DISEÑO, IMPLEMENTACIÓN Y EVALUACIÓN DE PROTOTIPO DE SISTEMA REMOTO DE CAPTURA DE DATOS DE PARÁMETROS ELÉCTRICOS EMPLEANDO UN GATEWAY INDUSTRIAL PROGRAMANDO EN LENGUAJE PYTHON

ANDERSON PÁEZ CHANAGÁ

HÉCTOR ANDRÉS QUINTERO GÜIZA

UNIVERSIDAD INDUSTRIAL DE SANTANDER

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

BUCARAMANGA, SANTANDER

2009

DISEÑO, IMPLEMENTACIÓN Y EVALUACIÓN DE PROTOTIPO DE SISTEMA REMOTO DE CAPTURA DE DATOS DE PARÁMETROS ELÉCTRICOS EMPLEANDO UN GATEWAY INDUSTRIAL PROGRAMANDO EN LENGUAJE PYTHON

ANDERSON PÁEZ CHANAGÁ

HÉCTOR ANDRÉS QUINTERO GÜIZA

Trabajo de grado presentado como requisito para optar al título de Ingeniero Electrónico

Director: MIE. JOSÉ DE JESUS RUGELES URIBE

UNIVERSIDAD INDUSTRIAL DE SANTANDER

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

BUCARAMANGA, SANTANDER

2009

Dedico este trabajo a mi madre Maria Edilia por haber sido mi soporte, mi compañía y modelo de persona. Por todo el apoyo brindado durante cada etapa de este proceso de crecimiento y por todo el esfuerzo invertido en mí y en mi formación. Por haberme criado, formado, comprendido, aconsejado y por haberme dado la oportunidad de afrontar los retos de la vida pero inculcándome todos los valores para ser una persona de bien, honesta y correcta.

A mi hermano Yovanny quien junto a mi mama han sido los que me han brindado su apoyo incondicional y son mi motivo de inspiración.

A Dios por darme sabiduría y fortaleza en todo el camino recorrido de mi formación personal.

A mi familia por el interés mostrado en el desarrollo de este proceso.

A mi novia Laura, que llegó en un momento en el que sentía desfallecer por la soledad del corazón, pero que ha sido un gran apoyo en los momentos difíciles y me ha mostrado nuevas cosas por las que vale la pena luchar.

A mis dos amigos de toda la vida, Jhon Hernández y Mauricio Patiño por haberme acompañado y haber compartido conmigo desde nuestra infancia los momentos que me llevan a ser la persona y el profesional que soy.

A mis compañeros de proyecto, Luis, Erik, Héctor, por haberme acompañado y aguantado durante todo este tiempo, pero también por haber compartido momentos que aportan a mi formación.

A mis valiosas amigas, Andréa del Pilar, Diana's, Adriana, Carolina, María Angelica, Denisse, Camila, Johanna, Jenny Rocío por haberme brindado su amistad, porque durante mi vida desde la época de colegio hasta ahora en la universidad me acompañaron y han compartido conmigo momentos muy especiales y porque me han enseñado mucho.

A mis compañeros de la carrera que me acompañaron en esta travesía.

Anderson.

A mis padres por todo su apoyo en los momentos más difíciles de mi vida, el amor infinito que espero retribuir toda mi vida, por todos valores inculcados durante mi formación que necesito siempre.

A mis tías por su amor incondicional, sus consejos y el apoyo espiritual que me entregan cada día. Mis tías que hacen la labor de mamás que adoro con toda el alma, siempre estaré agradecido.

A mi hermano y mis primos por estar conmigo siempre brindándome su amistad y cariño.

A mis amigos por todos los momentos alegres, las risas y desdichas que hemos compartido, Germán, Lisney, Javier, Diana, John, Ximena, Viterbo, Wilson, Andrés, que con muchos más tendrán siempre en mi un amigo incondicional.

Al excelente grupo de trabajo que formamos en el proyecto, Erik, Luis y Anderson. Pr la amistad, el compromiso y la camaradería que aprendimos a forjar día a día en el trabajo.

A los Ingenieros del grupo de investigación CPS por los consejos, correcciones y amistad incondicional.

A Dios por bendecir, guiar mi trabajo y permitirme culminar con éxito las metas que tengo y por todas estas personas tan valiosas que me rodean todos los días.

Héctor Andrés.

CONTENIDO

LISTA DE FIGURAS

RESUMEN

TÍTULO: Conectividad de redes de sensores inalámbricos empleando Gateway industrial programado en lenguaje Python. *

AUTORES: Anderson Páez Chanagá, Héctor Andrés Quintero Güiza. **

PALABRAS CLAVE: Conectividad, Gateway, Python, ZigBee, Prototipo.

DESCRIPCIÓN: Como resultado del proyecto se presenta en este documento la experiencia de diseño, implementación y evaluación de un prototipo de pruebas para la experimentación con redes de sensores inalámbricos (*TESTBED*) utililzando tecnología Zigbee 802.15.4. El sistema emplea un Gateway industrial programado en lenguaje Python, que permite administrar la comunicación desde y hacia los sensores que conforman la red. La programación realizada en este dispositivo permite administrar la red asociando dinámicamente nodos inalámbricos con una aplicación conectada a internet, los datos que se obtienen de los sensores son enviados mediante sockets TCP a un servidor remoto que se encarga de organizar y almacenar la información localmente y en una base de datos SQL, también es posible guardarla temporalmente como respaldo en una memoria USB conectada al dispositivo. Se describen las experiencias de implementación del algoritmo para la medida remota del parámetro RSSI (*Received Signal Strength Indicator*) requerido para la planificación inalámbrica de la red de sensores y de los algoritmos de lectura y almacenamiento de medidas de temperatura, humedad e intensidad luminosa para un grupo de 10 sensores inalámbricos. Se presentan además las posibilidades del sistema implementado para el desarrollo de diversos tipos de experimentos con redes de sensores para futuros trabajos de sistemas automáticos de medición y control de procesos.

* Proyecto de grado.

** Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, Director: MIE. José de Jesús Rugeles Uribe, Grupo de investigación en Conectividad y Procesamiento de Señales CPS

.

ABSTRACT

TITLE: Connectivity of wireless sensor networks using an industrial gateway in python programmed environment. *

KEYWORDS: Connectivity, Gateway, Python, ZigBee, Prototype.

AUTHORS: Anderson Páez Chanagá, Héctor Andrés Quintero Güiza. **

DESCRIPTION: As a result of the project is presented in this paper the experience of design, implementation and evaluation of a WSN testing prototype for the experimentation with wireless sensor networks (Testbed) using ZigBee 802.15.4 technology. The system uses an industrial Gateway programmed in Python language, which allows to manage the communication to and from the sensors that in the network. The programming of this device allows to manage the network associating dynamically wireless nodes with an application connected to the Internet, data obtained from sensors are sent using TCP sockets to a remote server that is responsible for organizing and storing information locally and in a SQL database, you can also save temporarily information as a backup in a USB memory device connected to. It describes the experiences of implementing the algorithm in measure of the remote parameter RSSI (Received Signal Strength Indicator) required for the planning of the wireless network of sensors and algorithms for reading and storage measurements of temperature, humidity and light intensity for a group of 10 wireless sensors. It also presents the possibilities of the system implemented for the development of various types of experiments with sensor networks for future work of automatic measurement systems and process control.

* Degree Project.

** School of Electrical, Electronics and Telecommunications Engineering, Director: MIE. Jose de Jesus Rugeles Uribe, Research Group in Connectivity and Signal Processing CPS.

.

1. INTRODUCCIÓN

En la actualidad, la necesidad de la monitorización y el control de distintos tipos de variables a nivel industrial y en el hogar está en aumento, además de la tendencia hacia la automatización de todos los procesos que impulsan el desarrollo de nuevas tecnologías que puedan soportar las demandas de los usuarios. Uno de los campos que está en constante cambio es el de las telecomunicaciones, pues es el que permite la interacción entre un usuario y el proceso que se está monitorizando. Es en este campo donde los desarrollos tecnológicos han tenido gran impacto y que la tendencia es el manejo de las comunicaciones de forma inalámbrica; las tecnologías con mayor auge son: Bluetooth, WiMax, WiFi, Zigbee entre otras, que permiten ejecutar las mismas tareas que anteriormente se realizaban con conexiones cableadas pero con ventajas como la versatilidad de la implementación en diferentes ambientes y situaciones, costos y en algunos casos mejor desempeño.

Debido al creciente número de aplicaciones que involucran monitorización y control de variables relacionadas con seguridad, control de acceso, control de temperatura, sistemas AMR (Automatic Meter Reading), manejo de carga, entre otros, es necesario entonces recurrir a dichas tecnologías de comunicación inalámbricas; entre las mencionadas, Zigbee se enfoca en procesos donde la prioridad es el control de las variables prescindiendo de la velocidad de transferencia de la información. Entre las características que hacen a Zigbee atractivo para estas tareas se encuentran: la posibilidad de manejar una gran cantidad de nodos (más de 65000), el rango típico de alcance (100m o más), el bajo costo en equipamiento inicial y el bajo consumo de energía.

Se busca con esta investigación evaluar una solución de monitorización de variables utilizando la tecnología Zigbee, con este fin se plantea el montaje de un prototipo de adquisición y almacenamiento de información utilizando módulos Xbee Serie 2 y un Gateway Industrial Connect Port X8 que se programa en lenguaje Python. Se quiere observar si una adquisición de datos sencilla puede ser implementada con estos dispositivos, determinando las capacidades y limitaciones, realizando también una integración con interfaces de almacenamiento (específicamente base de datos), encontrar errores en la transmisión si existen y en general observar el desempeño del prototipo y su estabilidad.

El estudio se divide en varias etapas, primero se tiene una breve explicación de los recursos que se van a utilizar en el TESTBED y sus características (Gateway Connect Port X8, lenguaje Python, y los módulos Xbee), seguido de un análisis de las pruebas para determinar las posibilidades del prototipo en la adquisición de datos de humedad y temperatura, la medición de RSSI y el almacenamiento de la información. Luego el protocolo de consulta implementado y a partir de esto, se presenta el prototipo de adquisición de información. Finalmente los resultados del desempeño del prototipo.

Con los resultados encontrados en la evaluación del TESTBED es posible implementar el prototipo para adquirir información proveniente de la red de módulos Xbee, integrándolo como solución de algunas de las aplicaciones mencionadas anteriormente (sistemas AMR, control de temperatura, control de acceso entre otros), pero se debe tener especial cuidado en la configuración de todo el sistema ya que existe limitantes en cuanto al tamaño del paquete de información que se puede transmitir y el tiempo de envío desde la red hasta el almacenamiento de los datos.

2. HARDWARE Y SOFTWARE DEL TESTBED

2.1. HARDWARE DEL GATEWAY CONNECTPORTX8

Es utilizado un dispositivo Gateway que conecta una red de módulos XBee con internet permitiendo administrar tanto el dispositivo como los módulos de forma remota. Lo más sobresaliente del Gateway ConnectPortX8, es la versatilidad en cuando a su programación y la multifuncionalidad que esto ofrece, posee un interpretador de lenguaje Python además de las librerías básicas de Python.

El Gateway ConnectPortX8 de la empresa DIGI presenta una serie de propiedades donde puede acoplarse efectivamente a diferentes procesos, debido a que permite una programación abierta en lenguaje Python. El dispositivo se acopla a una red ZigBee como coordinador.

Tabla 1. Tabla de comparación de dispositivos Gateway similares

Características	ConnectPortX2	ConnectPortX4	ConnectPortX8
Procesador	NS7520 ARM7	NS9360 ARM9	NS9750 ARM9
Entorno de programación	Python	Python	Python
RAM	8 MB	16 MB	16 MB
FLASH	4 MB	8 MB	8 MB
Puerto Ethernet (RJ-45)	Si y/o Wi-Fi	1	1
Puerto Serial (DB-9)	No	1	1
Puerto USB	No	1	2

El dispositivo compara en la tabla 1 de la página 3, las características más importantes con otros de su clase, la familia ConnectPort de DIGI, cada uno de ellos tiene diferentes especificaciones y características, el dispositivo ConnectPortX8 objeto de este manual es el que más prestaciones ofrece.

El procesador que utiliza el Gateway es el NS9750 basado en el microprocesador ARM 926EJ-S que es el más potente en su tipo; el procesador NET+ARM de 32 bits funciona a una velocidad auto-variable de hasta 200 MHz, con tecnología CMOS de 0.13 µm, conexión Ethernet 10/100Base T bidireccional potencia de procesamiento de banda ancha, el control de gran variedad de periféricos comúnmente utilizados como USB, RS232, PCI, I2C y 1-WIRE.

De las propiedades generales de red que tiene el dispositivo, los protocolos de red utilizados para la transferencia de datos y la configuración del equipo son los TCP/UDP y el DHCP respectivamente. El Gateway puede servir de enrutador de la red a la que se encuentra conectado, realizando labores de configuración de redes privadas, control de listas de acceso, y la asignación de puertos IP a módulos ZigBee en la red del Gateway.

2.2. EL LENGUAJE PYTHON EN EL GATEWAY INDUSTRIAL CONNECTPORTX8

Python es un lenguaje interpretado de alto nivel con una sintaxis sencilla que permite programar aplicaciones complejas sin mucho esfuerzo. Entre sus principales características se encuentran su código limpio y entendible donde reglas que en otros lenguajes son opcionales aquí son obligatorias como la identación de bloques de código que permite un mejor entendimiento del programa, el manejo de excepciones posibilitando la recuperación de errores en tiempo de ejecución, la habilidad de poder ser implementado en diferentes

plataformas (MAC, Windows, Linux entre otros), el aprovechamiento de la programación orientada a objetos (OOP), los diferentes módulos disponibles que aumentan las prestaciones del lenguaje teniendo como ejemplos: interfaces gráficas, soporte para conexiones con base de datos, computación numérica, módulos especializados de ingeniería, páginas web, juegos, entre otros. La integración con otros tipos de lenguaje que extienden las posibilidades al permitir la utilización de sus librerías (Java, .NET y otros) hacen de Python un lenguaje versátil y adecuado para un creciente número de aplicaciones ingenieriles. Es tal la capacidad de este lenguaje, que empresas tan importantes como NASA, Honeywell, Rackspace, AstraZeneca han escrito e implementado aplicaciones en sus procesos y dan soporte de configuración y pruebas de algunos de sus dispositivos utilizando éste lenguaje.

Entre el grupo de empresas que incluyen programación en Python entre sus productos está Digi International fabricante del Gateway Industrial Connect Port X8, ya que este dispositivo posee un intérprete Python y algunas librerías cargadas en memoria. La versatilidad en el uso de estas herramientas junto con la posibilidad de crear programas y cargarlos al Gateway, permite infinidad de variables en el control de la red Zigbee y la conexión por internet para la comunicación de información en ambos sentidos. Es esencial para las posibilidades del TESTBED la administración del a información de los sensores y el control en el flujo de información para su almacenamiento.

2.3. EL PROTOCOLO DE COMUNICACIÓN ZIGBEE 802.15.4

El 802.15.4 es el estándar que engloba las características técnicas relacionadas con la arquitectura Zigbee. Esta tecnología presenta varias ventajas respecto a otras similares (Bluetooth, GPRS, WiFi) entre las que destacan: bajo consumo de potencia, bajo costo y la posibilidad de manejar una gran cantidad de nodos en

comparación con otras implementaciones inalámbricas como Bluetooth. Estas ventajas hacen a Zigbee interesante ya que su principal campo de acción es el control y la monitorización de procesos. Se debe resaltar que si bien la velocidad de transferencia es baja (máximo 250 Kbps), es suficiente para el tráfico de información básica en largas distancias. La capacidad de manejar muchos nodos (más de 65000) y su posibilidad de trabajar con alimentación a partir de baterías hacen atractiva esta tecnología para operar en ambientes donde se necesita llevar un seguimiento de grupos de variables (humedad, temperatura, consumo de energía, presión, flujo entre otros) frente a otras opciones de conectividad similares.

Con la versatilidad en el uso de varios tipos de topologías y una configuración sencilla de los dispositivos en las redes de sensores, Zigbee esta tomando gran auge en la implementación de redes de monitorización en diferentes ambientes. Algunos ejemplos son el desarrollo de los recientes sistemas de AMR de monitorización de variables (como consumo de energía, gas ó agua), sistemas de control de procesos de carga, monitorización de pacientes en clínicas y sistemas de seguridad. En aplicaciones a del hogar con todo lo que encierra la domótica (como control de luces, temperatura ambiente, acceso y demás).

3. DESCRIPCIÓN DE ELEMENTOS EN LA CONFIGURACIÓN DEL TESTBED

Realizando una descripción general del montaje que se muestra en la figura 1 se destacan cuatro dispositivos específicos, el computador servidor, la base de datos, el router inalámbrico y los Gateway coordinando cada uno una red de sensores y una red de módulos Xbee. También se tiene la posibilidad de administrar localmente los dispositivos por medio de un computador conectado al router vía Wi-Fi.

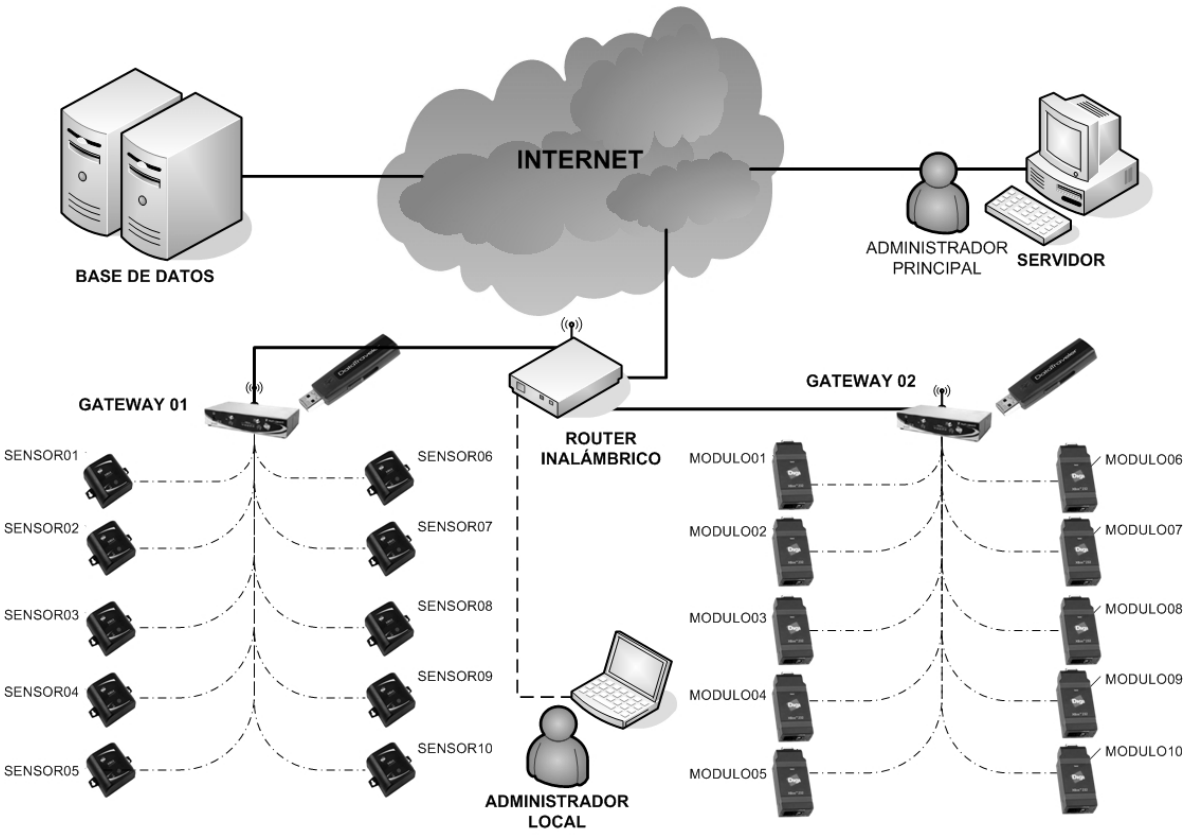

Figura 1 Esquema general del TESTBED

Dentro de la caracterización de las posibilidades que tiene el Gateway, se generan varios programas Python que ejecutados en el Gateway permiten administrar una red de módulos Xbee, que usados dentro de determinados circuitos permiten operar dentro de módulos de sensores o transmisores de datos que pueden ser almacenados en una memoria USB conectada al Gateway o enviada a un equipo remoto.

Desde la página de administración del Gateway es posible cargar los programas en Python al dispositivo para ejecutarlos mediante administración remota SSH. Una de las posibilidades del TESTBED es la comunicación con un servidor remoto transfiriendo la información mediante sockets TCP y desde allí se manipulan para ser almacenados en una base de datos.

3.1. ALMACENAMIENTO DE DATOS EN MEMORIA USB CONECTADA AL GATEWAY

El Gateway tiene la posibilidad de transferir información por medio de sus puertos USB, una forma es almacenar los datos que son recibidos de la red en una memoria USB para posteriores análisis de información. Por tanto, verificar el funcionamiento y revisar que se tiene acceso es parte fundamental del sistema, para ello se realiza el montaje.

Los dispositivos necesarios son el Gateway, una memoria USB Kingston (dentro del conjunto de dispositivos USB compatibles con el Gateway), y un computador de escritorio con el que se controla la transferencia de información. La configuración utilizada se muestra a continuación en la figura 2.

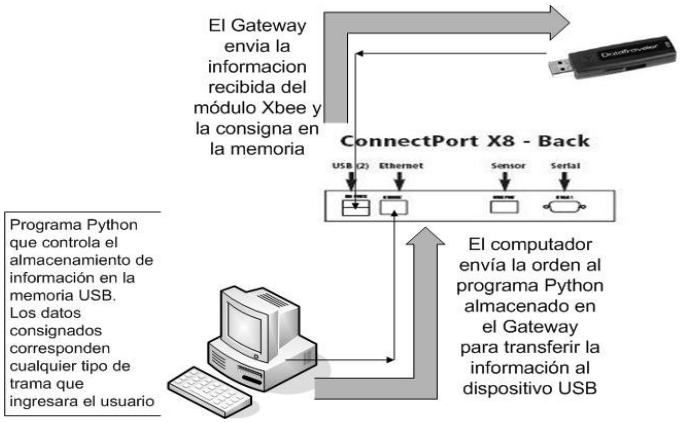

Figura 2 Montaje prueba transferencia de datos

Se ejecuta un programa Python con nombre usbp.py utilizando como medio de visualización el computador que para la prueba se llamara el servidor. El programa realiza en el servidor peticiones al usuario de ingresar información que desea sea guardada en la memoria USB; una vez consignada la información el programa vuelve a realizar la misma petición hasta el instante en que se ingrese la instrucción *quit* con la que finaliza el proceso y se procede a leer los datos guardados.

3.2. ALMACENAMIENTO DE DATOS EN UNA BASE DE DATOS

Para almacenar valores desde Python a una base de datos remota es necesario incluir una librería específica que permita realizar acciones en la base de datos MySQL. La librería MySQLdb establece una conexión SQL desde Python, esta librería permite la creación de una conexión estable con la base de datos remota, además de un cursor o apuntador que ubica el lugar dentro de la base de datos en el que se quiere realizar la inserción de datos o la modificación de alguno de estos. El proceso de conexión, las peticiones y las respuestas son mostrados en la figura 3.

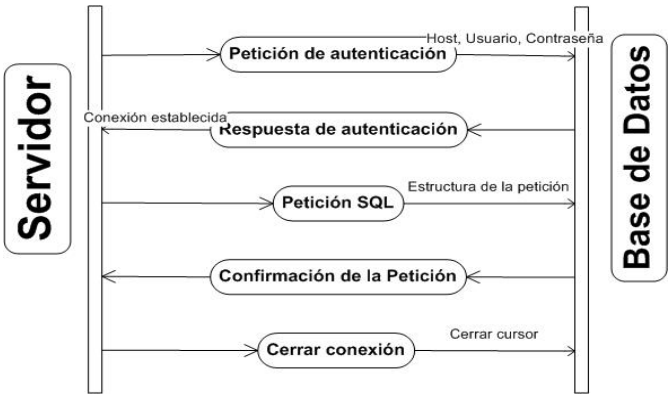

Figura 3 Protocolo de comunicación de la librería Python MySQLdb

En principio se realiza la conexión con la base de datos usando la clase *connect* donde es necesario señalar la dirección IP del servidor, el nombre de la base de datos y los datos de acceso como usuario y contraseña para la respectiva autenticación de la conexión. En la figura 3 se muestra la comunicación entre el servidor y la base de datos y las instancias necesarias para la administración de los datos. En el momento que se logra la conexión es necesario contar con un puntero para la ubicación en la base de datos de las diferentes tablas que eventualmente se tengan, este puntero es denominado cursor, que se obtiene con la ejecución de una clase de la librería con el mismo nombre *cursor*.

Para realizar una petición SQL dentro de Python se utiliza un módulo de la clase cursor, llamado *execute* en donde se establece la petición. En la figura 3 se muestra el momento en el que se realiza la petición. En el momento en que la petición es enviada; es necesaria la confirmación de la ejecución, y se realiza con la función *commit* de la clase *cursor*. El cursor puede cerrarse mediante el método *close* al igual que la conexión creada que se muestra en la figura 3 en la última instancia de comunicación.

3.3. PROTOCOLO DE CONSULTAS

Se establece como la mejor opción para el envío y recepción de información realizar solicitudes o pedidos de datos a los módulos Xbee de forma tal que se envíe la información de los sensores que responden a los comandos enviados por el Gateway, se utiliza además un computador (servidor) donde la información recibida por el Gateway se envía y se deja consignada en un archivo de texto. Esta es la mejor opción ya que permite tener control de la comunicación desde el servidor en cada instante, pues no se recibe información de manera continua ya que pueden ocasionar problemas de tráfico de datos.

Este protocolo de petición se implementa dentro del Gateway cuando se ejecuta el programa servidorgat.py que contiene las reglas necesarias para la comunicación. El programa realiza una encuesta a cada módulo en la red, captura la información y la envía al servidor para almacenarla, como no puede encuestar dos módulos en el mismo instante la información llega secuencialmente y es tarea del servidor identificar la procedencia de los datos. La figura 4 muestra el detalle del montaje utilizado y la forma como trabaja el protocolo.

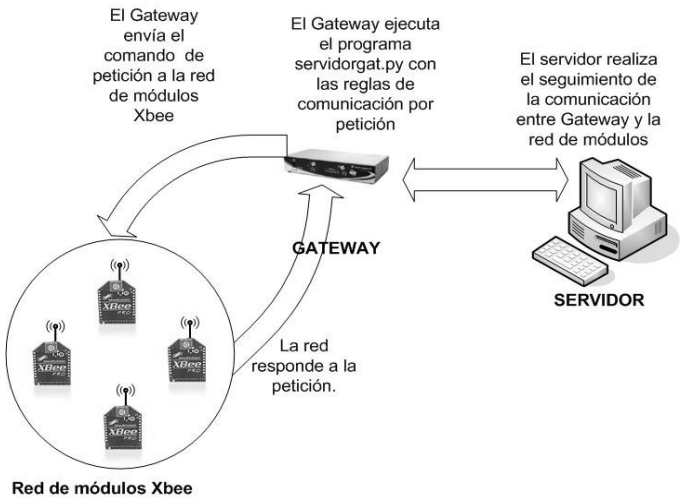

Figura 4 Protocolo de comunicación entre el Gateway y la red de módulos Xbee

Al realizar pruebas de desempeño del protocolo se consigue capturar las tramas que son enviadas por la red de módulos Xbee. En estas pruebas no se obtuvo perdida de información pero si se observó dos detalles importantes: el primero es que se genera un error cuando se llega a transmitir tramas de datos con tamaño superior a 72 Bytes, el segundo es un problema de traslape de las tramas recibidas en el momento de ser almacenadas en el servidor. Debido a que es tarea del servidor identificar la procedencia de los datos, existe la posibilidad de que el envío de información desde la red de los módulos sea más rápido que el tiempo que invierte el servidor en catalogar los datos, por tanto, se debe tener un control desde el Gateway del momento en el cual enviar las tramas al servidor. Se concluye entonces, que debe existir sincronismo ente el Gateway y el servidor, esto se logra con un retardo de al menos 0.3 segundos entre cada envío de información para asegurar que los datos que se almacenan no tengan problemas de traslape y que el servidor no bloquee el sistema.

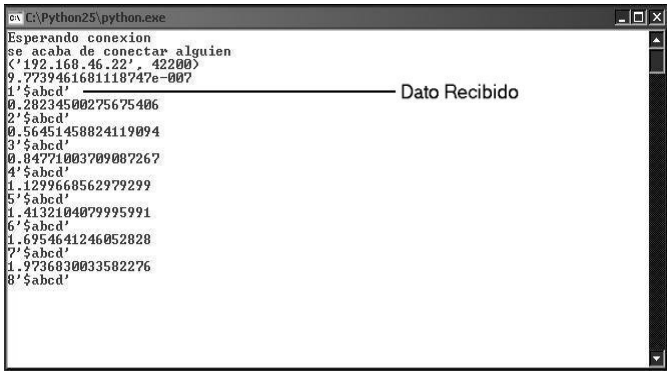

Figura 5 Detalle recepción de datos en el servidor

En la figura 5 se muestra una consola de comandos (bajo este sistema se ejecuta Python) con el fin de mostrar que efectivamente se está trabajando sobre el programa explicado anteriormente. La información recibida no son de los sensores de temperatura, luz y humedad, son datos de una tarjeta generadora de información, debido a que los módulos sensores están configurados de fábrica y tienen una función específica, los módulos sensores entregan los datos en un

formato definido y dentro de las limitaciones que tienen con el fin de hacerlos totalmente funcionales y no permiten llegar al límite en el tráfico de información. Esto se utiliza para determinar capacidades y limitaciones de estos dispositivos, aprovechando las opciones posibles para los módulos Xbee.

3.4. MEDICIÓN DE SENSORES DE LUZ, HUMEDAD Y TEMPERATURA

Cuando se tiene una red de módulos XBee con acople de sensores de luz, humedad y temperatura, se requiere de una librería específica para acceder y obtener la información de los datos de las medidas de los parámetros en ese instante de tiempo, esta librería necesariamente debe encontrarse dentro de los paquetes cargados en el Gateway, permitiendo la utilización de sus variables para la obtención de los parámetros de medida. Las clases y funciones de la librería adecúan los valores de entrada de los sensores en medidas digitales, que pueden ser enviadas a través del módulo XBee al a través del socket dedicado de la red ZigBee al Gateway y desde allí al servidor por medio de otro socket TCP. En el servidor estos datos son enviados a una base de datos remota.

Las medidas de temperatura están dadas en grados centígrados, medidas de iluminación en lumix y las de humedad en el porcentaje de humedad no condensada en el ambiente. Cabe aclarar que los sensores de temperatura y humedad están dentro de la protección del dispositivo, el sensor de iluminación tiene una pequeña ventana de salida necesaria para su medición. Las especificaciones técnicas de estos sensores y sus niveles de sensibilidad son las siguientes:

Sensor de temperatura: de -18° C a +55° C con una exactitud de +/- 2° C.

Sensor de iluminación de ambiente: opera entre 360 a 970 nanómetros de longitud de onda, similar a la sensibilidad del ojo humano. La longitud de onda de mayor sensibilidad se encuentra en los 570 nanómetros.

Sensor de humedad: en el rango de 0% a 100% de RH, con una variación de +/- 5% RH si la medida RH está entre 0% y 59%, y un +/- 8% RH si la medida RH está entre 60% y 100%.

3.5. MEDICIÓN DE RSSI

Dentro de las librerías que tiene el Gateway existe una dedicada a la comunicación ZigBee con la que se pueden obtener distintos parámetros propios de los módulos XBee, en este caso se aprovecha la propiedad de la medición de niveles de potencia RSSI entre los módulos de la red y el Gateway, mostrando estas medidas en decibeles.

Con el programa desarrollado se logra automatizar estas mediciones de potencia para determinar el estado de cada uno de los módulos en la red ZigBee, esto con el propósito de determinar el estado de cada módulo respecto al Gateway. Existen dos formas de mostrar las medidas, una es en la misma consola de administración remota del Gateway, la segunda desde el servidor realizando una conexión por sockets y almacenando los datos que son registrados. En el primer caso es necesaria para determinar de forma rápida el RSSI de cada modulo al instante y realizar los ajustes necesarios en la red y optimizarla. El almacenamiento de estos datos en la consola del servidor es utilizado para observar el comportamiento de cada módulo en el transcurso de un tiempo determinado, pues la comunicación inalámbrica se ve afectada por circunstancias en el entorno que no pueden ser controladas como es el caso del clima y la hora del día.

4. PROTOTIPO DE ADQUISICION Y ALMACENAMIENTO DE DATOS

Con la información recolectada anteriormente en las pruebas de conectividad y con el diseño del protocolo de comunicación, suficiente para presentar un prototipo de adquisición de datos robusto y que puede ser aplicado en la monitorización de distintos procesos. Este prototipo soporta su funcionamiento en dos programas principales: El que se ejecuta en el Gateway (servidorgat.py) y otro que se ejecuta en el servidor (ServidorProyectodeGrado.py).

La implementación de dos programas en los dispositivos encargados de la administración y transferencia de información, se realiza con el fin de separar estas tareas. Por su parte, el Gateway se encarga de la comunicación con la red de módulos Xbee como administrador de la misma (incluyendo el manejo de las peticiones de información), y la transferencia de la información al servidor, limitándose ser las veces de cliente y responder si se realiza la solicitud de esta operación. Por otro lado, el computador servidor es quien se encarga de la administración de los datos recolectados por el Gateway en la red Zigbee, controla el almacenamiento de información en una base de datos. Esta configuración permite que el sistema trabaje de forma sincronizada, que el control se localice exclusivamente en una entidad como el Gateway y la administración de la información en el computador servidor. La figura 6 muestra la configuración de dispositivos en el prototipo de adquisición de datos que se propone.

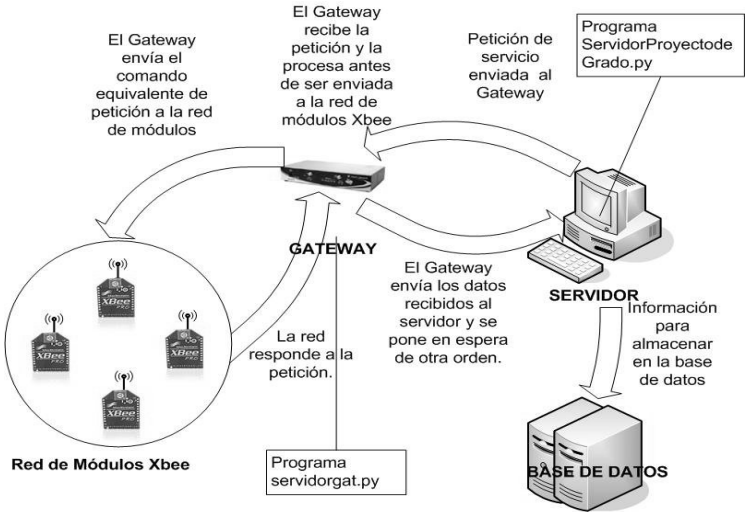

Figura 6 Prototipo de adquisición y almacenamiento de datos

Al realizar las pruebas de desempeño se recibe un conjunto de datos que son almacenados en una base de datos con el propósito de mantener organizada esta información. En el prototipo se tiene en cuenta los problemas del tamaño máximo de la trama que pueden transmitir los módulos Xbee y el tiempo entre transmisiones consecutivas, es decir, los errores por traslape de tramas o bloqueo del intérprete Python son controlados. Se muestra a continuación los detalles del proceso de que se lleva a cabo en el programa servidorgat.py de seguimiento de la comunicación y la información almacenada en la base de datos.

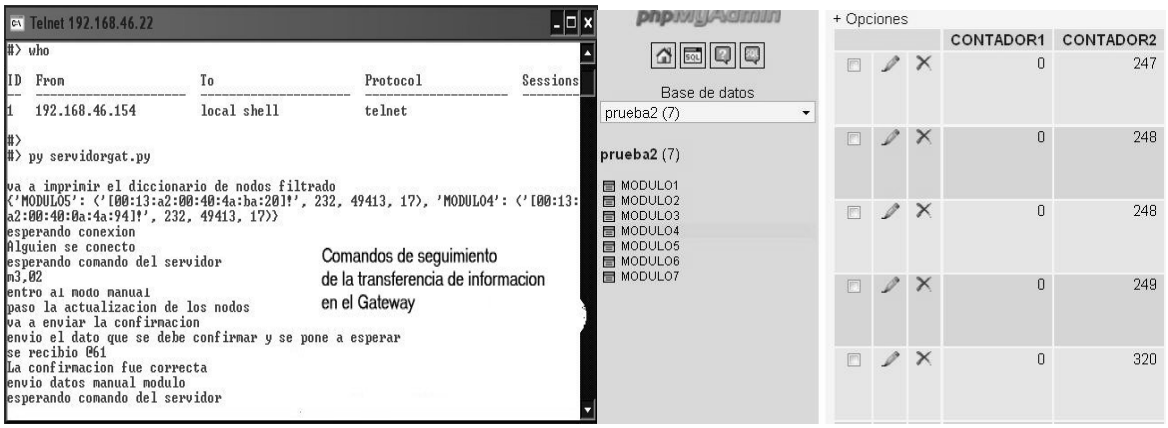

Figura 7 Desempeño del prototipo planteado

La figura 7 a la izquierda, se encuentra el seguimiento en el programa en el Gateway cuando se ha realizado una solicitud de información por parte del servidor, la figura 7 a la derecha muestra el detalle de la base de datos, en la columna marcada como “CONTADOR2” es donde existe la actualización del dato que se está almacenando. Para la prueba este el valor que se actualiza específicamente con el propósito de poder identificar fácilmente dicho cambio en el seguimiento de la consulta.

5. PROGRAMAS EN PYTHON RESULTADO DEL TESTBED

5.1. PROGRAMA DE MEDICIÓN DE SENSORES DE TEMPERATURA, LUZ Y HUMEDAD

La tarea del programa de medición de los sensores de temperatura, luz y humedad en esencia toma datos medidos por los sensores y transmitidos por el módulo al Gateway, almacenándolos en la base de datos. A continuación en la figura 7 se muestra la programación realizada para el procedimiento de adquisición de datos de los sensores.

En el primer conjunto de parámetros se realiza una llamada de las librerías necesarias para posterior ejecución de clases y funciones, el segundo punto es la búsqueda de los nodos asociados a la red con el método *getnodelist,* seguidamente se realiza un filtrado de los módulos coordinadores con el método *filter.* la tercera parte de líneas de comandos de la figura 7 es el almacenamiento de los nodos en un diccionario que por clave se tiene la etiqueta del nodo y el contenido es la dirección de cada módulo dentro de la red necesaria para realizar la conexión usando los el método *to_socket_addr;* seguidamente se crea el socket ZigBee utilizando la librería *socket* y las funciones *bind* con la que se establece la conexión, la función *settimeout* permite establecer el tiempo máximo de espera por conexión dado en segundos, esto con el fin de limitar el tiempo de petición de cada módulo.

Con el diccionario de nodos se utiliza cada uno de los campos del diccionario, esto se realiza en el cuarto conjunto de instrucciones que filtra la información necesaria para obtener únicamente la dirección MAC única para cada módulo XBee, esta dirección junto con la clase *XBeeLTHN* propio del Gateway crea un objeto con

propiedades de lectura de cada sensor, luego con el método *sample* genera nuevamente un diccionario esta vez con los parámetros de lectura.

Figura 8 Programa de medición local de sensores

En el quinto conjunto de instrucciones de la figura 8 se imprimen los datos en consola de las medidas de cada sensor, se hace referencia que cada parámetro en el diccionario es obtenido por medio de la llave entregando los datos de lectura dependiendo de la clave utilizada, sea para temperatura, luz o humedad. En la consola de administración remota del Gateway se muestran los resultados de temperatura, luz y humedad del módulo al momento de la ejecución del programa.

La sexta parte de instrucciones da cuenta del tratamiento de errores que pueden existir dentro de la comunicación de los nodos, si existe alguna demora en la recepción de datos o si el nodo encuestado ya no se encuentra dentro de la red de módulos se imprime una excepción y se cierra la conexión.

5.2. PROGRAMA DE MEDICIONES RSSI

Para realizar la consulta de la potencia de la señal de cada módulo con respecto al Gateway se genera el programa que se muestra en la figura 9 donde agrupan instrucciones por finalidad puntual. El primer paquete de parámetros se importa las librerías necesarias dentro del programa general, la segunda parte realiza un barrido de toda la red de módulos ZigBee con la función *getnodelist* de la clase zigbee, se filtran los módulos por tipo con *filter* de la misma forma del programa anterior. En la tercera parte se genera un diccionario que por clave tiene la etiqueta del módulo y por valor la dirección propia del módulo XBee con la propiedad *to_socket_addr,* seguidamente se crea un socket de conexión ZigBee y se establece la conexión de la misma forma del programa anterior.

```
import zigbee, struct
from socket import *
```
1

```
nodos = zigbee.getnodelist(refresh=True)
nodos = filter(lambda x: x.type != 'coordinator', nodos)
nodos_dict = dict()
```
2

```
for i in nodos:
    nodos_dict[i.label] = i.to_socket_addr(0xE8, 0xC105, 0x11)
print "este es el diccionario de nodos"
print nodos_dict
s = socket(AF_ZIGBEE, SOCK_DGRAM, ZBS_PROT_TRANSPORT)
s.bind(("", 0xE8, 0, 0))
s.settimeout(1)
try:
    for j in nodos_dict:
        aux = nodos_dict[j]
        dire = aux[0]
        print j
        rssi_raw = zigbee.ddo_get_param(dire, 'DB')
        print "Medicion de RSSI"
        rssi_val = struct.unpack('=B', rssi_raw)
        print "RSSi : " + str("%.2f" % rssi_val)
except:
    print "No hubo conexion con el nodo" + j
    s.close()
s.close()
```
3

4

5

Figura 9 Programa de medición RSSI desde el Gateway

La cuarta parte del programa mostrado en la figura 9 realiza un filtro de la información necesaria del módulo para realizar la encuesta RSSI que en este caso es la dirección MAC. La propiedad de la clase *zigbee* para obtener los parámetros

físicos del módulo es *ddo_get_param* que como parámetro lleva la dirección del módulo y la opción consultada que para el caso es el nivel de potencia, a continuación se utiliza la clase *struct* con la función *unpack* que interpreta el RSSI del módulo determinado por el parámetro *'=B'* y *rssi_raw*, seguidamente se convierte este valor para ser impreso en la consola de administración del Gateway.

Finalmente se ejecuta el mismo tratamiento de errores del programa anterior donde se esperan módulos inexistentes en la red o demora en la recepción de la información.

6. DISCUSION DEL TESTBED

Con el prototipo de adquisición de datos se cumplen con las expectativas para esperadas, pero existen puntos específicos que deben ser tratados, el primero de ellos es el traslape en la información que es enviada desde el Gateway hacia el servidor, debido a que el protocolo utilizado es la petición puede darse el caso -si no es bien sincronizado el prototipo- que el Gateway le solicite a la red de módulos Xbee información y la envíe a una velocidad mayor que a la que el servidor la consigna en la base de datos. Esta falta de sincronización es debido al tiempo que toma cada línea de código en ejecutarse, generalmente las consultas a la base de datos toman bastante tiempo (entre 2 y 5 segundos) mientras la transferencia desde la red de módulos Xbee dura solo algunos milisegundos. Por este motivo se implementó una rutina de retardo en el programa del Gateway con el fin de ralentizar el envío de los datos de la red hacia el servidor. En las pruebas de transferencia realizadas con el prototipo implementando este retardo es de 0.3 segundos logrando un buen sincronismo de la comunicación evitando los problemas expuestos.

Otro detalle es el tamaño de la trama que se puede transmitir desde la red de módulos Xbee. Al tratar de enviar tramas de un tamaño superior a 72 Bytes el intérprete Python genera una excepción y el prototipo falla. Una revisión en foros (*Digi Wiki for Developers*) sobre programación en Python permitió corroborar que efectivamente el intérprete genera un error con tramas de ese tamaño, debido a una limitación de fábrica al trabajar con los módulos Xbee Serie 2.

La utilización de los puertos USB es factible, pero en el momento que se intenta acceder a un dispositivo USB cuando se encuentran dos conectados no es posible, y se genera un bloqueo. En la revisión del programa Python se concluye

que el error no radica en el código si no que es una limitación de hardware o del intérprete Python del Gateway, por tanto la recomendación es conectar solo un dispositivo en el momento de tener el soporte de la información con la memoria USB.

7. CONCLUSIONES

Después de presentar el protocolo de comunicación y el prototipo de adquisición de datos del TESTBED, se concluye que a partir de la implementación de este sistema es posible almacenar información de la red de sensores en la base de datos y tener respaldo de la misma en una memoria USB conectada al Gateway, también, a partir del protocolo de petición escogido se mantiene un control de la comunicación entre la red de sensores, el Gateway y el servidor, donde las tramas de datos pueden tener cualquier valor y tamaño dentro de los límites de funcionamiento; entonces, se plantea la integración de este prototipo en algunas de las aplicaciones como las que fueron citadas. Se demuestra además que los dispositivos seleccionados permiten conectividad enfocada al diseño de aplicaciones inalámbricas, igualmente se plantean situaciones particulares en las que el sistema puede generar errores, por lo que se sugiere que estas situaciones deben tenerse en cuenta en las mejoras del sistema o en la integración en algún proceso especifico de forma que funcione correctamente.

8. REFERENCIAS

Digi International Inc. *Demystifying 802.15.4 and ZigBee,* (En línea), consultado 28 de Abril de 2009, http://www.digi.com/pdf/wp_zigbee.pdf, publicado 2007-2008

Digi Wiki for Developers, *Tcp to Ip port binding,* (En línea), consultado 28 de Abril de 2009, http://www.digi.com/wiki/developer/index.php/Tcp_to_Zigbee_port_binding,

E. Oliphant, Travis, *Python for Scientific Computing*, IEEE Computing Society and the AIP, 1521-9615, 2007

Guido Van Rossum, Fred L. Drake. HOWTO Release 2.6, Python Software Foundation. Octubre 2 de 2008.

Guido Van Rossum, Fred L. Drake. The Python Library Reference Release 2.6, Python Software Foundation. Octubre 2 de 2008.

John Goerzen. Foundations of python network programming. APRESS. 2004.

Magnus Lie Hetland. Beginning Python: From Novice to Professional. APRESS. 2005.

Santiago. P Claudia, *¿Bluetooth o Zigbee, Competencia o complementarios?*, (En línea), consultado 28 de Abril de 2009, www.acis.org.co/memorias/JornadasTelematica/IIJNT/BlueTooth_**Zigbee**.pdf

ZigBee Alliance, *ZigBee Specification,* (Archivo Zip descargable), consultado 28 de Abril de 2009, www.zigbee.org, publicado Enero 17 de 2008