desarrollo de nuestro proyecto consiste en las siguientes etapas:

1. Análisis de la solución.
2. Diseño del modelo de la Base de Datos del Sistema de Administración Web.
3. Diseño y desarrollo del Sistema de Administración Web.
4. Configuración de la minicomputadora Raspberry pi.
5. Instalación y configuración del módulo de transmisión inalámbrica Xbee.
6. Instalación y configuración del módulo de acceso a Internet.
7. Instalación del Sistema de Administración Web.
8. Configuración de servicios en la nube.
9. Ejecución de pruebas a la solución.
10. Análisis de resultados.

1.5. Limitaciones

Las limitaciones para la implementación de nuestro proyecto están relacionadas con la minicomputadora Raspberry Pi modelo B, las cuales son detallas a continuación:

- Tiene 512 Mb de memoria no expandible y un procesador ARM 1176JZF-S de solo 700 MHz [3] integrados a la tarjeta principal.
- Tiene únicamente 2 puertos USB.
- No posee puerto VGA.

CAPÍTULO 2

2. Marco teórico

2.1. Red de sensores inalámbricos

La red de sensores inalámbricos es una agrupación de dispositivos de censado intercomunicados por el medio inalámbrico, que poseen una baja capacidad de consumo de energía y permiten la recolección de datos, encontrándose limitados en su capacidad computacional y de comunicación. La principal función de estos dispositivos es el de trabajar de forma colaborativa con los diferentes nodos que existen dentro de la red [4].

Este tipo de redes puede llegar a tener miles de sensores conectados entre sí, por lo que hace que la mayoría de sus aplicaciones directas sean para monitoreo o detección de perturbaciones o cambios de algún parámetro [4].

En la actualidad los sensores inalámbricos son usados para distintos tipos de aplicaciones tales como: medición de temperatura, humedad, salinidad, pH, etc. También podemos ver que son utilizados en el área de domótica, administración de alimentos, control de cargas, vehículos etc. Las facilidades que brindan los sensores inalámbricos a los usuarios son innumerables y evita tener que estar restringidos por los molestosos cables.

Figura 2.1 MeshLium Xtreme – basado en redes ZigBee

[5]

Los beneficios principales que brinda la red de sensores inalámbricos son:

- Seguridad: Estos dispositivos inalámbricos nos permiten instalarlos en zonas de difícil acceso por lo cual la recolección de datos va ser realizada de manera remota.
- Convergencia: Todos los datos serán recolectados de forma centralizada a un solo servidor, para su posterior análisis.
- Escalabilidad: Los sensores inalámbricos nos brindan la capacidad de poder aumentar la cantidad de dispositivos recolectores de datos.
- Reducción de costos: Los costos de equipos e implementación no requieren inversiones de grandes sumas de dinero ya que todos estos sensores y dispositivos se encuentran en el mercado a un costo realmente accesible. [6]

2.2. Estándar Zigbee

2.2.1. Protocolo Zigbee

Zigbee está basado en el estándar IEEE 802.15.4, que define una asociación de protocolos para la transmisión de datos en redes inalámbricas de bajo consumo. Las bandas en las que opera son las de 868 MHz, 915 MHz y 2.4 GHz. Al operar en esas frecuencias, hacen que la señal funcione correctamente con ambientes que posean obstáculos [7],

llegando a trasmitir datos hasta 250 Kbps. En la frecuencia de 2.4 GHz se establecen hasta 16 canales.

Este estándar está diseñado con las siguientes especificaciones:

- Consumo mínimo de energía.
- Dispositivos y mantenimiento de los mismos a un costo bajo.
- El alcance puede llegar en ambientes ideales hasta 50 mts.
- La tasa de transmisión puede llegar hasta 250 Kbps.

2.2.2. Nodos Zigbee

El modo en la que opera cada nodo [6] pueden ser como: Coordinador, Ruteador o Dispositivo Terminal.

Coordinador: Este es un punto clave para la red ZigBee, sólo un nodo puede ser coordinador dentro de la red, Actúa como nodo raíz en la topología y está encargado de estas funciones:

- Formar la red con los otros Xbee.
- Administra la red ZigBee.
- Coordinar las direcciones de red.
- Asignación de direcciones de red.
- Configuración de los parámetros de red.

Ruteador: El nodo ruteador sirve como intermediario entre nodos que están fuera de su alcance extendiendo su cobertura de la red y para aumentar la creación de rutas adicionales de datos formadas por el coordinador.

Dispositivo Final: Este nodo se comunica con un nodo ruteador o el coordinador, tienen menos potencia y usualmente son alimentados a batería.

Una de las características importantes en una red de sensores es que tienen capacidad de encontrar nuevas rutas para enviar los paquetes de datos para que estos no se pierdan en la red, esta es una propiedad que tiene este tipo de redes para solventar problemas que no eran posibles con otras tecnologías. Los módulos Xbee soportan el protocolo ZigBee.
En toda red con módulos Xbee deben existir dos o más dispositivos donde un módulo se le asigna una configuración especial denominada Coordinador. El coordinador es el responsable de formar la red y los otros son los dispositivos finales.

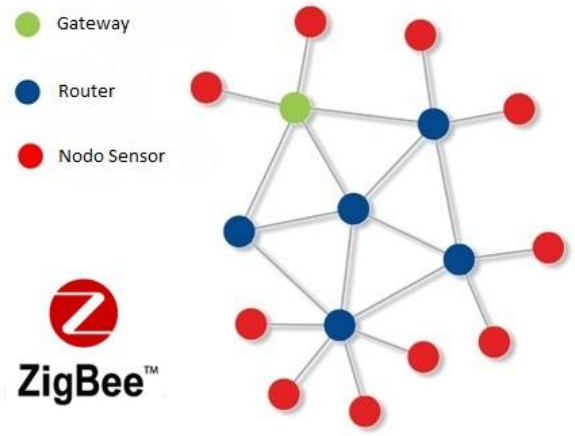


Figura 2.2 Nodos ZigBee

[8]

2.2.3. Topología

La red ZigBee se puede conectar de diferentes formas y diseños para crear su estructura [6]. A continuación se detalla cada uno de los tipos:

Red en Par: Es la topología de red más básica formada por la unión de un coordinador y un enrutador.

Red de tipo estrella: Este diseño simple, emplea la comunicación directa con un dispositivo central, que en este caso será el nodo en modo coordinador.

Red en malla: Es una topología de comunicación conformada por nodos tipo enrutador y coordinador, donde existen además varios dispositivos

finales conectados a los nodos tipo enrutador. A continuación mencionamos algunas características importantes de esta topología:

- Varios dispositivos finales se pueden comunicar con cualquier enrutador o coordinador de la red.
- Se puede generar y recibir información entre nodos distantes.
- Identificación automática de nodos y rutas faltantes para que la red siga funcionando.

En la siguiente Figura se puede observar los distintos tipos de topologías.

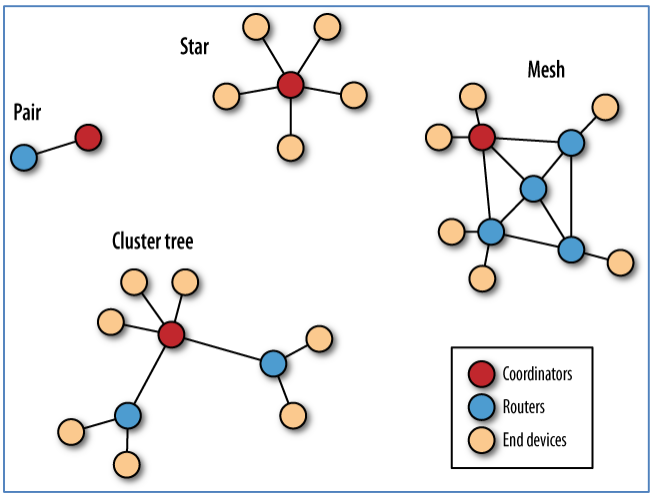


Figura 2.3 Diseños de Topologías [6]

2.2.4. Modo XBEE-API

El modo API [6] que en español significa Interfaz de Programación de Aplicaciones, representa un conjunto de interfaces estándar que han sido creados para que un software pueda interactuar con otro.

Este modo permite que el computador realice peticiones de solicitud de una aplicación a otra de manera estándar, logrando que la interacción sea mucho más eficiente con los demás equipos. El tipo de comunicación que emplea este modo es de dispositivo a dispositivo.

2.3. Minicomputadora Raspberry Pi

Durante mucho tiempo las computadoras han venido evolucionando de una manera muy rápida y uno de los puntos más importantes dentro de los avances tecnológicos ha sido el campo de los micro controladores, que son pequeños dispositivos electrónicos que permiten realizar un sinnúmero de funciones. Estos avances llevaron a la aparición del Raspberry Pi en el año 2006.

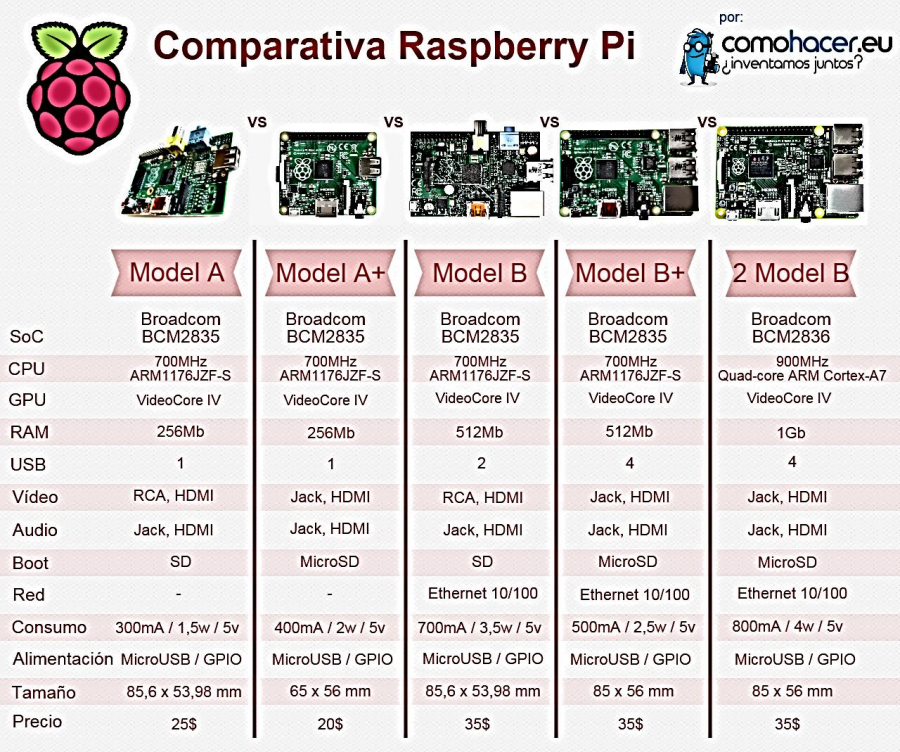
El Raspberry Pi [9] es una minicomputadora que posee características similares con una de escritorio pero a bajo costo. Esta minicomputadora no incluye un disco duro sólido pero es necesario una tarjeta SD con una capacidad mínima de 2 Gb para cargar el sistema operativo basado en Linux en su versión Debían con la distribución Raspbian Wheezy [9].

Figura 2.4 Minicomputadora Raspberry Pi

2.3.1. Características de la Minicomputadora Raspberry Pi

En la actualidad han surgido varios modelos de la minicomputadora Raspberry Pi, con diferentes capacidades en procesamiento, memoria RAM, conectores y costo.

En la siguiente Figura podemos observar las distintas características de los diferentes modelos de la minicomputadora Raspberry Pi.



	Model A	Model A+	Model B	Model B+	2 Model B
SoC	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2836
CPU	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	900MHz Quad-core ARM Cortex-A7
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV
RAM	256Mb	256Mb	512Mb	512Mb	1Gb
USB	1	1	2	4	4
Vídeo	RCA, HDMI	Jack, HDMI	RCA, HDMI	Jack, HDMI	Jack, HDMI
Audio	Jack, HDMI	Jack, HDMI	Jack, HDMI	Jack, HDMI	Jack, HDMI
Boot	SD	MicroSD	SD	MicroSD	MicroSD
Red	-	-	Ethernet 10/100	Ethernet 10/100	Ethernet 10/100
Consumo	300mA / 1,5w / 5v	400mA / 2w / 5v	700mA / 3,5w / 5v	500mA / 2,5w / 5v	800mA / 4w / 5v
Alimentación	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO
Tamaño	85,6 x 53,98 mm	65 x 56 mm	85,6 x 53,98 mm	85 x 56 mm	85 x 56 mm
Precio	25$	20$	35$	35$	35$

Figura 2.5 Comparativas de la minicomputadora Raspberry PI [10]

La minicomputadora Raspberry Pi a partir del modelo B cuenta con un puerto Ethernet integrado a la tarjeta, permitiendo establecer la conexión con otros equipos que pertenezcan a la misma red LAN. Otra característica adicional es que nos permite conectar a una red inalámbrica a través de un adaptador USB.

2.3.2. Sistema Operativo

En cuanto al sistema operativo, ocupa poco espacio en la memoria SD (2gb) debido a que viene instalado con un mínimo de aplicaciones. A

medida que se requieran más aplicaciones, se las puede fácilmente bajar e instalar usando líneas de comando[9].

La distribución Raspbian Wheezy viene con algunas aplicaciones preinstaladas, entre las que se encuentran los navegadores de internet Midori, Dillo y NetSurf y herramientas de programación para Python, Scratch y Squeak.

El entorno de escritorio por defecto es el LXDE. Un ejemplo de esto lo podemos ver en la siguiente Figura que nos muestra lo simple y minimalista que es este entorno.

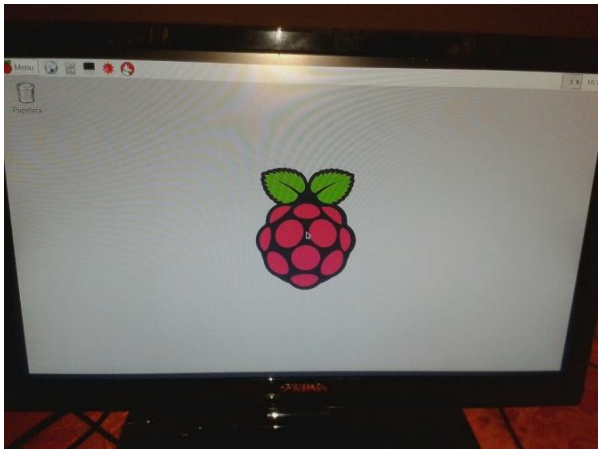


Figura 2.6 Entorno de escritorio LXDE de la distribución Raspbian Wheezy

2.3.3. Aplicaciones de la minicomputadora Raspberry Pi

La minicomputadora Raspberry Pi modelo B puede tener infinidades de aplicaciones, todo dependerá de la imaginación y conocimiento del usuario que lo use. Entre las distintas aplicaciones destacamos las siguientes:

- Servidor Web.
- Servidor de Base de Datos.
- Servidor de Archivos.
- Servidor Multimedia.

La minicomputadora Raspberry Pi es útil para automatizar determinadas actividades en el hogar, desdé el control remoto del aire acondicionado, iluminación, apertura o cierre de puertas o control de sensores y actuadores, etc.

Sus conexiones y los diferentes dispositivos que podemos conectar en la minicomputadora Raspberry Pi modelo B se explican con la siguiente imagen.

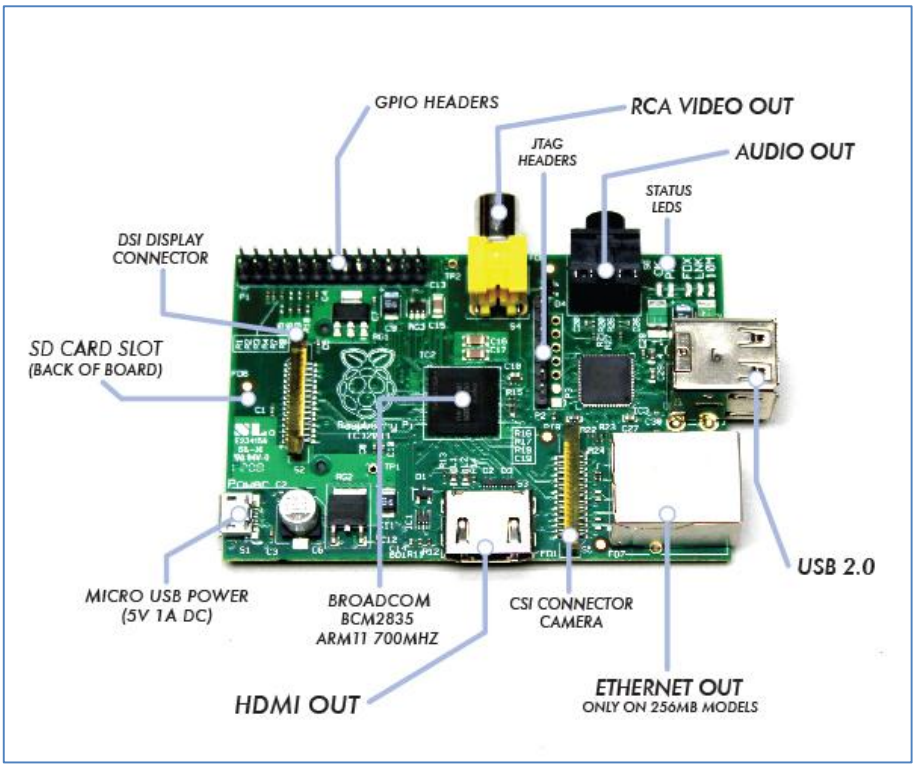


Figura 2.7 Puertos y Entradas de la minicomputadora Raspberry PI modelo B [12]

2.3.4. Ventajas y desventajas de la minicomputadora Raspberry Pi

Su fuente de alimentación es una ventaja ya que su procesador requiere un bajo consumo de energía, por lo tanto basta con un cargador con conexión micro USB con un voltaje de 5V.

Su núcleo es un chip integrado Broadcom BCM2835, que contiene un procesador ARM11 con una tarjeta gráfica VideoCore IV con 512 MB con

la posibilidad de reproducir videos en 1080p por medio de su puerto HDMI que está integrado al Raspberry.

El minicomputador Raspberry Pi incluye puertos en todos los lados de la placa, y esto hace que sea un poco complicado al usar el dispositivo en espacios muy reducido.

2.4. Estándares Web

2.4.1. PHP

PHP [13] representa a un lenguaje de alto nivel de código abierto que permite la elaboración de páginas web dinámicas. La característica principal es que es un lenguaje interpretado, por lo que no es necesario pasar un proceso de compilación.

PHP es un lenguaje del lado del servidor, que está compuesto de un intérprete que se encarga de analizar el script que ha sido solicitado por un navegador. Una vez realizado este proceso, el resultado es enviado al navegador. La sintaxis que PHP prácticamente es muy parecida al lenguaje en C por lo que le da la característica de ser un lenguaje práctico, tradicional y fácil de implementar.

El lenguaje PHP tiene soporte para ser implementado en una gran cantidad de sistemas operativos tales como: Windows, Linux, Solaris, etc.

2.4.2. HTML

HTML o lenguaje de marcado de hipertexto [13], se encuentra presente en la mayoría de las páginas web que se encuentran en internet y está permanentemente siendo desarrollado para mejorar la atención demandada por los usuarios. Está formado por etiquetas, las cuales son a la vez instrucciones que son contenidas por corchetes angulares como por ejemplo: <HTML> que define el inicio de una página web.

HTML no está definido como un lenguaje de programación sino como un lenguaje de marcado o etiquetas, ya que puede integrar mediante los famosos tags o etiquetas un sinnúmero de cosas tales como [13]: tablas, pequeñas o grandes listas, párrafos, textos, imágenes, viñetas, números, etc. Debido a este lenguaje podemos observar a través de nuestras páginas web una gran gama de información.

2.4.3. CCS

CCS [14] en español conocido como hojas de estilo en cascada, es un lenguaje utilizado junto con HTML, cuya principal función es encargarse de toda la apariencia de los documentos programados.

Este tipo de apariencia que se le da al documento tiene varios rangos de complejidad, pudiendo ir desde la más simple presentación, hasta la más llamativa o compleja creación.

El CCS puede dar o brindar gestión a todo lo que representa la imagen o apariencia del documento, dejando que únicamente el HTML se encargue de todas las funciones estructurales y de codificación.

2.5. Servicios en la nube

Computación en la nube [15] es prácticamente un modelo que refiere a un conjunto o agrupación de recursos que son distribuidos y estructurados, estos pueden ser asignados ágilmente y también se pueden liberar de igual forma, con una mínima gestión de parte del proveedor que brinda el servicio. Estos servicios son compartidos públicamente y también pueden ser configurables como por ejemplo: Almacenamiento, aplicaciones, base de datos, entre otros. Este tipo de servicios traen consigo siempre grandes cantidades de beneficios considerables, principalmente para la economía mundial.

CAPÍTULO 3

3. Análisis y Diseño de la Solución

3.1. Descripción de la solución

Nuestro proyecto está basado en la implementación de un dispositivo que permitirá enrutar los datos adquiridos de una red Zigbee hacia la nube utilizando la minicomputadora Raspberry Pi. Los datos provenientes de los sensores de la red serán recolectados utilizando un script desarrollado con el lenguaje Python, para posteriormente ser almacenados en una tabla de una base de datos previamente configurada en el SGBD Mysql. En la parte final se utilizará otro script desarrollado con Python que se encargará de enrutar los datos hacia el hosting llamado siteground [16].

Los servicios en la nube en nuestro proyecto pretenden hacer referencia a todos los beneficios que se tendrán al usar todos los recursos en internet, permitiendo que el proveedor se encargue de manera eficiente y óptima del procesamiento y almacenamiento de datos.

Siteground como empresa proveedora de servicios en la nube nos ofrece lo siguiente:

- Disponibilidad: Permite tener los recursos disponibles 24 horas del día y los 7 días de la semana. De manera adicional, el proveedor se encargará de toda la gestión de recursos.
- Movilidad: ya que es un servicio online, vamos a tener disponibles todos los datos recolectados de los sensores desde cualquier parte del planeta, con sólo tener una conexión a Internet.

3.2. Diagrama de Casos de Uso

En este diagrama se explicará los roles que tiene el administrador y el operador en cuanto al uso del sistema.

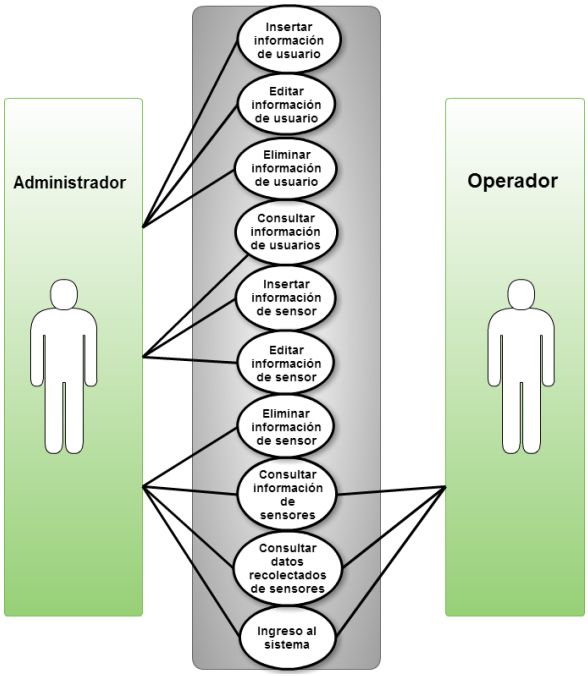


Figura 3.1 Diagrama de Casos de Usos

3.3. Diagrama de Procesos

A continuación se puede observar los diagramas relacionados a los procesos principales de nuestro proyecto.

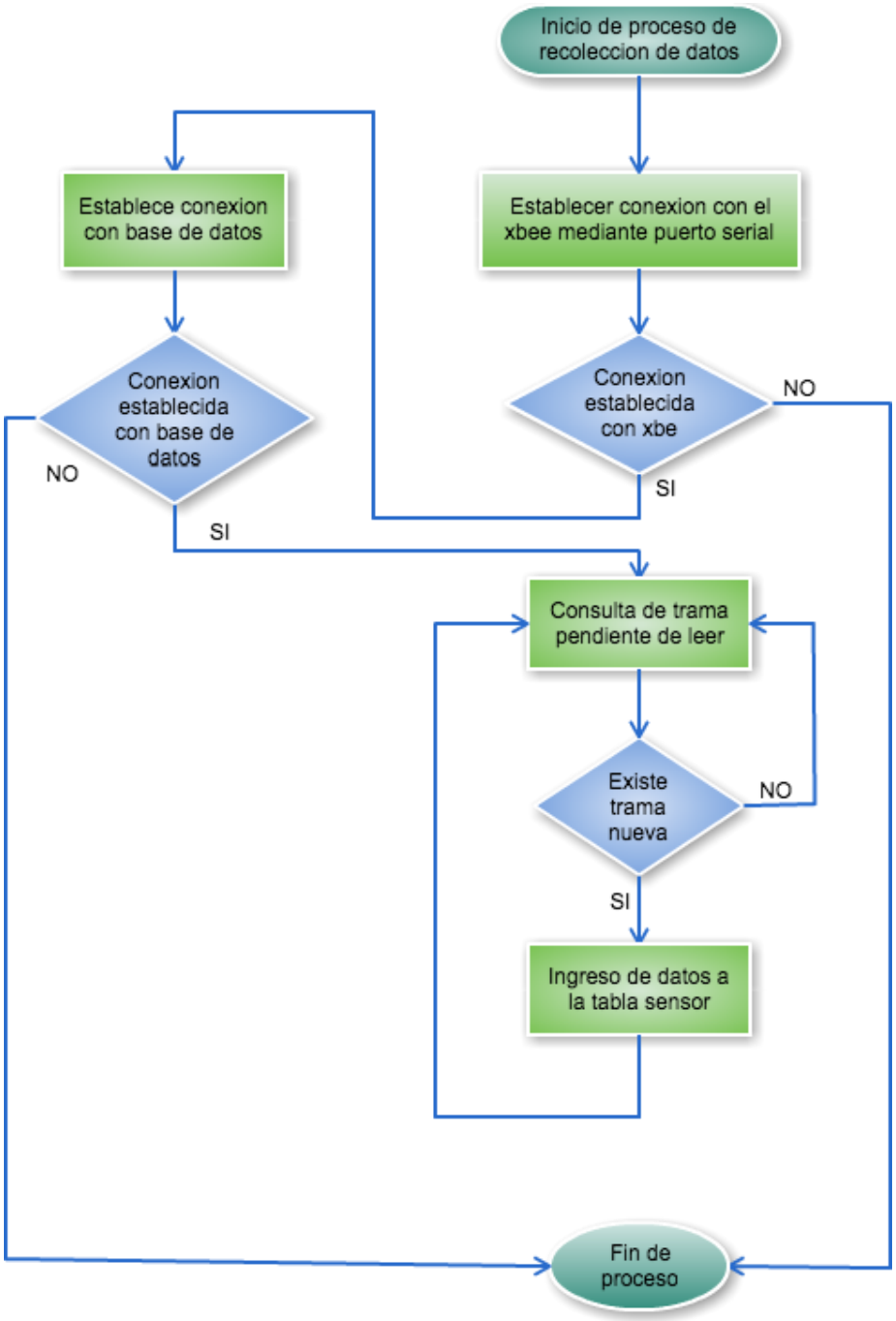


Figura 3.2 Diagrama de proceso de recolección de datos por parte del dispositivo enrutador

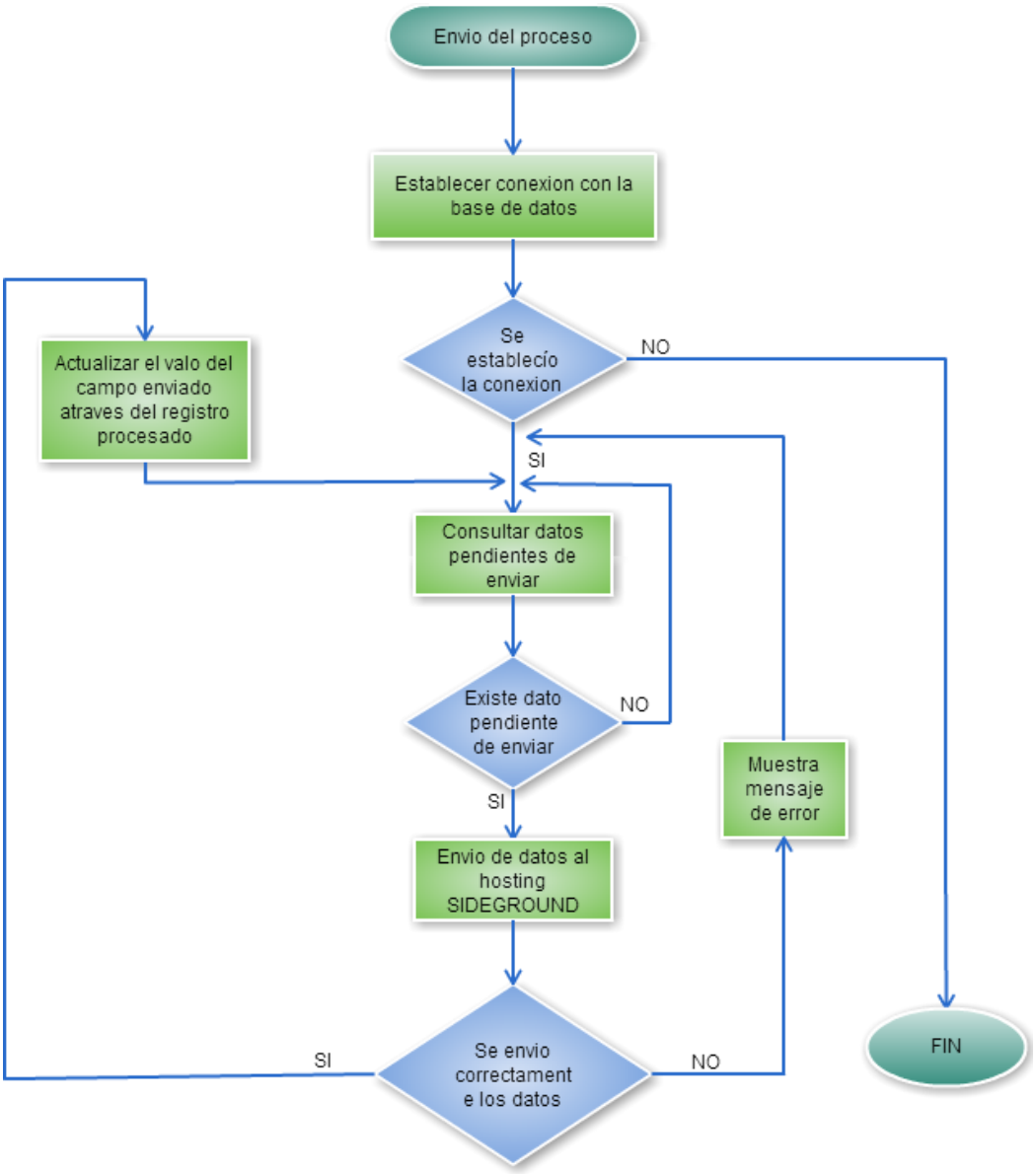


Figura 3.3 Diagrama de proceso de envío de datos hacia el hosting Siteground

3.4. Diagrama de Bloques

El diagrama de bloques explicará cómo está formado nuestro sistema de recolección y enrutamiento de datos.

En la siguiente gráfica mostraremos el Bloque del enrutador:

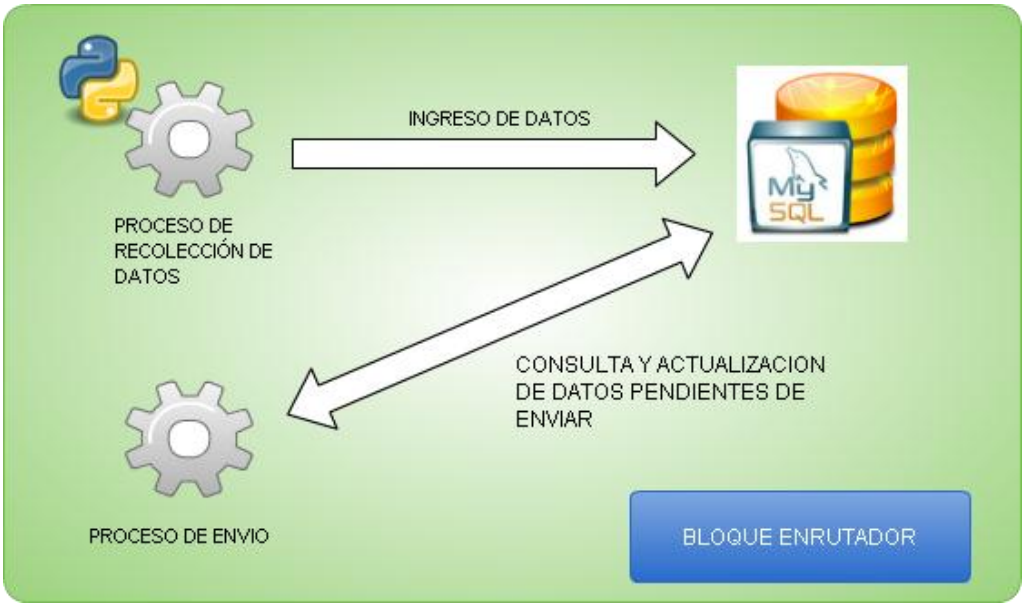


Figura 3.4 Diagrama de Bloque del Enrutador

En la siguiente gráfica mostraremos el Diagrama de Bloque general:

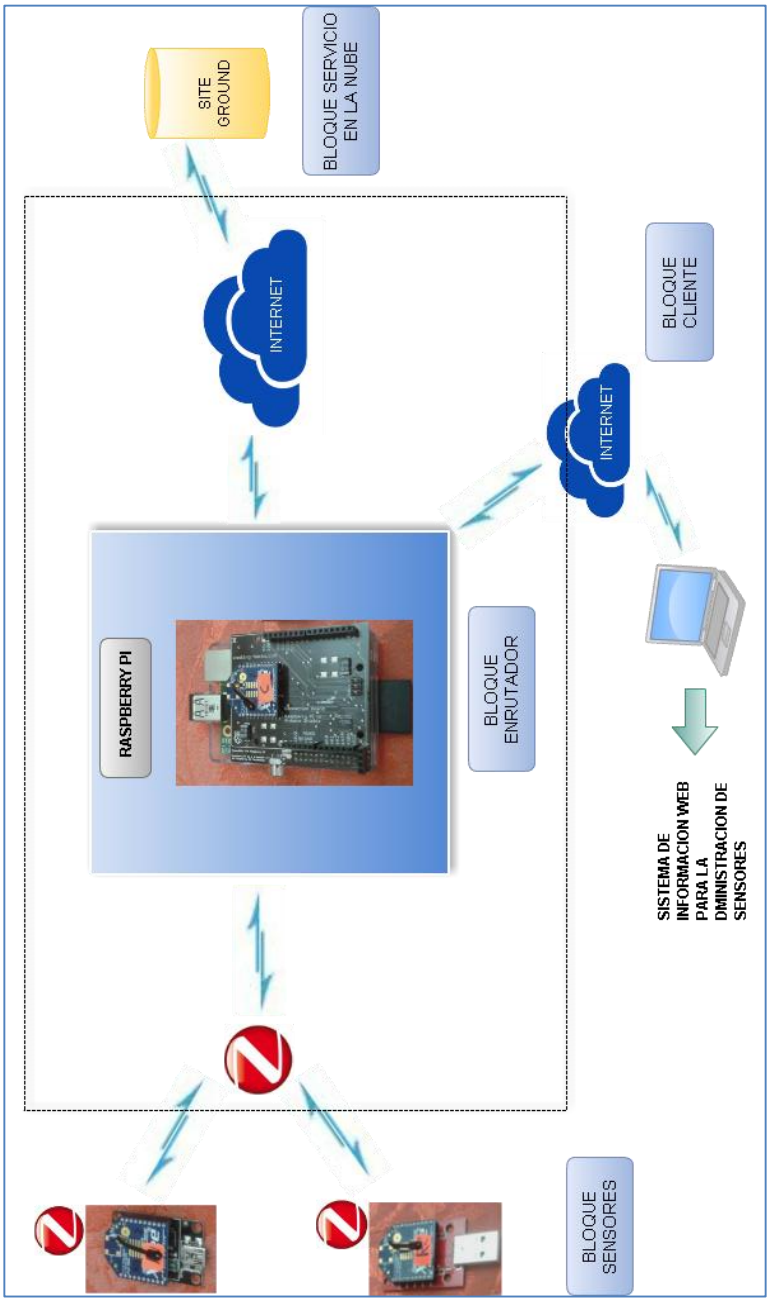


Figura 3.5 Diagrama de Bloques General

3.5. Diseño del Modelo de la Base de Datos

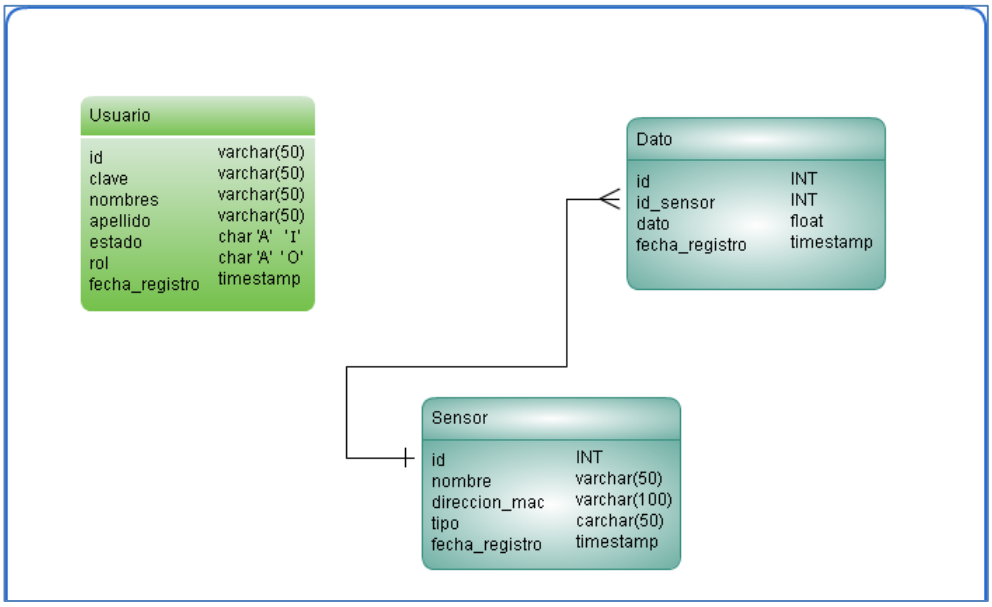


Figura 3.6 Diseño del Modelo de la Base de Datos

3.6. Arquitectura del Hardware

A continuación se detalla el hardware necesario para la implementación de nuestra solución tecnológica:

- Una minicomputadora Raspberry Pi Modelo B.
- Una tarjeta SD de 8 GB marca Kingston que almacenará el sistema operativo.
- Un adaptador USB wifi 802.11 g/b/n que proporcionará la conexión inalámbrica al enrutador.

Figura 3.7 Memoria SD

- Dos módulos Xbee Series 2, que proporcionarán la comunicación inalámbrica bajo el estándar Zigbee.

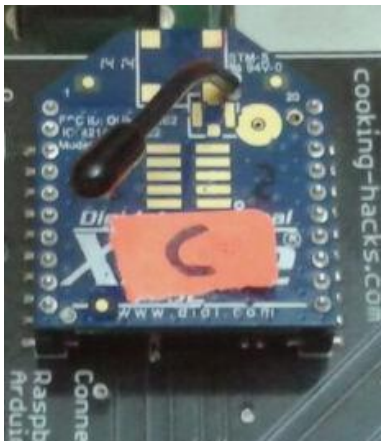


Figura 3.8 Xbee S2

- Un Raspberry PI Shields Connection Bridge cooking-hacks, que es un dispositivo creado por la empresa cooking hacks, que nos brindará la conectividad y comunicación con el Xbee coordinador.

Figura 3.9 Xbee S2

- Un Xbee Xplorer, que permitirá la configuración de parámetros de los dispositivos Xbee.

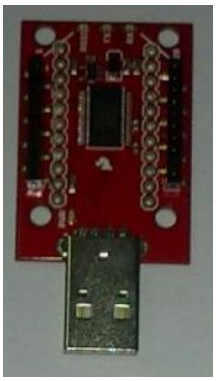


Figura 3.10 Xbee Xplorer para configuración

3.7. Arquitectura del Software

Nuestro dispositivo contará en la parte de software con dos scripts desarrollados con el lenguaje Python y un aplicación web para la gestión de datos de la red ZigBee.

3.7.1. Script de recolección de datos

Para que nuestro dispositivo funcione correctamente tiene que tener un script que inicialice la captura de la trama enviada por los sensores desde la red ZigBee.

A continuación se explica el funcionamiento del script:

- Se importa las librerías necesarias para el funcionamiento del script.
- Se establece la conexión con la base de datos mediante un constructor.
- Se apertura el puerto serial que es el ttyAMA0.
- Se crea un objeto del API XBEE para poder realizar la captura de la trama. Usamos ZigBee ya que nuestros radios son XBEE series 2.
- Se inicia la captura y almacenamiento continúo de información.
- Finalizamos cerrando la comunicación serial.

En la siguiente imagen podemos observar el script de recolección de datos.

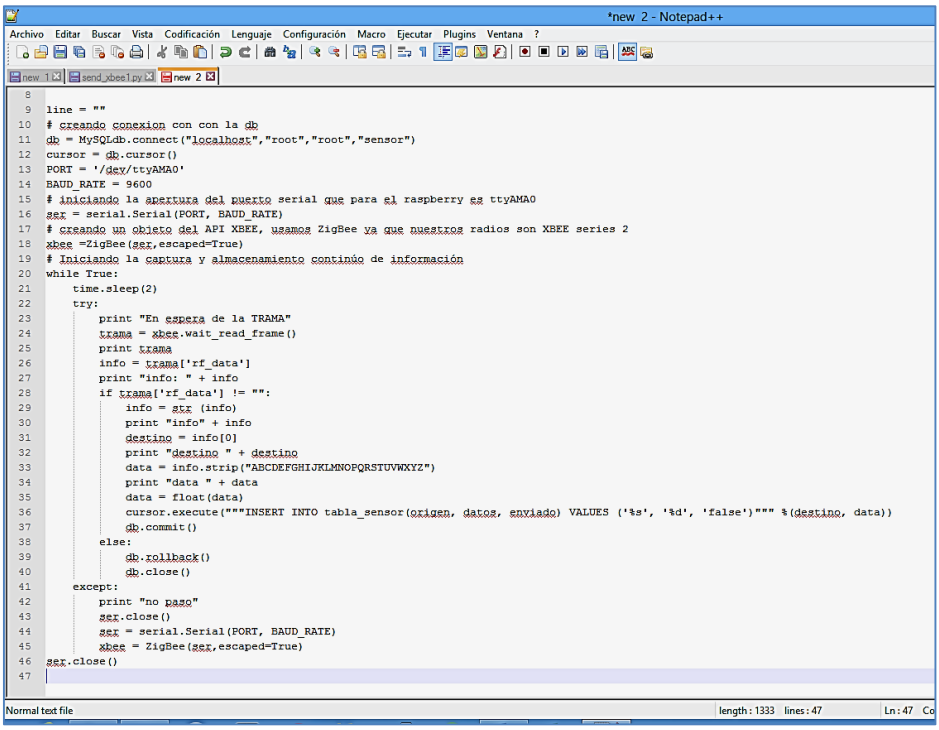


Figura 3.11 Captura de Pantalla del Script de Python para recepción de Tramas (ver ANEXO IV)

3.7.2 Script de envío de datos hacia la nube

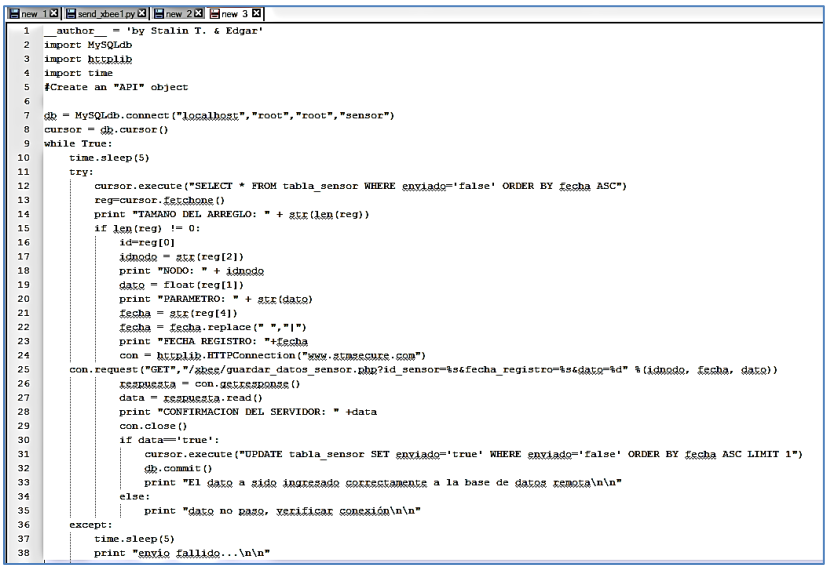
Este script es muy importante ya que él se encargará de enviar la información a la nube para ser almacenada y usada posteriormente como lo requiera el usuario.

A continuación se explica el funcionamiento del script:

- Conexión con la base de dato local.
- Lectura de datos.

- Petición http al hosting Siteground.
- Actualización de tablas.

En la siguiente imagen podemos observar el script de envío de datos hacia la nube.



```
__author__ = 'by Stalin T. & Edgar'
import MySQLdb
import httplib
import time
#Create an "API" object

db = MySQLdb.connect("localhost","root","root","sensor")
cursor = db.cursor()
while True:
    time.sleep(5)
    try:
        cursor.execute("SELECT * FROM tabla_sensor WHERE enviado='false' ORDER BY fecha ASC")
        reg=cursor.fetchone()
        print "TAMANO DEL ARREGLO: " + str(len(reg))
        if len(reg) != 0:
            id=reg[0]
            idnodo = str(reg[2])
            print "NODO: " + idnodo
            dato = float(reg[1])
            print "PARAMETRO: " + str(dato)
            fecha = str(reg[4])
            fecha = fecha.replace(" ","|")
            print "FECHA REGISTRO: "+fecha
            con = httplib.HTTPConnection("www.stsecure.com")
    con.request("GET","/xbee/guardar_datos_sensor.php?id_sensor=%s&fecha_registro=%s&dato=%d" %(idnodo, fecha, dato))
            respuesta = con.getresponse()
            data = respuesta.read()
            print "CONFIRMACION DEL SERVIDOR: " +data
            con.close()
            if data=='true':
                cursor.execute("UPDATE tabla_sensor SET enviado='true' WHERE enviado='false' ORDER BY fecha ASC LIMIT 1")
                db.commit()
                print "El dato a sido ingresado correctamente a la base de datos remota\n\n"
            else:
                print "dato no paso, verificar conexión\n\n"
    except:
        time.sleep(5)
        print "envio fallido...\n\n"
```

Figura 3.12 Script de envío de datos hacia la nube

(Ver ANEXO V)

3.7.2. Aplicación web para la gestión de los datos de la red Zigbee

La herramienta PHP generator for MySql [17] en su versión gratuita nos permitirá generar el código basado en PHP de nuestra aplicación web para la gestión de datos de la red Zigbee.

A continuación se detalla el uso de la herramienta PHP generator for MySql:

Se coloca la IP del servidor MySql, luego se selecciona la base a usar, que en este caso sería BASE_SENSOR con su respectivo usuario y contraseña de administración.

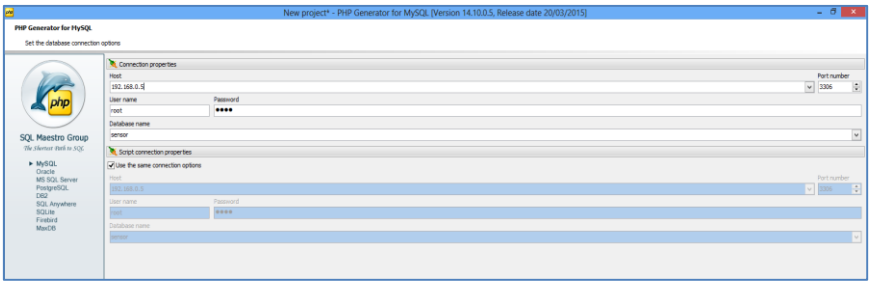


Figura 3.13 Php generator

Una vez realizado el proceso de conexión con la base de datos, se procede con la selección de las tablas.

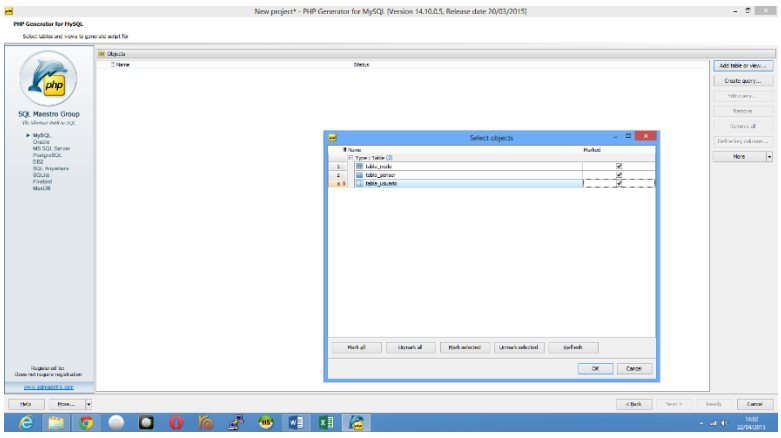


Figura 3.14 Selección de tablas

El siguiente paso es definir los permisos de administración para el usuario final y el administrador.

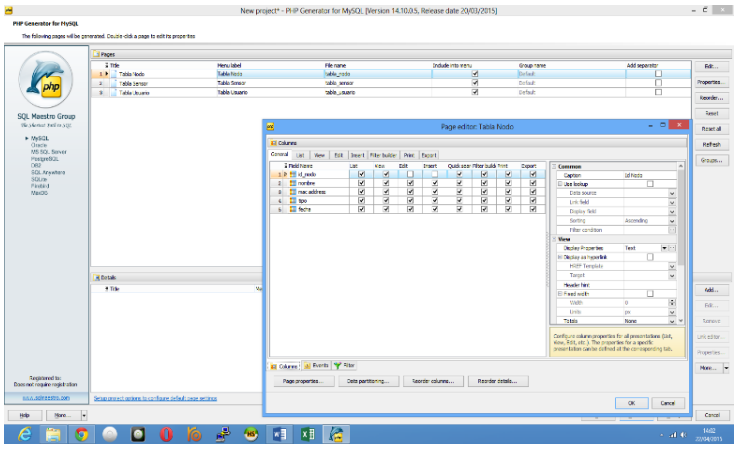


Figura 3.15 Permisos Generales para la tabla

Después seleccionaremos y configuraremos la paraciencia de la página con CCS y JavaScript.

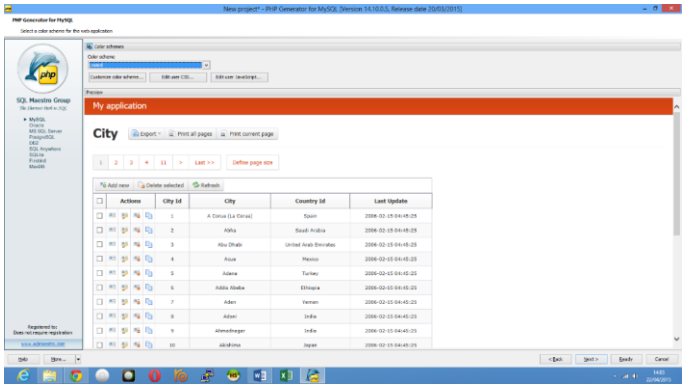


Figura 3.16 Selección de Apariencia

Posteriormente se almacenará en una carpeta llamada sensor los archivos de la aplicación web.

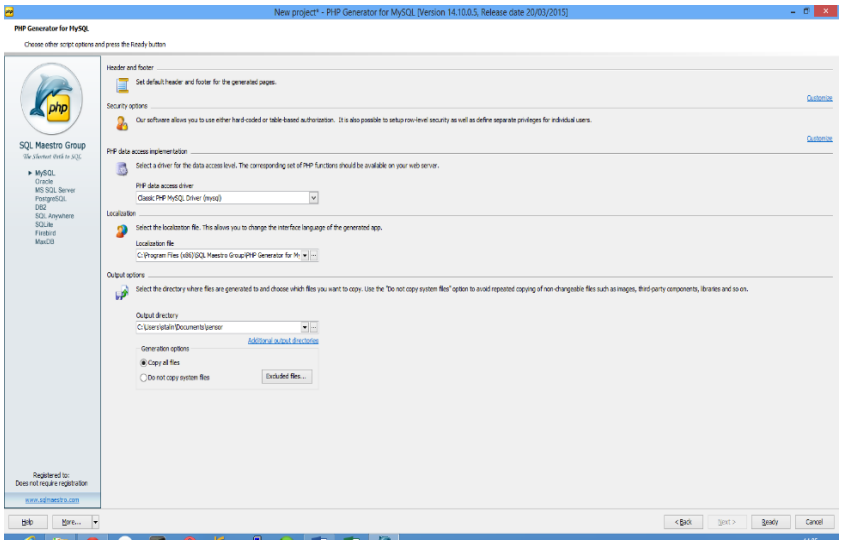


Figura 3.17 Selección de Almacenado en Disco

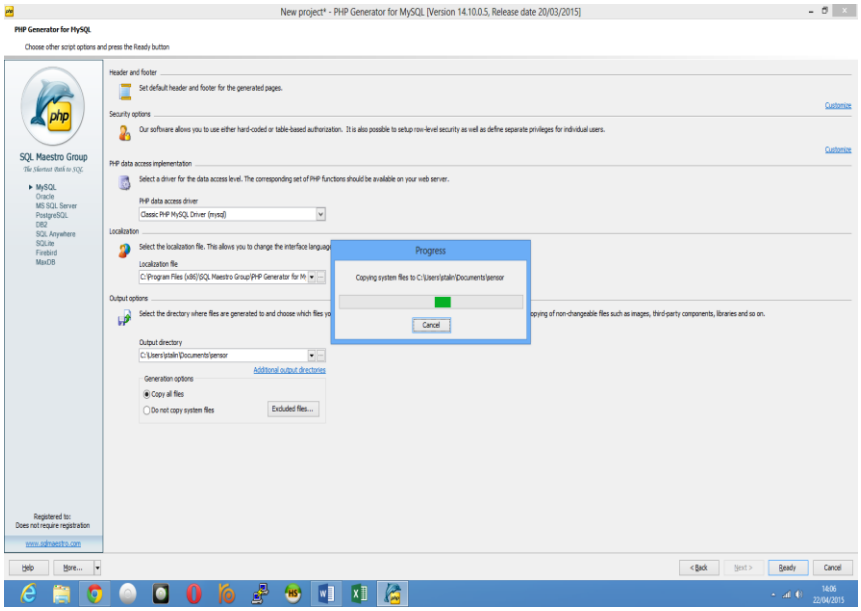


Figura 3.18 Finalización de procesos

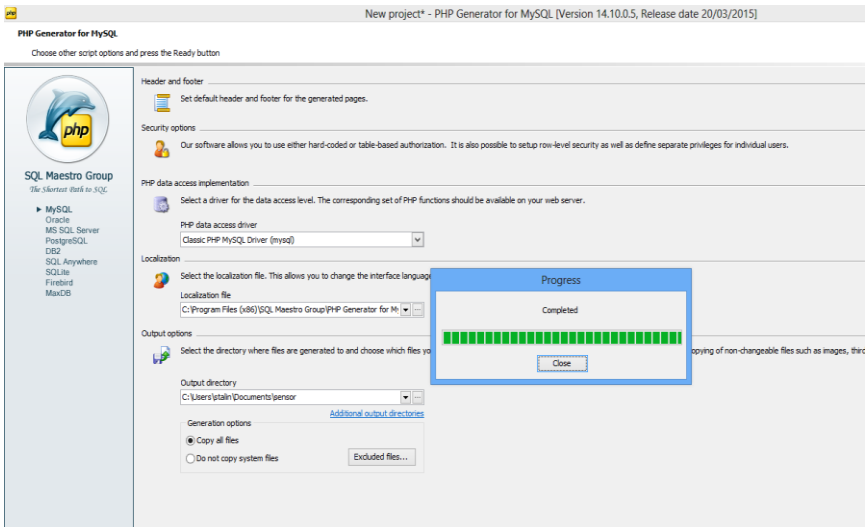


Figura 3.19 proceso completado

3.8. Diseño de Pruebas de la solución

Para poder probar nuestra solución tecnológica, crearemos un ambiente controlado con la simulación de 2 sensores con comunicación Xbee, para lo cual contaremos con 2 computadoras con la distribución Linux Ubuntu [19], las mismas que tendrán precargada las configuraciones necesarias para que nuestro script plantilla funcione (ver ANEXO I).

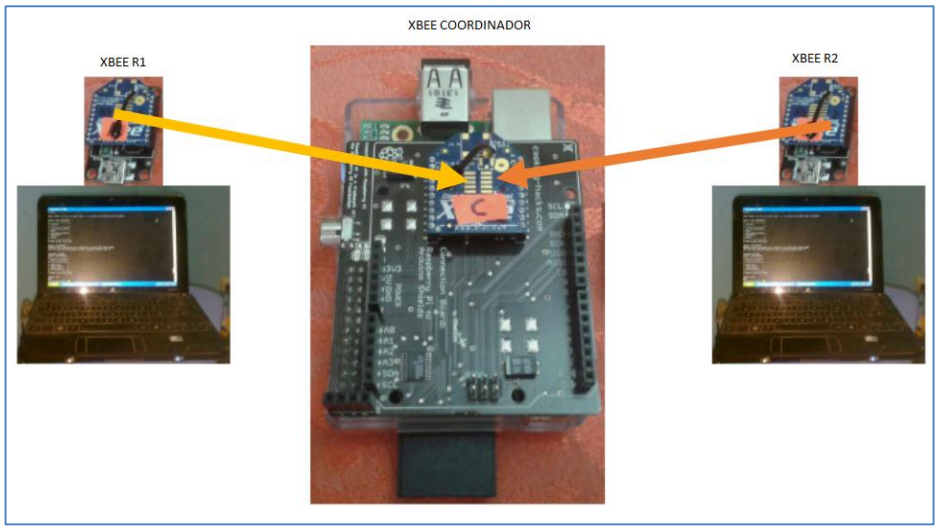


Figura 3.20 Diseño de escenario y simulación de sensores

CAPÍTULO 4

4. Implementación y Pruebas de la Solución

4.1. Implementación del Hardware

Los componentes principales que forman parte del hardware de nuestro proyecto son los siguientes:

- Minicomputadora Raspberry pi
- Módulos Xbee S2. que proporcionarán la comunicación inalámbrica bajo el estándar Zigbee.

- Placa Raspberry PI Shields Connection Bridge de cooking-hacks [20] que permitirá conectar el módulo Xbee en modo coordinador con la minicomputadora Raspberry Pi.

Figura 4.1 Implementación del Hardware

4.2. Implementación del Software

4.2.1. Implementación del Servidor Web Apache

El primer paso inicia con la otorgación de permisos al grupo que usa apache por defecto.

```
# sudo addgroup www-data
# sudo usermod -a -G www-data www-data
```

Después de esto procedemos a la actualización de los repositorios, para que nos permita la instalación de Apache y PHP.

```
# sudo apt-get update
```

```
# sudo apt-get install apache2 php5 libapache2-mod-php5
```

Una vez realizado el paso anterior procedemos con el reinicio del servicio Apache por medio de la siguiente sentencia.

```
# sudo /etc/init.d/apache2 restart
```

El siguiente paso consiste en la instalación del motor de base de datos MySql y la aplicación web phpmyadmin:

Para esto debemos activar la interfaz de loopback, para que podamos visualizar consultar o realizar peticiones hacia el propio host.

```
# loopback
# sudo ifup lo
```

A continuación instalamos el servicio de mysql y la aplicación web phpmyadmin.

```
# sudo apt-get install mysql-server mysql-client php5-mysql phpmyadmin
```

Luego el siguiente paso consiste en configurar el archivo php.ini.

```
# sudo nano/etc/php5/apache2/php.ini
```

Procedemos a colocar antes de la línea “Dynamics Extensions” lo siguiente:

extension=mysql.so

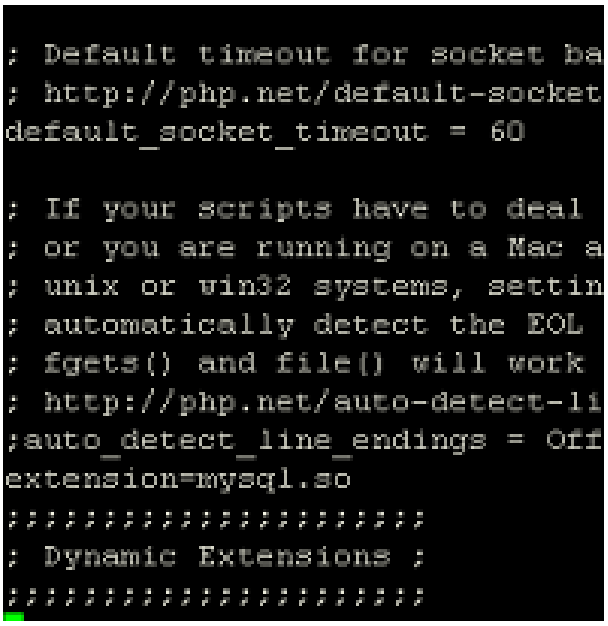


Figura 4.2 Dynamics Extensions

Finalmente ejecutamos los siguientes comandos y reiniciamos el servicio de apache.

sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf.d/phpmyadmin.conf

sudo /etc/init.d/apache2 reload

Luego de haber instalado todo, probamos su funcionamiento, para esto creamos un archivo en la ruta /var/www llamado testphp.php que incluye el comando “<?php phpinfo(); />”. Por último introducimos en el navegador el URL como se puede observar en la siguiente Figura.

Figura 4.3 testphp.php

El último paso a realizar es la prueba de operatividad del phpmyadmin, colocando el URL que se puede observar en la siguiente Figura.

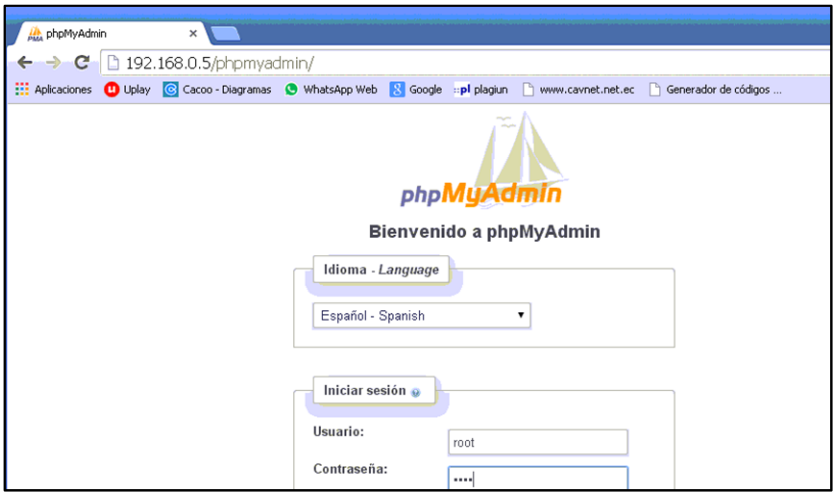


Figura 4.4 Prueba de operatividad del MySQl

Si todo ha funcionado de manera correcta, podremos observar inmediatamente toda la información brindada por MySql, como por ejemplo: base de datos, tablas, datos registrados, etc.

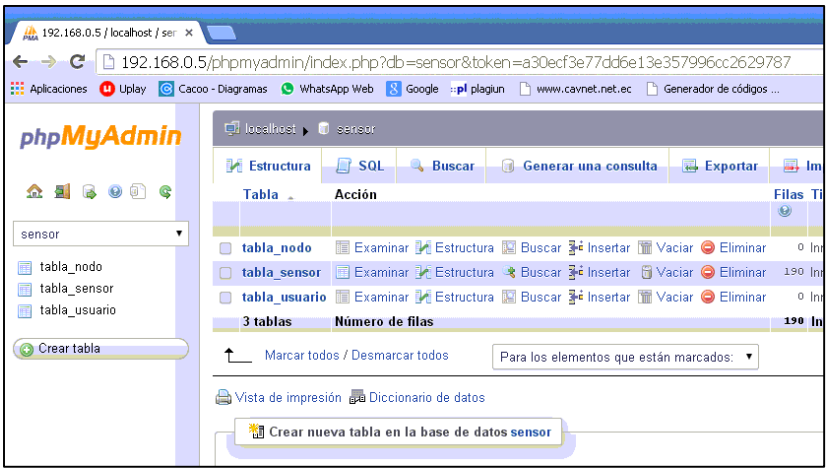


Figura 4.5 visualización de la bases de datos y tablas

Para finalizar con la instalación nos vamos a la ruta /etc/apache2/apache2.conf y agregamos lo siguiente:

Include /etc/phpmyadmin/apache.conf

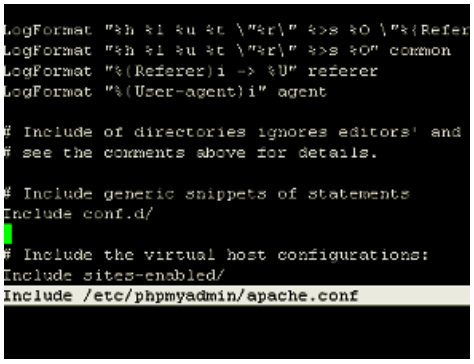


Figura 4.6 Archivo de configuración Apache.conf

Luego reiniciamos el servicio de apache2.

/etc/init.d/apache2 restart

Figura 4.7 proceso de reinicio del servicio Apache

4.2.2. Implementación del Intérprete Python

Python es el pilar fundamental de este nuestro proyecto, ya que su lenguaje de alto nivel nos permite interactuar directamente con el hardware perteneciente y agregado a la minicomputadora Raspberry PI.

El proceso de instalación se detalla a continuación:

Primero se procede con la descarga de la versión de Python acorde a las necesidades del programador con el siguiente comando.

sudo apt-get install python-dev python-rpi.gpio

Una vez hecho esto, el lenguaje de programación queda listo para ser usado (ver ANEXO II), luego creamos los archivos que vamos a programar, en nuestro caso creamos los archivos .py en la carpeta var/etc/www/proyecto

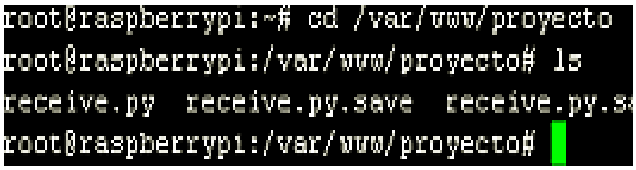


Figura 4.8 Rutas de Script del proyecto

Para saber si toda la instalación se realizó con éxito, procedemos a ejecutar un archivo con extensión ".py" colocando en la interfaz por línea de comando la siguiente sentencia "python nombre_del_alchivo" tal como indica la siguiente Figura.

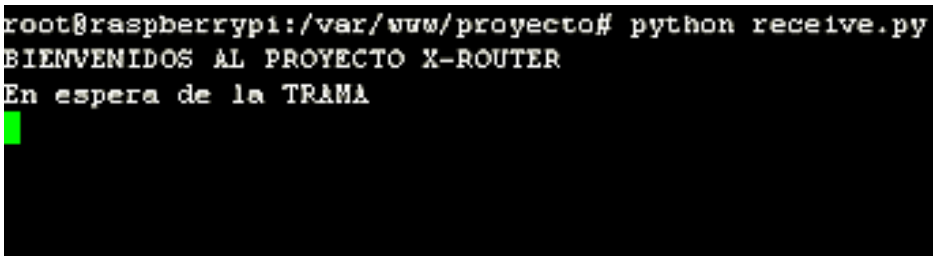


Figura 4.9 Prueba de funcionamiento de Python con el Script de recepción

4.2.3. Implementación del Motor de Base Datos MySql

Una parte de suma importancia es el motor de base de datos MySql ya que será el encargado de almacenar los datos que serán enviados por los sensores al enrutador principal.

Para instalar el servicio MySql utilizamos los siguientes comandos:

```
# apt-get install mysql-server
# apt-get install php5-mysql
```

Una vez realizado la instalación del servicio procedemos a ingresar al sistema utilizando el siguiente comando:

```
# mysql –p –u root
```

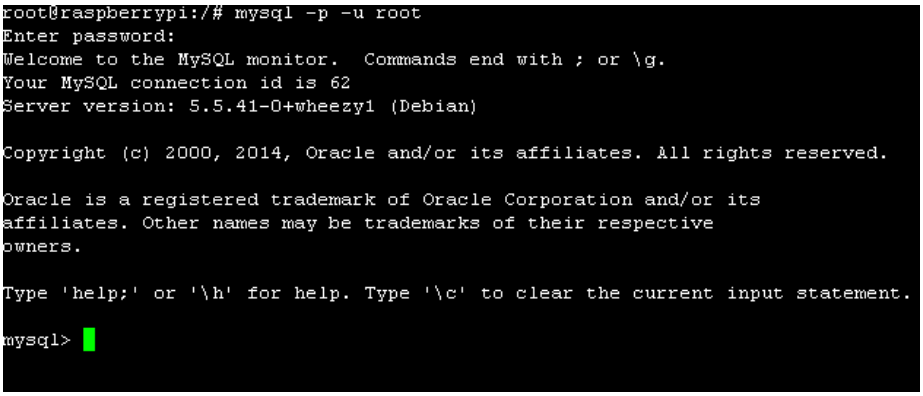


Figura 4.10 Ingreso a la interfaz por línea de comandos MySql

Después que nos autenticamos procedemos a la creación de la base de datos junto con las distintas tablas necesarias para nuestro proyecto.

En las siguientes ilustraciones se puede observar la existencia de la base de datos “sensor” junto con sus respectivas tablas.

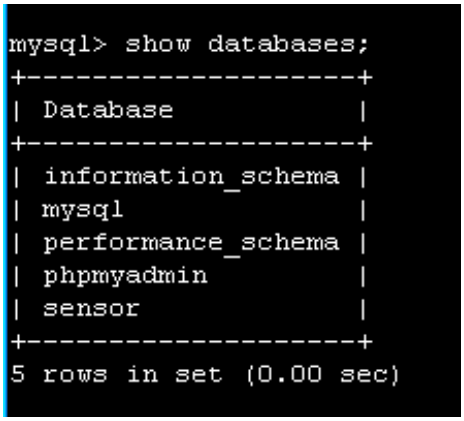


Figura 4.11 lista de base de datos

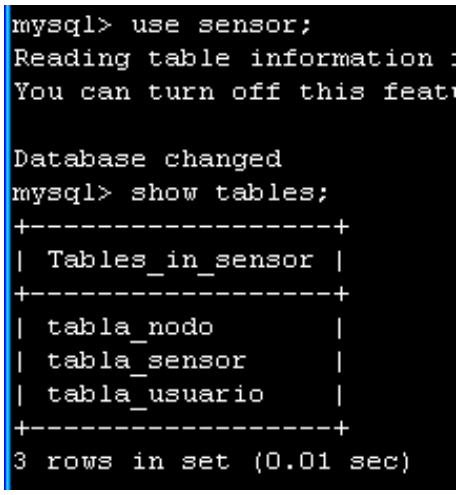


Figura 4.12 listado de Tablas pertenecientes a la base sensor

4.3. Configuración de servicios en la nube

4.3.1. Configuración de Base de datos

En la configuración de los servicios brindados por el hosting SITEGROUND, se creó una base de datos llamada sensor con una tabla que almacenará los datos recolectados por nuestra solución tecnológica.

Ingresamos al hosting y accedemos a la base de datos en la nube con MySql

Figura 4.13 sitio web del hosting

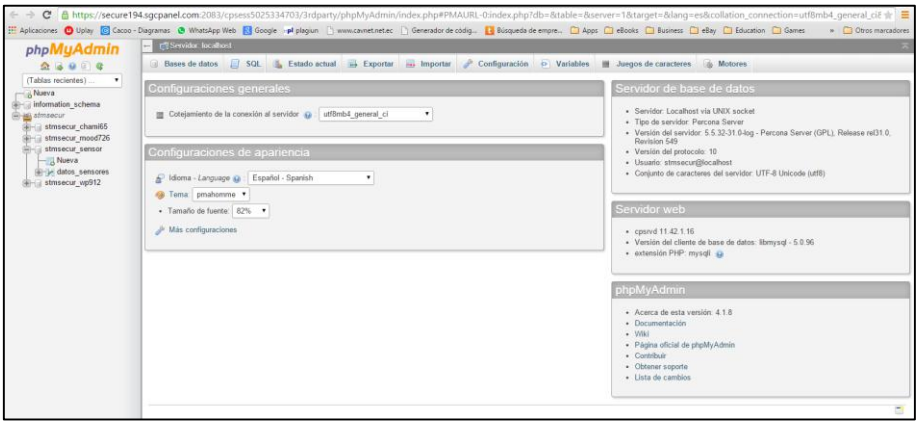


Figura 4.14 Servicio de MySql en la Nube de Siteground

En el panel de SQL ingresamos las siguientes sentencias para la creación de la base de datos

```
28
29  CREATE TABLE IF NOT EXISTS `datos_sensores` (
30  `id` bigint(20) unsigned NOT NULL,
31    `id_sensor` int(11) NOT NULL,
32    `fecha_registro` varchar(20) COLLATE utf8_spanish_ci NOT NULL,
33    `dato` float NOT NULL,
34    `fecha_almacenado` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP
35  ) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci AUTO_INCREMENT=2 ;
36
37  --
38  -- Volcado de datos para la tabla `datos_sensores`
39  --
40
41  INSERT INTO `datos_sensores` (`id`, `id_sensor`, `fecha_registro`, `dato`, `fecha_almacenado`) VALUES
42  (1, 2, '2015-04-21', 20, '2015-04-21 15:40:39');
43
44  --
45  -- Índices para tablas volcadas
46  --
```

Figura 4.15 Creando la Tabla y subiendo datos pruebas

```
51    ALTER TABLE `datos_sensores`
52     ADD PRIMARY KEY (`id`), ADD UNIQUE KEY `id` (`id`);
53
54  --
55    -- AUTO_INCREMENT de las tablas volcadas
56    --
57
58  --
59    -- AUTO_INCREMENT de la tabla `datos_sensores`
60    --
61    ALTER TABLE `datos_sensores`
62    MODIFY `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=2;
63    /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
64    /*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
65    /*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
66
```

Figura 4.16 Definiendo la llave primaria de la tabla

4.4. Interfaz de acceso al usuario

La aplicación web para la administración de datos de la red Zigbee está basado en php, javascript, html y ccs. Cuenta con una interfaz de autenticación para brindar mayor seguridad e integridad a los datos.

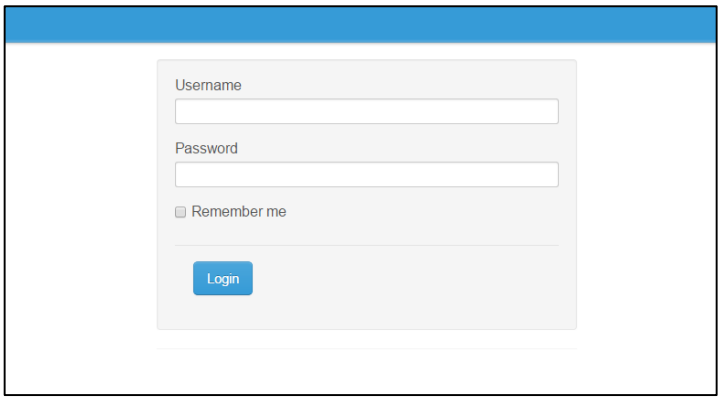


Figura 4.17 Interfaz de autenticación de la interfaz de usuario.

La aplicación web cuenta con dos roles de acceso, el uno que es de administrador y el otro de operador. Cada uno con sus respectivos privilegios para acceder a las distintas opciones de la aplicación.

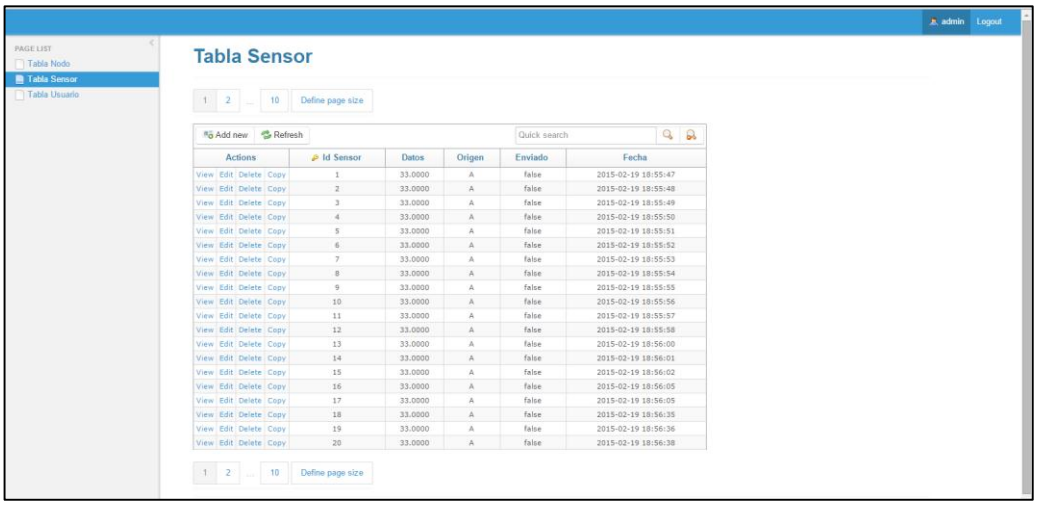


Figura 4.18 Ingreso a la aplicación con el rol de administrador

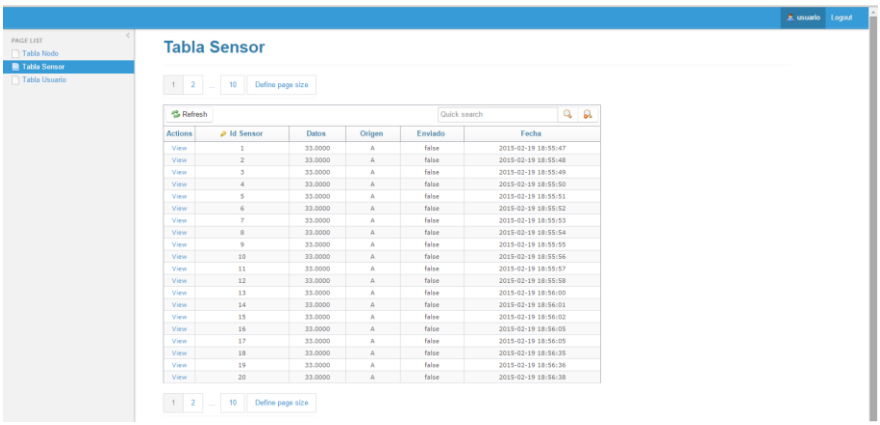


Figura 4.19 Ingreso a la aplicación con el rol de operador

4.5. Pruebas de Funcionamiento del dispositivo

La primera prueba que se realiza es la visual, donde observamos que los leds incluidos en la minicomputadora Raspberry pi se enciendan.

Figura 4.20 Verificación de leds y encendido de la minicomputadora Raspberry pi

A continuación se procede con el escaneo por la red para poder identificar si la minicomputadora Raspberry Pi adquirió una IP del segmento del cual se encuentre el DHCP al cual le está realizando la solicitud de petición. Esta prueba la realizamos con una herramienta de código abierto llamada Angry IP Scanner [21].

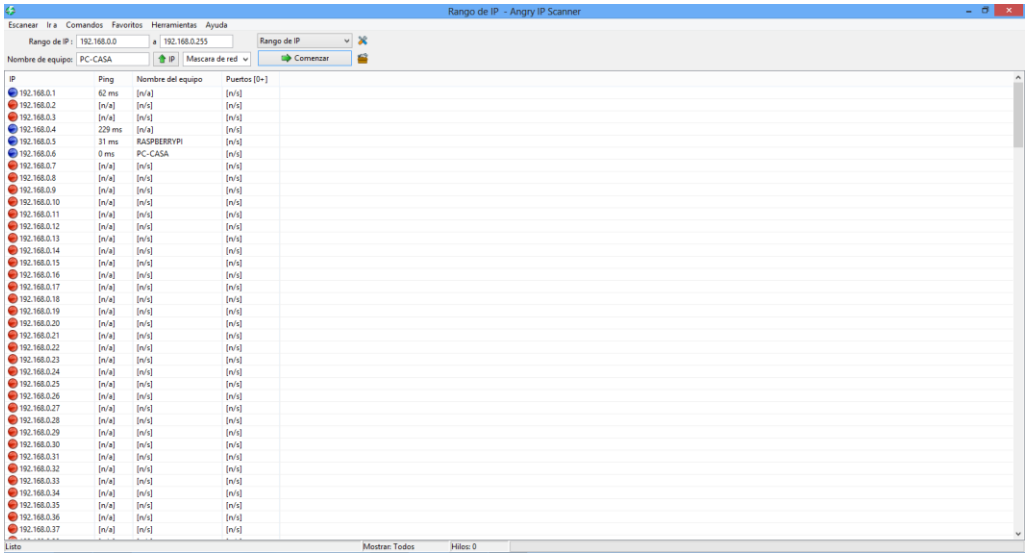


Figura 4.21 Búsqueda de la IP asignada al Raspberry PI

Luego realizamos una prueba de ping hacia la IP de la minicomputadora Raspberry Pi con el CMD de Windows o Terminal de Linux, con la finalidad de verificar la comunicación con el equipo.

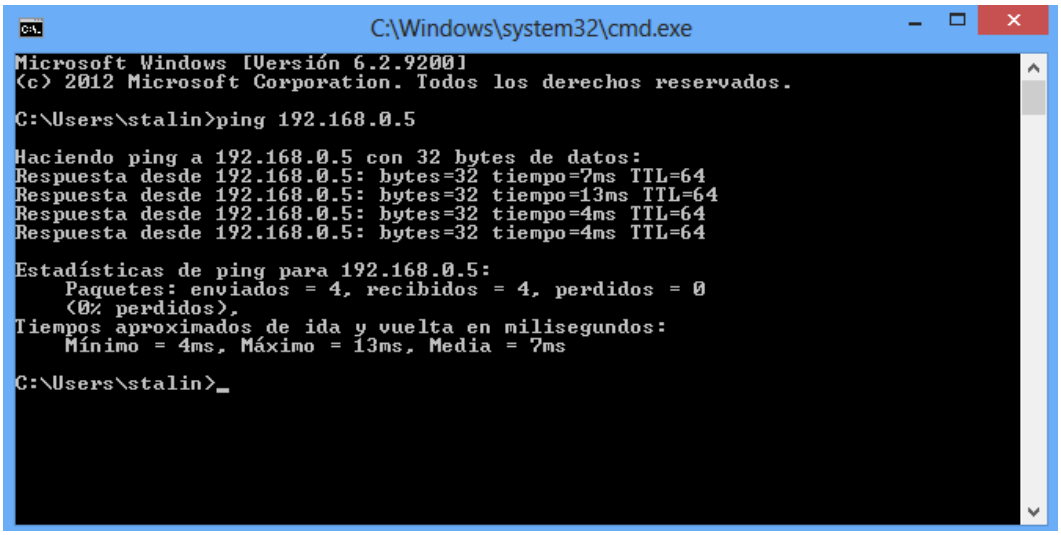


Figura 4.22 Prueba de Ping hacia la IP del Raspberry PI

4.6. Pruebas de Comunicación y transmisión de datos

Este proceso inicia con la ejecución del script de prueba en cada una de las computadoras que van a simular un sensor de temperatura. Una vez iniciado los scripts, estos comenzarán a enviar datos de pruebas al Xbee Coordinador.

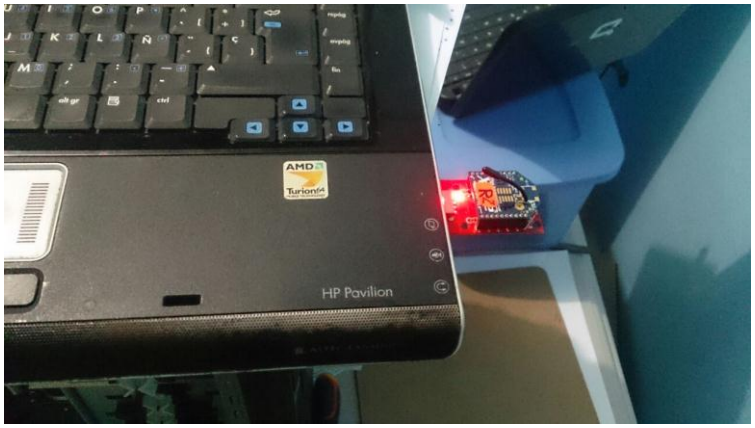


Figura 4.23 El equipo se encuentra enviando datos hacia el coordinados.

Si la comunicación efectuada entre el Xbee Router/Dispositivo final y el Coordinador es la correcta inmediatamente se recibirá el siguiente mensaje de respuesta en la pantalla o interfaz del sensor:

```
#   root@PC:/home/stalin/Público# python send_xbee1.py
#   {'\x88': {'name': 'at_response',
#   'parsing': [('parameter', <function <lambda> at 0xb715ae9c>),
#   ('parameter', <function <lambda> at 0xb715aed4>)],
#   'structure': [{'len': 1, 'name': 'frame_id'},
#   {'len': 2, 'name': 'command'},
#   {'len': 1, 'name': 'status'},
#   {'len': None, 'name': 'parameter'}]},
#   '\x8a': {'name': 'status', 'structure': [{'len': 1, 'name': 'status'}]},
#   '\x8b': {'name': 'tx_status',
#   'structure': [{'len': 1, 'name': 'frame_id'},
#   {'len': 2, 'name': 'dest_addr'},
#   {'len': 1, 'name': 'retries'},
#   {'len': 1, 'name': 'deliver_status'},
#   {'len': 1, 'name': 'discover_status'}]},
#   '\x90': {'name': 'rx',
#   'structure': [{'len': 8, 'name': 'source_addr_long'},
#   {'len': 2, 'name': 'source_addr'},
#   {'len': 1, 'name': 'options'},
```

```
#    {'len': None, 'name': 'rf_data'}]},
#    '\x91': {'name': 'rx_explicit',
#    'structure': [{'len': 8, 'name': 'source_addr_long'},
#    {'len': 2, 'name': 'source_addr'},
#    {'len': 1, 'name': 'source_endpoint'},
#    {'len': 1, 'name': 'dest_endpoint'},
#    {'len': 2, 'name': 'cluster'},
#    {'len': 2, 'name': 'profile'},
#    {'len': 1, 'name': 'options'},
#    {'len': None, 'name': 'rf_data'}]},
#    '\x92': {'name': 'rx_io_data_long_addr',
#    'parsing': [('samples', <function <lambda> at 0xb715ae64>)],
#    'structure': [{'len': 8, 'name': 'source_addr_long'},
#    {'len': 2, 'name': 'source_addr'},
#    {'len': 1, 'name': 'options'},
#    {'len': None, 'name': 'samples'}]},
#    '\x95': {'name': 'node_id_indicator',
#    'structure': [{'len': 8, 'name': 'sender_addr_long'},
#    {'len': 2, 'name': 'sender_addr'},
#    {'len': 1, 'name': 'options'},
#    {'len': 2, 'name': 'source_addr'},
#    {'len': 8, 'name': 'source_addr_long'},
```

```
#    {'len': 'null_terminated', 'name': 'node_id'},
#    {'len': 2, 'name': 'parent_source_addr'},
#    {'len': 1, 'name': 'device_type'},
#    {'len': 1, 'name': 'source_event'},
#    {'len': 2, 'name': 'digi_profile_id'},
#    {'len': 2, 'name': 'manufacturer_id'}]},
#    '\x97': {'name': 'remote_at_response',
#    'parsing': [('parameter', <function <lambda> at 0xb715af0c>)],
#    'structure': [{'len': 1, 'name': 'frame_id'},
#    {'len': 8, 'name': 'source_addr_long'},
#    {'len': 2, 'name': 'source_addr'},
#    {'len': 2, 'name': 'command'},
#    {'len': 1, 'name': 'status'},
#    {'len': None, 'name': 'parameter'}]}}
#    A24
#    send data
```

Si recibimos esta trama con toda esta información se da por entendido que la comunicación entre los Xbee está totalmente funcional.

4.7. Pruebas de Conexión con los servicios en la nube

La verificación de la conexión con los servicios en la nube la realizamos bajo peticiones HTTP. El primer paso es realizar de manera manual una petición para

verificar si se tiene una respuesta positiva por parte del servidor web, utilizando el siguiente URL:

http://stmsecure.com/xbee/guardar_datos_sensor.php?id_sensor=2&fecha_registro=2015-04-21&dato=20

Si el servidor funciona de manera correcta, al momento de realizar la petición recibiremos un mensaje de confirmación con la palabra “TRUE”.

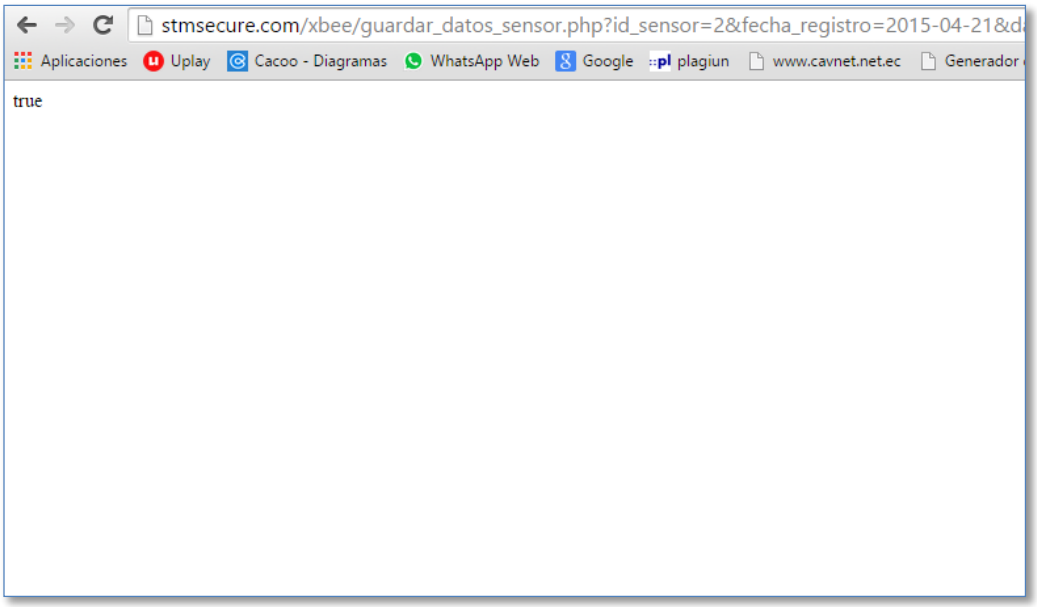


Figura 4.24 Confirmación de Petición manual

Después de haber confirmado de manera manual el funcionamiento del servicio en la nube, procedemos a realizar la prueba desde el enrutador, para esto ejecutamos el script respectivo.

En la siguiente Figura podemos observar el funcionamiento del script que se encarga de enviar los datos hacia el hosting Siteground.

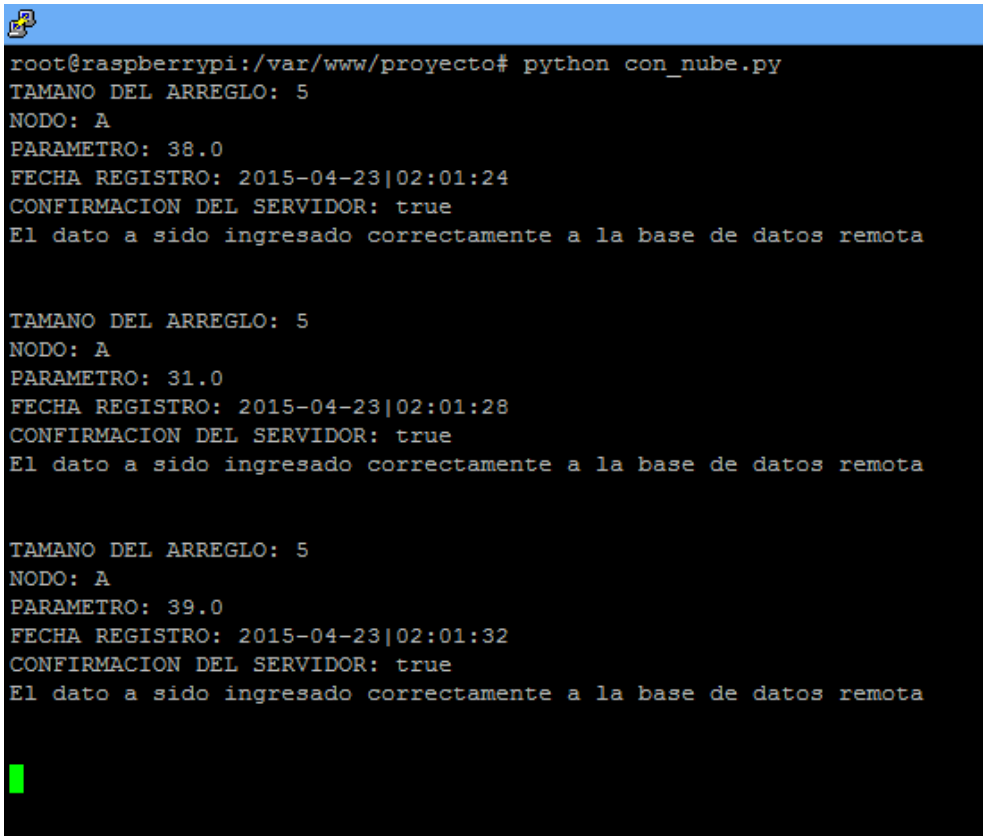


Figura 4.25 Muestra de mensaje de confirmación de recepción de datos en la nube

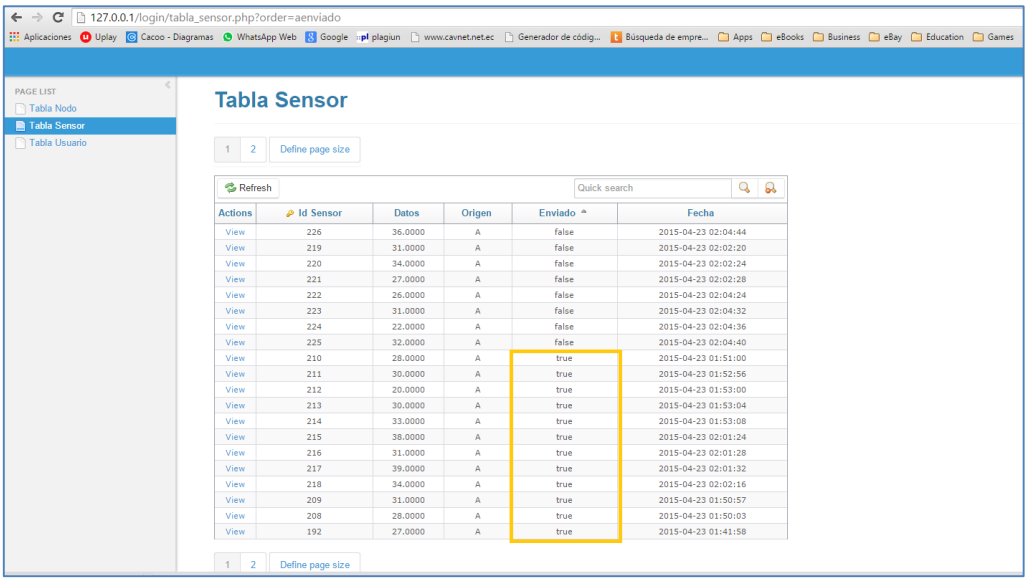


Figura 4.26 Confirmación de cambios realizados en la base de datos

4.8. Pruebas de estrés

Las pruebas que sometimos a la solución tecnológica son básicamente en la trasmisión y recepción de los datos, con la finalidad de detectar algún punto en que se pierda la comunicación o se llegué a saturar el procesador de la minicomputadora Raspberry Pi.

Para la prueba se utilizó 2 sensores Xbee que funcionaron al mismo tiempo. Ambos sensores enviaban datos al receptor con dos segundos de diferencia, ocasionando pérdidas de tramas en algunos instantes.

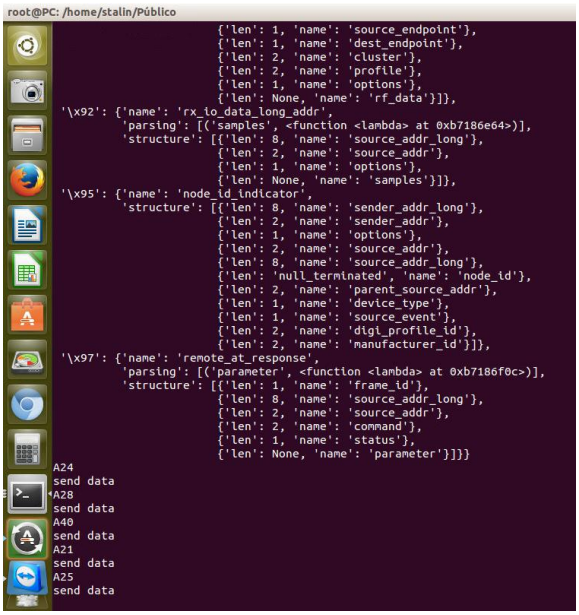


Figura 4.27 Envío de datos desde el sensor A

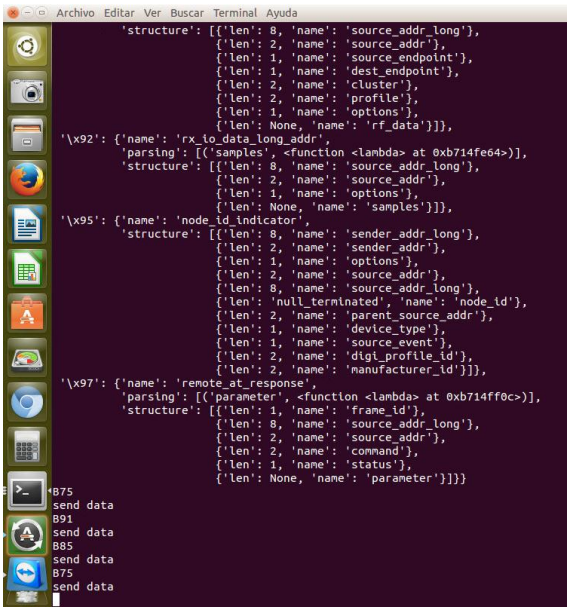


Figura 4.28 Envío de datos desde el Sensor B

4.9. Análisis de resultados

En la prueba de estrés se pudieron obtener los siguientes resultados:

Tabla 1 Resultados de la prueba de estrés transmisión - recepción

SENSOR	FRAME x min (ENAVIADOS)	RECIBIDOS	PERDIDOS	% DE PERDIDA DE FRAMES	% DE EFECTIVIDAD
SENSOR A	10	8	2	25	75
SENSOR B	8	5	3	60	40

Se puede notar claramente que en el proceso de recolección de datos existe una pérdida considerable de las tramas. Esto podría ser debido a la capacidad reducida de procesamiento y memoria que posee la minicomputadora Raspberry pi.

El proceso de almacenaje de los datos recolectados en el SGBD MySql local tuvo como porcentaje de efectividad el 100%.

Tabla 2 Porcentaje de efectividad de envío a base de datos local

SENSOR	FRAME RECEPTADOS CON ÉXITO	RECIBIDOS LOCALMENTE (mysql)	PERDIDOS	% DE PERDIDA DE FRAMES	% DE EFECTIVI-DAD	TIEMPO
SENSOR A	8	8	0	0	100	1 MIN
SENSOR B	5	5	0	0	100	1 MIN

De igual manera el proceso de envío de los datos hacia la nube tuvo como porcentaje de efectividad el 100%.

Tabla 3 Porcentaje de efectividad de envío a base de datos Siteground Mysql

SENSOR	FRAME RECEPTADOS CON ÉXITO	RECIBIDOS EN SITEGROUND (mysql)	PERDIDOS	% DE PERDIDA DE FRAMES	% DE EFECTIVIDAD	TIEMPO
SENSOR A	**8**	**8**	**0**	**0**	**100**	**1 MIN**
SENSOR B	**5**	**5**	**0**	**0**	**100**	**1 MIN**

Se puede apreciar que conforme el tiempo transcurre, hay un incremento en la perdida de datos, pero la comunicación con los servicios en la nube continua de forma eficiente.

Tabla 4 Análisis General de efectividad transcurrido 1 hora en tiempo

SENSOR	FRAME RECEPTADOS CON ÉXITO	RECIBIDOS LOCALMENTE (mysql)	PERDIDOS	% DE PERDIDA DE FRAMES	% DE EFECTIVIDAD	TIEMPO
SENSOR A	480	480	0	0	100	1H
SENSOR B	300	300	0	0	100	1H
SENSOR	FRAME RECEPTADOS CON ÉXITO	RECIBIDOS EN SITEGROUND (mysql)	PERDIDOS	% DE PERDIDA DE FRAMES	% DE EFECTIVIDAD	TIEMPO
SENSOR A	480	480	0	0	100	1H
SENSOR B	300	300	0	0	100	1H
SENSOR	FRAME x min (ENAVIADOS)	RECIBIDOS	PERDIDOS	% DE PERDIDA DE FRAMES	% DE EFECTIVIDAD	TIEMPO
SENSOR A	600	387	213	35,5	64,5	1H
SENSOR B	480	321	159	33,125	66,875	1H

Una vez obtenida la información procedemos a mostrar las gráficas de Efectividad con respecto al tiempo

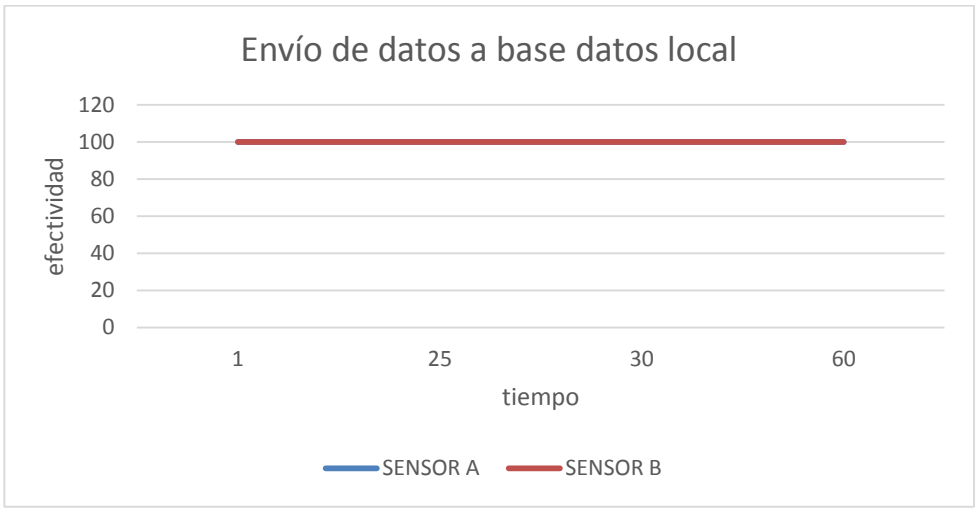


Figura 4.29 Envío de datos a base datos local

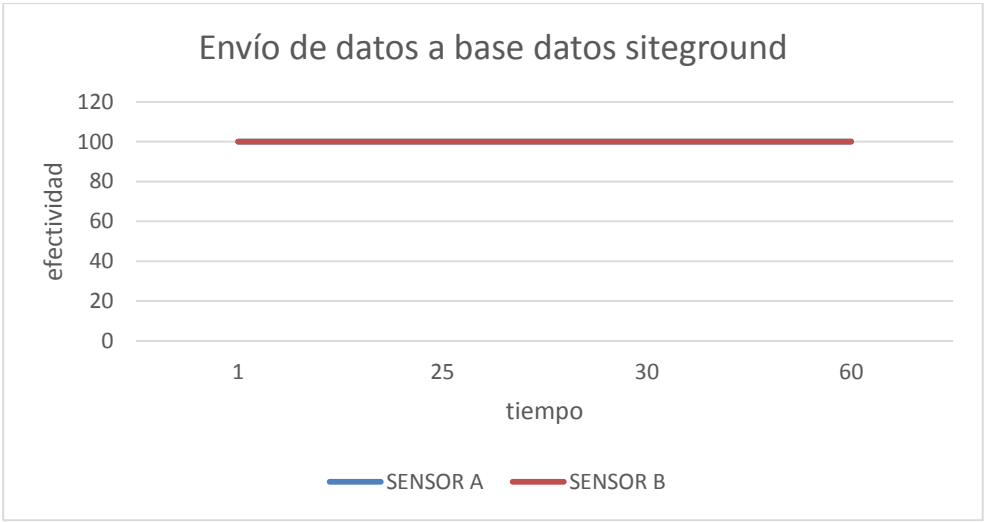


Figura 4.30 Envío de datos a base datos siteground

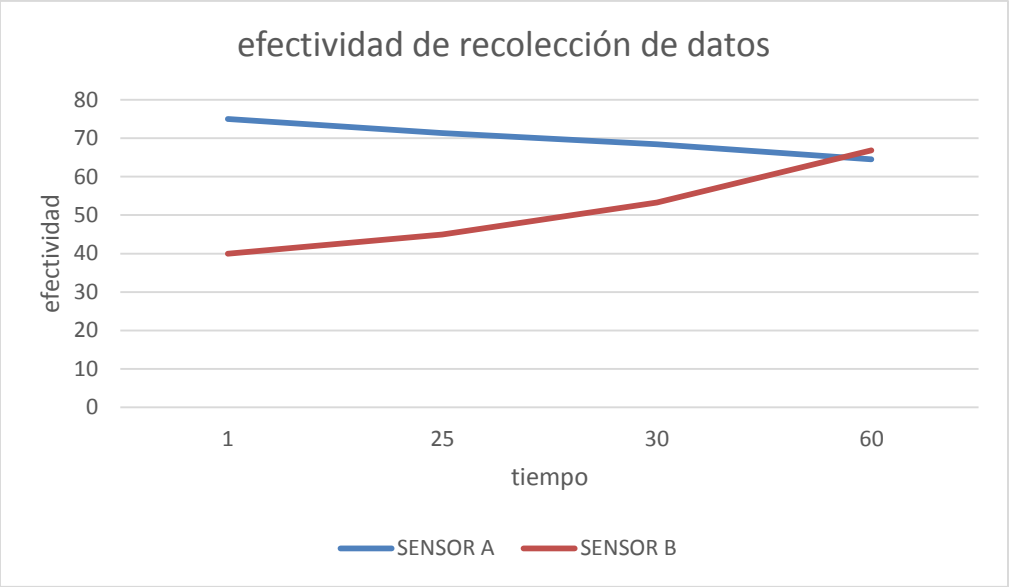


Figura 4.31 Efectividad de recolección de datos

CONCLUSIONES Y RECOMENDACIONES

Con los datos ya expuestos podemos concluir lo siguiente:

1. Al momento de iniciar el proceso de recolección de datos, los recursos de la minicomputadora Raspberry Pi llegan a su máxima capacidad.
2. Debido a la capacidad de nuestro Raspberry PI, la comunicación con los Xbee es muy limitada, así mismo si se conecta más de 3 sensores en el Raspberry PI este puede llegar a colapsar o enviar información errónea debido a los problemas antes mencionados.
3. Los procesos que realizan un consumo significante de memoria y CPU son del servidor web APACHE y del motor de la Base de datos MySql.
4. Con todos los beneficios que se encuentra implícitos en el concepto de computación en la nube, podemos considerar que nuestra propuesta, tendría éxito a gran escala

únicamente mejorando la capacidad de procesamiento y memoria de nuestro Raspberry Pi.

5. Cabe recalcar y destacar que hemos utilizado la tecnología ZigBee para mejorar la arquitectura de comunicación entre dispositivo y dispositivos, ayudando a tener un éxito rotundo al momento del envío, captura, y almacenamiento de datos.

En este proyecto se han abierto una gama de posibilidades de ampliación en temas de investigación, que se deberían tener en cuenta y que consideramos que son muy importantes. Nuestras recomendaciones con respecto a nuestro proyecto son las siguientes:

1. Realizar pruebas con dispositivos de mayor capacidad de procesamiento como es el caso de la minicomputadora Raspberry Pi 2.
2. Plantear nuevos escenarios de pruebas, donde los sensores puedan estar a distintas distancias con respecto al enrutador con la finalidad de medir el porcentaje de efectividad en la comunicación entre ellos.
3. Realizar pruebas con otros tipos de módulos Xbee en el lado del enrutador con la finalidad de medir el porcentaje de efectividad en la recolección de los datos emitidos por los sensores de la red Zigbee.

BIBLIOGRAFÍA

[1] D. Evans, «Internet de las cosas - Cómo la próxima evolución,» Cisco Internet Business Solutions Group (IBSG), Estados unidos, 2011.

[2] L. G. Feng Zhao, Wireless Sensor Networks: An Information Processing Approach, londres: morgan kaufmann publishers inc, 2004, pp. 291-305.

[3] R. P. FOUNDATION, «raspberrypi,» [En línea]. sitio web: http://es.wikipedia.org/wiki/Raspberry_Pi.

[4] ANTONIO y J. M. R. C. MOLINA MARTÍNEZ, Automatización y telecontrol de sistemas de riego, Murcia-España: marcombo ediciones técnicas, 2010, p. 371.

[5] libelium, «libelium,» libelium, 01 01 2006. [En línea]. sitio web: http://www.libelium.com/libelium-images/generico/wsn-475.png. [Último acceso: 25 04 2015].

[6] R. Faludi, «Building Wireless Sensor Networks,» de *Building Wireless Sensor Networks*, Estados Unidos, O'Reilly, 2010, pp. 268-270.

[7] J. B. M. R. R. D. C. R. T. R. R. S. A. S. C. Julio Barbancho Concejero, Redes locales, Madrid (España): paraninfo ciclos formativos, 2014, p. 236.

[8] tbagrosensor, http://www.tbagrosensor.com/, 2014. [En línea]. sitio web: http://www.tbagrosensor.com/images/Redes%20de%20Sensores/Red%20Zigbee.jpg. [Último acceso: 18 04 2015].

[9] M. R. a. S. Wallace, Getting Started with Raspberry Pi, United States of America.: O'Reilly Media, Inc., 2013.

[10] comohacer.eu, «comohacer.eu,» comohacer.eu, 2014. [En línea]. sitio web: http://comohacer.eu/comparativa-y-analisis-raspberry-pi-vs-competencia/.

[11] retro-kit, «http://www.retro-kit.co.uk/,» retro-kit, 2013 11 10. [En línea]. sitio web: http://www.retro-kit.co.uk/page.cfm/content/Control-IT-buffer-box/. [Último acceso: 2015 04 25].

[12] didactic, «didactic,» [En línea]. sitio web: http://www.ididactic.com/edblog/wp-content/uploads/2013/10/351321-raspberry-pi.jpg.

[13] O. C. Uceda, «Desarrollo Web con PHP: Aprende PHP paso a paso,» Chiclayo Peru, 2013.

[14] D. Barcia, «maestros del web,» 08 11 2003. [En línea]. sitio web: http://www.maestrosdelweb.com/introcss/.

[15] D. C. J. d. Parga, Cloud computing: retos y oportunidades, Madrid: Fundación Ideas, 2011.

[16] siteground, «siteground,» [En línea]. sitio web: https://www.siteground.com/.

[17] sqlmaestro.com, «sql maestro,» 2014. [En línea]. sitio web: https://www.sqlmaestro.com/products/mysql/phpgenerator/.

[18] Python.org, «python,» 02 enero 2014. [En línea]. sitio web: https://pypi.python.org/pypi/MySQL-python.

[19] R. Barra, «placer digital,» 18 10 2011. [En línea]. sitio web: http://placerdigital.net/que-es-ubuntu-por-que-usarlo-y-como-probarlo/.

[20] libelium, «cooking-hacks,» 2006. [En línea]. sitio web: http://www.cooking-hacks.com/.

[21] angryziber, «Angry IP Scanner,» [En línea]. sitio web: http://angryip.org/.

ANEXOS

ANEXO I

Script Muestra para Sensores

```
#! /usr/bin/python

import time

from xbee import XBee, ZigBee
import serial
from random import randint

#seleccion de puerto serial

PORT = '/dev/ttyUSB0'
BAUD_RATE = 9600

# aperturando puertos serial port
ser = serial.Serial(PORT, BAUD_RATE)

# Crear el API object
xbee = ZigBee(ser,escaped=True)
import pprint
```

```
pprint.pprint(xbee.api_responses)


# define direccion destino o mac address del coordinador
DEST_ADDR_LONG = "\x00\x13\xA2\x00\x40\xB9\x71\xB9"


# Continuously read and print packets
while True:
    try:
        prueba = "A" + str(randint(20,40)) #captura de la informacion del sensor en
esta parte borrar: "A" + str(randint(20,40))
        print prueba
      print "send data"
      xbee.send("tx",data= prueba,dest_addr_long=DEST_ADDR_LONG,
            dest_addr="\xFF\xFE",broadcast_radius="0",options="0",frame_id="1")
      #response = xbee.wait_read_frame()
      #print response
      time.sleep(4)
    except KeyboardInterrupt:
      break


ser.close()
```

ANEXO II

Manual de instalación y configuración

Raspberry pi

Para iniciar con la instalación primero necesitamos lo siguiente:

Placa Raspberry Pi

Tarjeta SD al menos de 4gb

Hub usb

La instalación del sistema operativo es muy simple, primero nos dirigimos a la web de descargas de Raspberry pi (www.raspberrypi.org/downloads) y descargamos el sistema operativo para introducir esa imagen en la tarjeta de memoria.

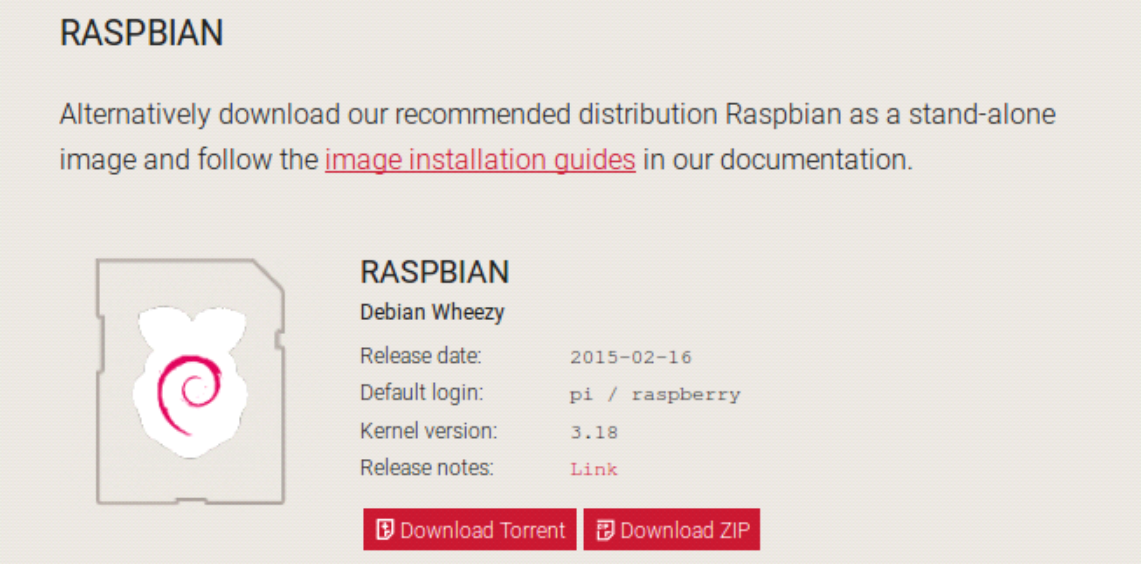


Figura 0.1 Descarga del sistema operativo Raspbian

Una vez descargado el iso del sistema operativo vamos a necesitar la herramienta Win32 Disk Imager para montar la imagen en la tarjeta SD, el URL donde podemos acceder a este programa es el siguiente:

http://sourceforge.net/projects/win32diskimager/files/latest/download

Ejecutamos el programa y ponemos write para que copie la imagen, una vez terminado el proceso introducimos la tarjeta SD en nuestro Raspberry

Figura 0.2 Inserción de micro-sd

La forma más sencilla para conectar la minicomputadora raspberry Pi es mediante un puerto HDMI. Adicional a esto conectamos un hub usb para usar un teclado y un mouse, quedando como se muestra en la siguiente figura.

Figura 0.3 visualización de encendido de la minicomputadora Raspberry pi

Ingresamos el comando sudo raspi-config para iniciar con la configuración y nos aparecerá la siguiente pantalla:

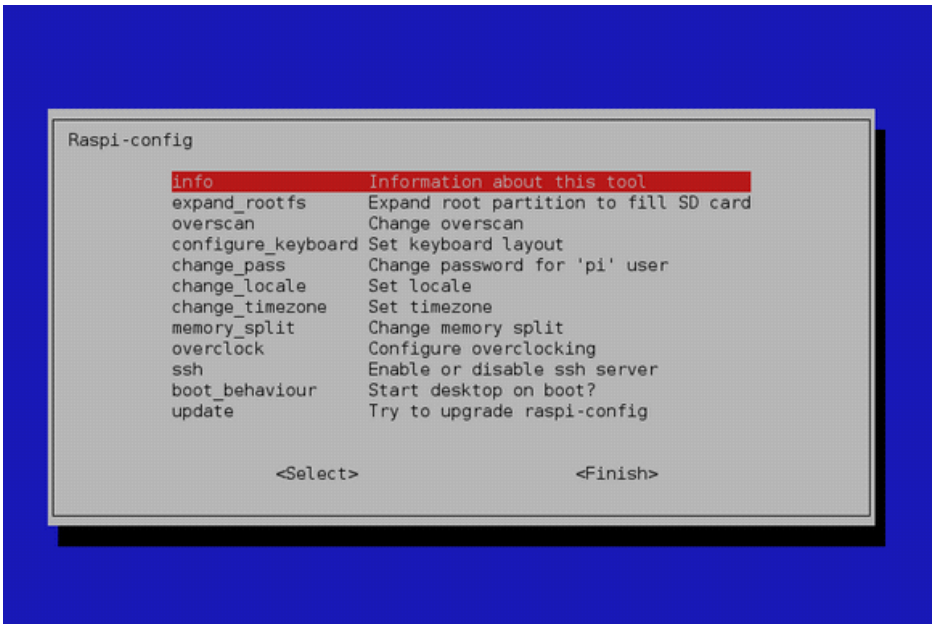


Figura 0.4 Opciones pre-configuración Raspi-config del sistema operativo Raspbian

Configuramos los parámetros que nos aparecen en la pantalla como el keyboard , timezone y la más importante es cambiar el password al user pi. Este nos servirá para acceder al sistema operativo. Por ultimo asignamos una ip manual a nuestro raspberry o lo dejamos por dchp.

El comando para configurar la ip es el siguiente

sudo nano /etc/network/interfaces

Una vez que hayamos ingresado la ip de nuestra red o con dhcp, finaliza el proceso de configuración de la minicomputadora Raspberry Pi.

Posterior a esto reiniciamos la minicomputadora ejecutando el siguiente comando

sudo restart

Otra forma de acceder a nuestra red es por un módulo wifi , para configurarlo tenemos que iniciar en modo gráfico, para esto debemos iniciar sesión y ejecutamos el siguiente comando

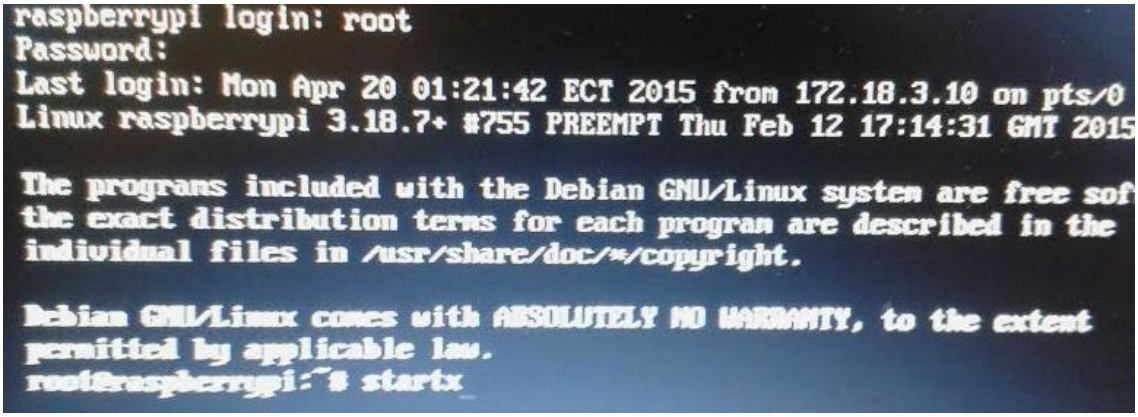


Figura 0.5 Inicio de interfaz gráfica

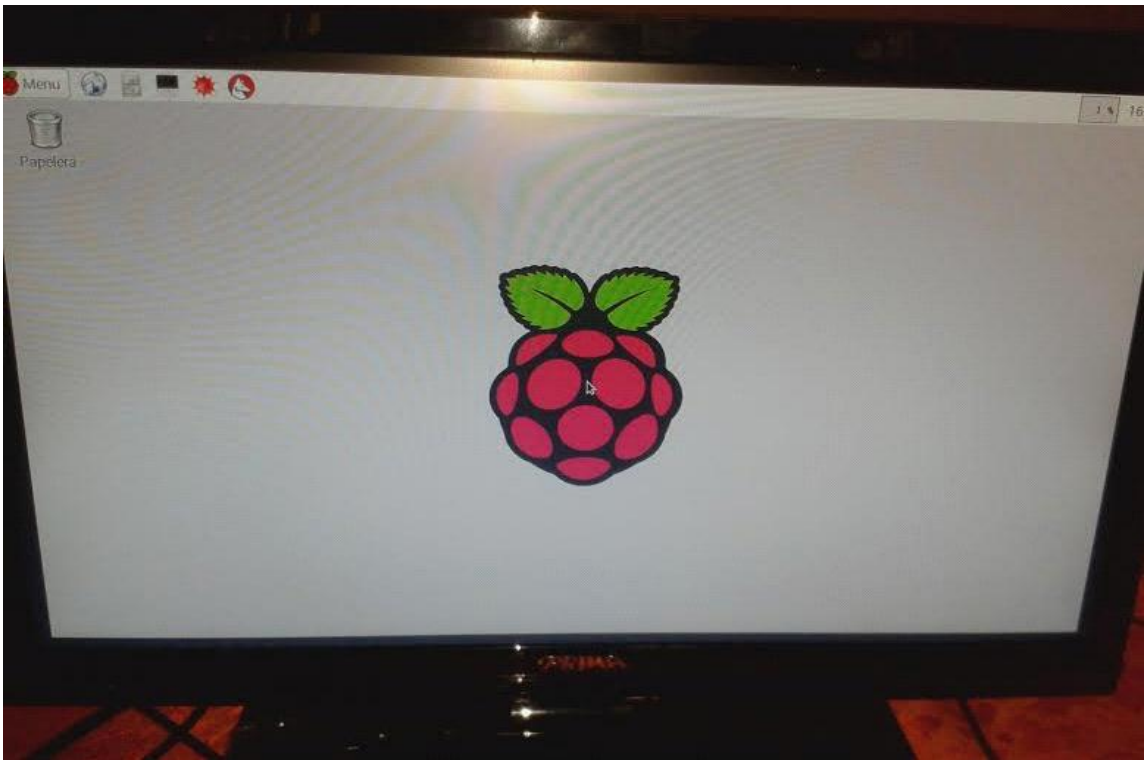


Figura 0.6 Visualización de Interfaz gráfica

Una vez iniciado el entorno de escritorio, damos clic en **menu** y vamos a la opción **preferencias** y buscamos el programa **wpa_gui**

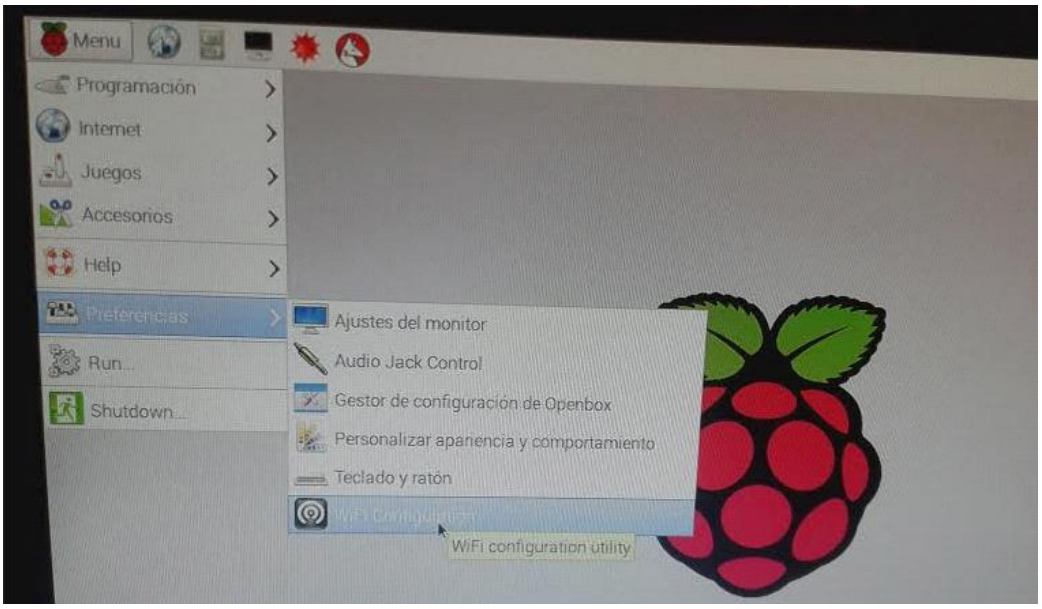


Figura 0.7 WPA software para conexión de redes inalámbricas

Luego se nos abre el programa y hacemos un scan de las redes como se muestra en la siguiente Figura

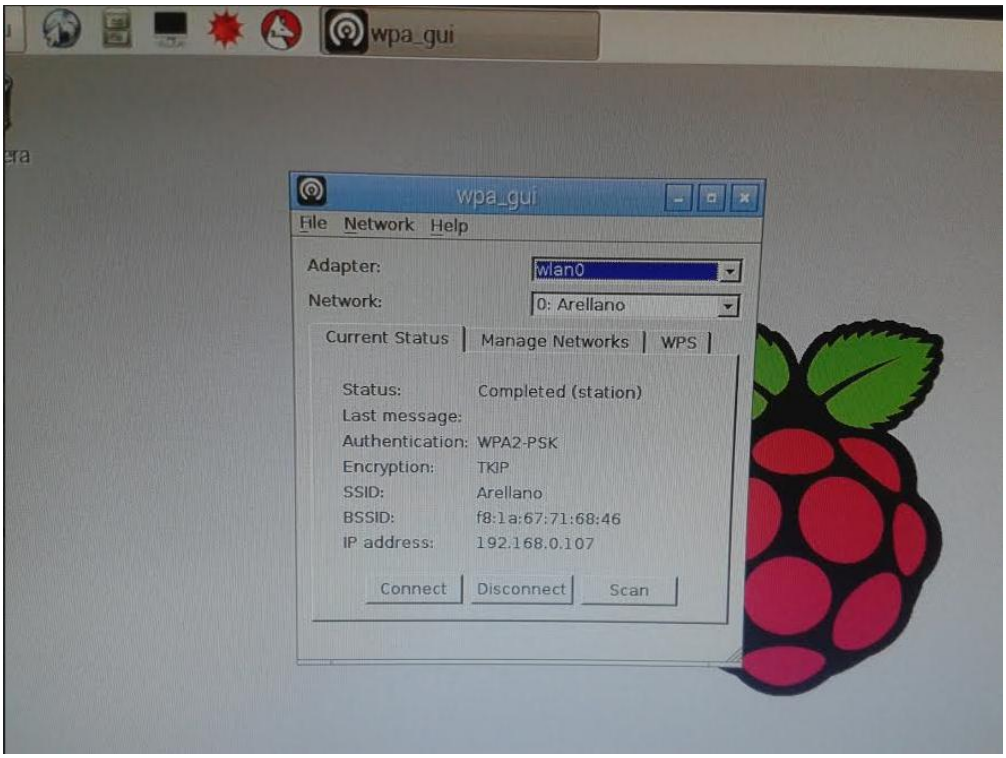


Figura 0.8 Configuración de La red inalámbrica

Por último nos conectamos a la red y ya tenemos configurada nuestra minicomputadora Raspberry Pi

ANEXO III

MANUAL DE CONFIGURACIÓN XBEE POR X-CTU

En este manual se explicará la instalación y configuración del programa x-ctu con los xbee en modo api.

Las herramientas que necesitamos son:

- 1 o 2 xbee de la misma serie

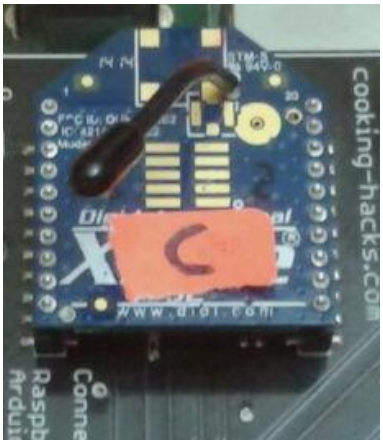


Figura 0.9 Xbee Series 2 configurado como coordinador

- 1 explorer para el xbee

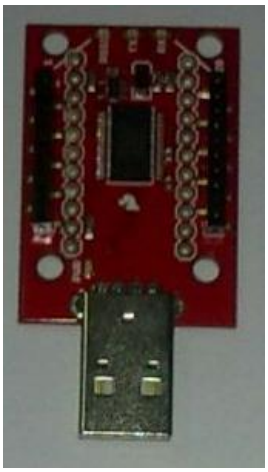


Figura 0.10 XBEEE XPLORER listo para configuración

- Programa x-ctu que lo podemos descargar desde el siguiente URL

  http://www.digi.com/support/productdetail?pid=3352&osvid=57&type=utilities

Una vez descargado el programa procedemos a la instalación, cuando el programa pregunte si quiere actualizarse desde Digi le ponemos si para que descargue todos los firmware de los Xbee.

Iniciamos el programa y buscamos los Xbee previamente conectados en el computador

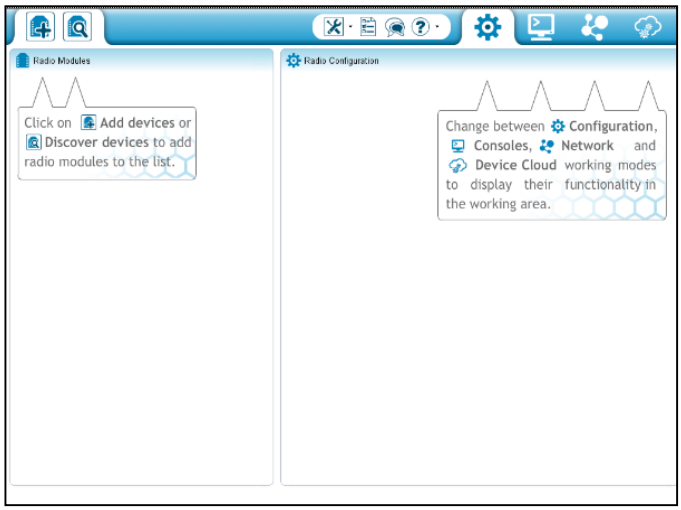


Figura 0.11 Programa XCTU

Seleccionamos el puerto COM para establecer la conexión con el módulo Xbee.

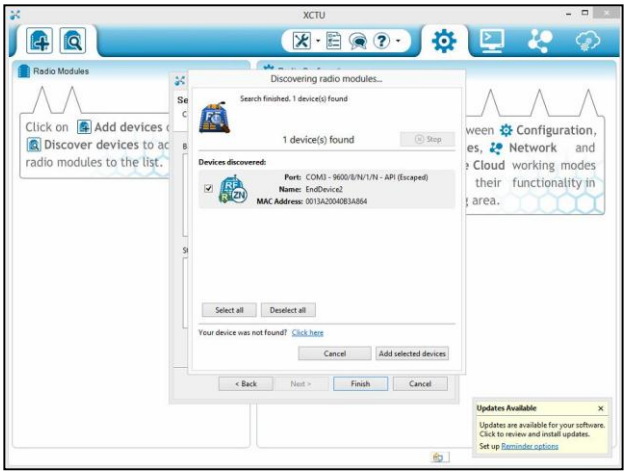


Figura 0.12 Agregando Xbee al XCTU

Le damos clic en la opción de firmware y buscamos la XB24-B en modo API el

Router/End Device

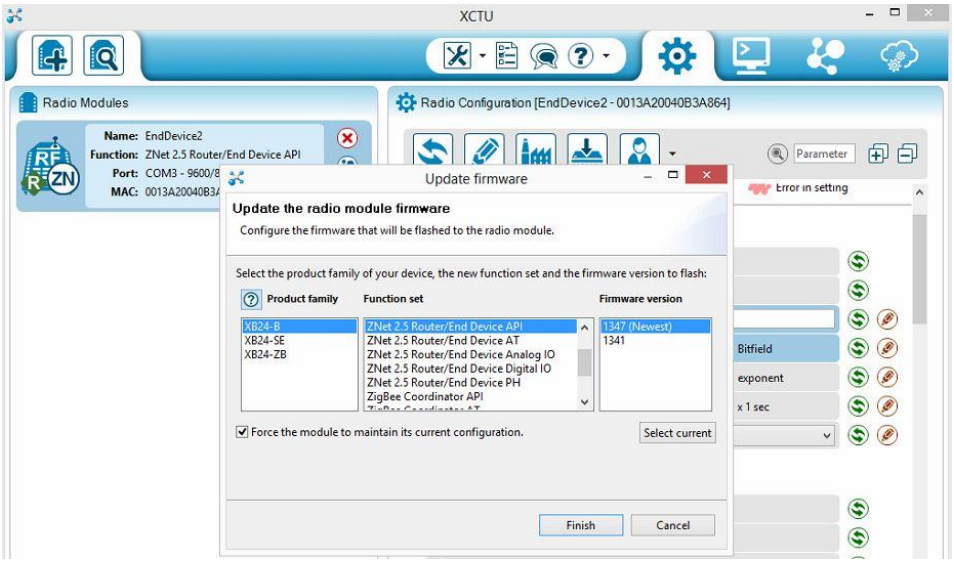


Figura 0.13 Selección del Frimware a usar en cada Xbee

En el modo API nos da más flexibilidad al momento de hacer él envío y recepción de datos atreves de la red ZigBee. Este modo hace que el Xbee espere por una secuencia específica de bytes que le indica que operación tiene que realizar.

Una vez agregados los xbee en el x-ctu configuramos los siguientes parámetros:

- PAN ID, aquí se pone la red en la cual los xbee se van a comunicar.
- DH, ponemos el nodo principal en nuestro caso es el 13A200 del xbee coordinador.
- DL, este es el serial del xbee que se encuentra en la parte de atrás del dispositivo.
- NI, Nombre como queremos diferenciar a nuestros xbee.

P

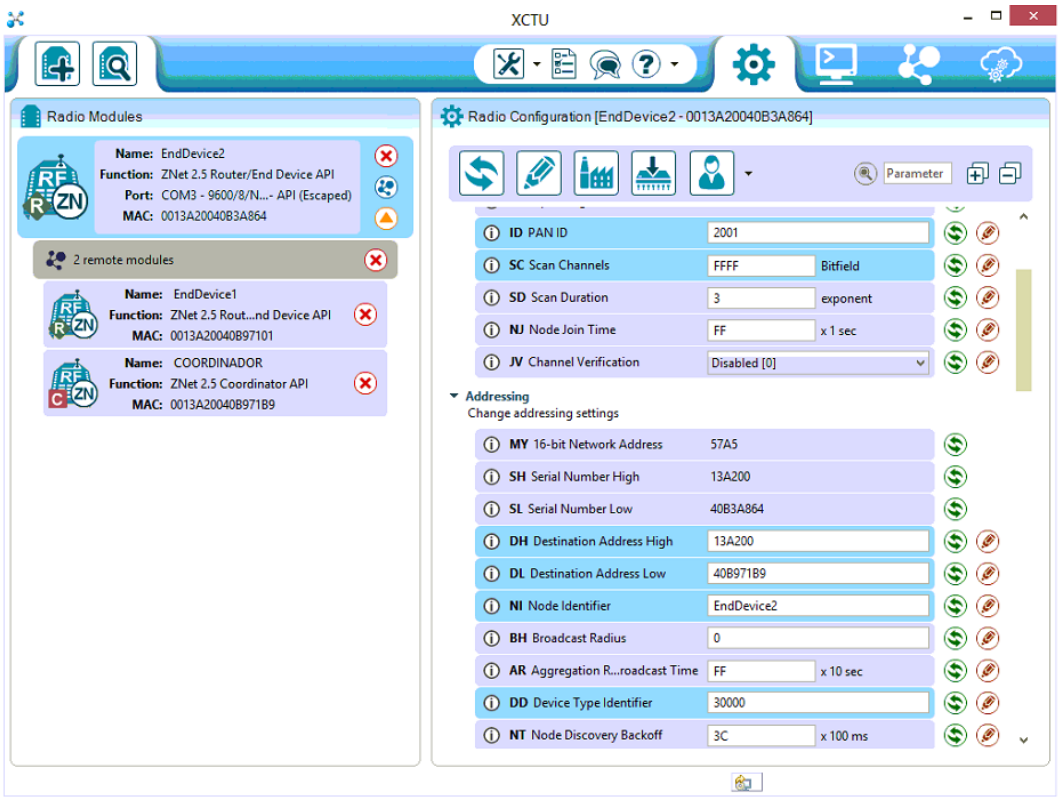


Figura 0.14 Configuración de parámetros Xbee

Q

Por ultimo hacemos una prueba de conexión entre los xbee que se encuentran en la red Zigbee.

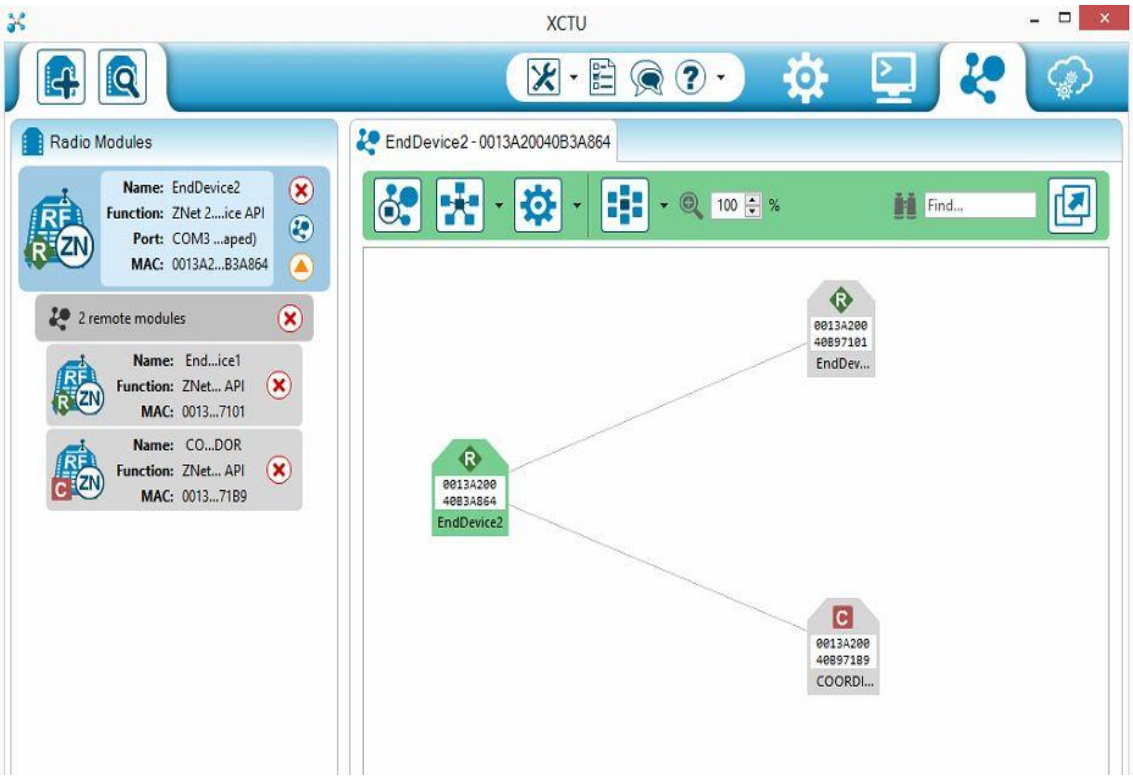


Figura 0.15 Visualización de topologías

ANEXO IV

```
#! /usr/bin/python
from xbee import XBee,ZigBee
import serial
import MySQLdb
#import config
import time
print "BIENVENIDOS AL PROYECTO X-ROUTER"


line = ""
# creando conexion con con la db
db = MySQLdb.connect("localhost","root","root","sensor")
cursor = db.cursor()
PORT = '/dev/ttyAMA0'
BAUD_RATE = 9600
# iniciando la apertura del puerto serial que para el raspberry es
ttyAMA0
ser = serial.Serial(PORT, BAUD_RATE)
# creando un objeto del API XBEE, usamos ZigBee ya que nuestros radios
son XBEE series 2
xbee =ZigBee(ser,escaped=True)
# Iniciando la captura y almacenamiento continúo de información
```

```
while True:
      time.sleep(2)
      try:
            print "En espera de la TRAMA"
            trama = xbee.wait_read_frame()
            print trama
            info = trama['rf_data']
            print "info: " + info
            if trama['rf_data'] != "":
                  info = str (info)
                  print "info" + info
                  destino = info[0]
                  print "destino " + destino
                  data = info.strip("ABCDEFGHIJKLMNOPQRSTUVWXYZ")
                  print "data " + data
                  data = float(data)
                  cursor.execute("""INSERT INTO tabla_sensor(origen,
datos, enviado) VALUES ('%s', '%d', 'false')""" %(destino, data))
                  db.commit()
            else:
                  db.rollback()
                  db.close()
```

```
        except:
                print "no paso"
                ser.close()
                ser = serial.Serial(PORT, BAUD_RATE)
                xbee = ZigBee(ser,escaped=True)
ser.close()
```

ANEXO V

```
__author__ = 'by Stalin T. & Edgar'
import MySQLdb
import httplib
import time
#Create an "API" object

db = MySQLdb.connect("localhost","root","root","sensor")
cursor = db.cursor()
while True:
      time.sleep(5)
      try:
            cursor.execute("SELECT * FROM tabla_sensor WHERE
enviado='false' ORDER BY fecha ASC")
            reg=cursor.fetchone()
            print "TAMANO DEL ARREGLO: " + str(len(reg))
            if len(reg) != 0:
                  id=reg[0]
                  idnodo = str(reg[2])
                  print "NODO: " + idnodo
                  dato = float(reg[1])
                  print "PARAMETRO: " + str(dato)
                  fecha = str(reg[4])
                  fecha = fecha.replace(" ","|")
                  print "FECHA REGISTRO: "+fecha
                  con = httplib.HTTPConnection("www.stmsecure.com")
      con.request("GET","/xbee/guardar_datos_sensor.php?id_sensor=%s&fe
cha_registro=%s&dato=%d" %(idnodo, fecha, dato))
                  respuesta = con.getresponse()
                  data = respuesta.read()
                  print "CONFIRMACION DEL SERVIDOR: " +data
                  con.close()
                  if data=='true':
                        cursor.execute("UPDATE tabla_sensor SET
enviado='true' WHERE enviado='false' ORDER BY fecha ASC LIMIT 1")
                        db.commit()
                        print "El dato a sido ingresado correctamente a
la base de datos remota\n\n"
```

```
            else:
                  print "dato no paso, verificar conexión\n\n"
except:
      time.sleep(5)
      print "envío fallido...\n\n"
```