ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES Y REDES

"*EVALUACIÓN DE LAS TOPOLOGÍAS FÍSICAS DE WSN MEDIANTE LA IMPLEMENTACIÓN DE UN PROTOTIPO DE MEDICIÓN DE VARIABLES AMBIENTALES PARA EL G.E.A.A-ESPOCH*"

TESIS DE GRADO

Previa a la obtención del Título de:

INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

Presentado por:

SAIGUA CARVAJAL SILVIA FERNANDA

VILLAFUERTE HARO MARÍA VERÓNICA

RIOBAMBA – ECUADOR

- 2013-

A Dios por la vida, la inteligencia y por guiarnos en el transcurso de la carrera.

A nuestros padres les agradecemos de manera especial por la paciencia, la motivación y sobre todo por el apoyo incondicional que siempre nos han brindado.

A nuestro tutor Ing. Alberto Arellano por la orientación, supervisión, y contribuciones con el presente trabajo, de igual manera al Ing. Jorge Yuquilema por las sugerencias e interés brindado.

A nuestros amigos por el ánimo, la ayuda y los buenos momentos compartidos, y a todas aquellas personas que contribuyeron para cumplir este objetivo.

Silvia y Verónica

Les dedico este trabajo a mis queridos padres Yilma y Octavio por ser ejemplo de vida y fuente de motivación, a la memoria de mi abuelita Dolores por sus valiosos consejos, a mi tía Luzmila por su apoyo y cariño, a mis hermanos Jenny, Byron y Henry por el ánimo y sobre todo por confiar en mí.

Silvia

Al cerrar una etapa más en mi vida, llena de esfuerzo, perseverancia, y mucho estudio quiero dedicar este trabajo a mis padres Víctor y Lupe ya que con su esfuerzo, apoyo, ánimos, mimos, amor, paciencia y comprensión, me dieron todo lo necesario y mucho más de lo que necesitaba, por lo que siempre serán el pilar fundamental para lograr esta meta y de todo lo que he podido conseguir en mi vida, todo es gracias a ustedes. Los amo papi y mami.

Quiero agradecerles a mis hermanos Cristhian, Víctor y David que con su alegría me motivan a nunca rendirme y continuar luchando, a mi tía Norma por siempre estar pendiente y poner un granito de arena en la realización de mis metas.

Gracias Andrés, por la paciencia, comprensión y apoyo en momentos muy difíciles ya que al querer sacrificar todo por verme feliz me daba ánimos para seguir adelante y poder cumplir con el con mi sueño.

¡GRACIAS!

María Verónica

FIRMAS RESPONSABLES Y NOTA

NOMBRE	FIRMA	FECHA
Ing. Iván Ménes **DECANO FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**	_____	_____
Ing. Wilson Baldeón **DIRECTOR DE ESCUELA INGENIERÍA ELECTRÓNICA EN TELECOMUNICACIONES Y REDES**	_____	_____
Ing. Alberto Arellano **DIRECTOR DE TESIS**	_____	_____
Ing. Jorge Yuquilema **MIEMBRO DEL TRIBUNAL**	_____	_____
Tlgo. Carlos Rodríguez **DIRECTOR CENTRO DE DOCUMENTACIÓN**	_____	_____

NOTA DE TESIS ESCRITA: _____

RESPONSABILIDAD DEL AUTOR

Nosotras, María Verónica Villafuerte Haro y Silvia Fernanda Saigua Carvajal, somos las responsables de las ideas, doctrinas y resultados expuestos en esta Tesis y el patrimonio intelectual de la misma pertenecen a la Escuela Superior Politécnica de Chimborazo.

María Verónica Villafuerte Haro

Silvia Fernanda Saigua Carvajal

ÍNDICE DE ABREVIATURAS

ACRÓNIMO	DESCRIPCIÓN
A/D	Analógico/Digital
AODV	Ad Hoc On-Demand Vector Routing
API	Aplicación de Interfaz de programación
DB	Date Base (Base de Datos)
DC	Corriente Directa
DCA	Diseño Completamente Aleatorio
DESITEL	Departamento de Sistemas y Telemática
DTH11	Digital Humidity Temperature Sensor
E/S	Entradas/Salidas
EPEC	Escuela de Posgrado y Educación Continua
GEAA	Grupo de Energías Alternativas y Ambiente
GND	Ground (Conexión a tierra)
GPS	Global Positioning System (Sistema de Posicionamiento Global)
IEEE	Institute of Electrical and Electronics Engineers
MAC	Control de acceso al medio
MANET	Mobile Ad-Hoc Network
MIT	Massachusetts Institute of Technology
NAM	Network Animator
NS2	Network Simulator 2
OSI	Open System Interconnection(Sistema de interconexión abierta)

PAN	Personal Area Network (Red de Área Personal)
PWM	pulse-width modulation (Modulación por ancho de pulso)
RF	Radio Frecuencia
RXD ó Rx	Recepción de datos
TCP/IP	Transmission Control Protocol/Internet Protocol
TICs	Tecnologías de la información y la comunicación
TXD ó Tx	Transmisión de datos
UART	Receptor/ Transmisor asíncrono universal
UDP	User Datagram Protocol (Protocolo de datagramas del Usuario)
USB	Universal Serial Bus
WBAN	Wireless Body Area Network (Red de área corporal)
WiFi	Wireless Fidelity (Fidelidad Inalámbrica)
WLAN	Wireless Local Area Network(Red Inalámbrica de Área Local)
WPAN	Wireless Personal Area Network(Red de área Personal)
WSN	Wireless Sensor Network (Red inalámbrica de sensores)
WWAN	Wireless Wide Area Network (Red de área Extensa)

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Hace poco tiempo, los servicios a los que se podía acceder de manera inalámbrica estaban limitados al servicio básico de comunicación por voz, sin embargo hoy en día gracias a la evolución rápida que han experimentado las redes inalámbricas se puede dar fe de grandes aportes en la educación, la salud, automatización, control remoto, etc. aportando de manera sustancial al mejoramiento de la calidad de vida de los usuarios.

El termino WSN (Wireless Sensor Network) es relativamente nuevo, estas redes pueden estar formadas por decenas, cientos o incluso miles de sensores que intercambian información entre sí sin necesidad de cables y con un bajo consumo de energía, generalmente están ubicadas en lugares remotos sin condiciones para tener una red cableada como por ejemplo en bosques, en ríos, en cultivos, etc, aunque también son muy útiles en ambientes indoor como en domótica, medicina, entretenimiento, etc.

No se espera que las redes inalámbricas lleguen a remplazar a las redes cableadas ya que estas últimas tienen mayor velocidad de transmisión/recepción y son más seguras, sin embargo las múltiples ventajas e infinidad de aplicaciones que muestran las redes WSN las han convertido en un objetivo clave de investigación alrededor del mundo.

En este trabajo se realizó un estudio comparativo entre las topologías físicas que soportan las WSN con la finalidad de determinar cuál es la más eficiente aplicada a una red inalámbrica de sensores ambientales.

El estudio se realizó mediante el uso del software NS2 que permite crear un ambiente similar al real y simular su funcionamiento, los resultados obtenidos de las pruebas de envío de paquetes, de consumo de energía y de cobertura, nos permite analizar las debilidades y fortalezas de cada topología y finalmente determinar cuál es la más óptima.

Por último para comprobar los resultados de la simulación se diseñó e implementó un prototipo empleando la topología estipulada, para crear la red se utilizó micro-controladores Arduino, módulos de comunicación XBee y para mostrar los datos al usuario se diseñó una aplicación que permite monitorear el comportamiento de la red en tiempo real.

CAPÍTULO I

MARCO REFERENCIAL

1.1. INTRODUCCIÓN

En el presente capitulo se expone la situación actual de las redes de sensores inalámbricas en el mundo, así como su funcionamiento general, sus principales ventajas y sus diversas aplicaciones. Además se propone nuestro trabajo investigativo como solución a la necesidad que tiene el Grupo de Energías Alternativas y Ambiente (GEAA) de transmitir los datos recogidos en la estación

meteorológica de la ESPOCH hacia un servidor que procese e interprete los mismos.

Se detallan también los objetivos, métodos y técnicas que nos ayudarán a la investigación.

1.2. ANTECEDENTES.

La evolución constante en los últimos años de las tecnologías inalámbricas han permitido facilitar las comunicaciones y así obtener datos de variables muy importantes, así el MIT identificó en Febrero de 2003 las 10 tecnologías emergentes que cambiarán el mundo entre las cuales esta WSN-Wireless Sensor Networks [1] .

Por lo que a partir de este acontecimiento el desarrollo a lo largo del mundo están implementadas millones de redes de este tipo ya que a ser usadas inicialmente en aplicaciones militares, ahora se han extendido a muchas áreas industriales y de interés social, como controles del proceso de producción, monitorización de la salud, automatización de la casa o control de tráfico. Otro proyecto gracias al esfuerzo en conjunto entre las Universidades de Harvard, New Hampshire, y la Universidad de Carolina del Norte han conducido recientemente a la implementación de una WSN para monitorear erupciones en el Volcán Tungurahua, un volcán activo en el centro de Ecuador [2].

La historia de los sensores inteligentes (o Smart Sensors) ha sido un tema apasionante, desde el primer y muy sencillo sensor desarrollado por Honeywell

en 1969.Los sensores de primera generación no tenían electrónica asociada, eran exclusivamente mecánicos, poco a poco con los desarrollos tecnológicos los sensores fueron evolucionando pasando de ser puramente analógicos, luego incluyendo en su arquitectura un Chip, hasta la actualidad que se encuentra dispositivos complejos de alta tecnología. [3]

Esta tecnología integra múltiples funciones automáticas: identificación, calibración, comprobación, transmisión, procesamiento digital, convirtiéndose en dispositivos inteligentes que permiten tener sensores que no solo entregan una señal digital sino además, linealizada, calibrada, robusta y compatible con otros dispositivos. El desarrollo de estos revolucionarios componentes ha permitido aumentar la eficiencia, calidad y velocidad de los procesos industriales, la investigación y el desarrollo científico.

Hoy en día, las Redes de Sensores Inalámbricas (WSN) se encuentran en auge debido a que son una tecnología emergente muy prometedora para una amplia variedad de aplicaciones en ingeniería debido a su fácil instalación y mantenimiento. Las redes de sensores inalámbricas están actualmente en el punto de mira de muchos investigadores y empresas tecnológicas. Su principal objetivo es la adquisición y el tratamiento de datos de forma rápida, flexible y autónoma. Los dispositivos que se utilizan (nodos) se comunican gracias a la tecnología ZigBee. [4]

La expansión de la tecnología ZigBee obedece al creciente, amplio y extremadamente flexible uso, de banda libre, frecuencias, bajo coste, capacidad de funcionamiento en tiempo real y principalmente bajo consumo de

potencia lo cual impone uno de sus principales criterios de diseño: deben gastar la menor cantidad de energía para asegurar que sus baterías perduren el máximo tiempo posible.

En lo que se refiere al hardware con el que funcionan las WSN se puede mencionar entre las más importantes a la plataforma electrónica Open Hardware Arduino [5] especializada en la creación de prototipos basada en software y hardware flexibles y fáciles de usar.

Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y luego procesar los datos, interactuando con diferentes dispositivos de salida, controlando luces, motores y otros actuadores según las condiciones requeridas que han sido programadas. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino basado en Wiring y el entorno de desarrollo basado en Processing. Los proyectos implementados con esta tecnología pueden ejecutarse sin necesidad de conectar los dispositivos a un computador, aunque bien tienen la posibilidad de hacerlo y comunicarse con diferentes tipos de software como Flash, Processing, MaxMSP.

WSN como tecnología ha sido adoptada he implementada en distintas disciplinas y diversos proyectos los mismos que contribuyen principalmente con la sociedad, por ejemplo en el campo de la meteorología en nivel científico una red de sensores con las características de WSN permite la fácil obtención, recolección y transmisión de datos de distinta naturaleza, los mismos que al ser procesados y evaluados permitirán describir la situación actual y desarrollar

modelos matemáticos que nos ayude a pronosticar futuros cambios en el sistemas, información importante para ciertos sectores como agricultura, también se podría pensar en sistemas de alertas tempranas activados por algún fenómeno ambiental que está siendo monitoreado y que permitirá precautelar la integridad física de cierta población.

1.3. JUSTIFICACIÓN DEL PROYECTO DE TESIS

El Instituto Nacional de Meteorología e Hidrología es el encargado de suministrar información vital sobre el tiempo, el clima y los recursos hídricos del pasado, presente y futuro, que necesita conocer el país para la protección de la vida humana y los bienes materiales, por lo cual está invirtiendo una gran cantidad de recursos para el análisis de los fenómenos medio ambientales en Ecuador [6]

La mayor parte de estaciones meteorológicas en nuestro País no poseen un sistema de comunicación en tiempo real para la obtención de los datos y por ende el análisis de los datos recogidos siempre dependerá de un factor humano que traslade la información hacia los centros de estudio.

Es evidente que las oportunidades que brindan las TICs, en el caso de monitorización y prevención temprana de fenómenos medio ambientales mediante el uso de WSN, resultaría beneficioso para la sociedad y las autoridades ecuatorianas al reducir el costo de los equipos, potencializar la monitorización de eventos en tiempo real y principalmente precautelar la

seguridad de la población al contar con un sistema que disminuya el tiempo de propagación de la información.

El desempeño de los equipos en la teoría es muy diferente que cuando se los implementa físicamente, por lo tanto al tener una comparación, se podría determinar las variaciones que existen entre ellos y en qué afectarían al utilizarlos en ambientes reales.

Las WSN son, en definitiva, un subconjunto de la categoría superior formada por las Redes Móviles Ad-Hoc Inalámbricas (MANET, Mobile Ad-Hoc Network), capaces de proporcionar soluciones a las exigencias actuales relativas al establecimiento de redes autónomas que no presentan restricciones en cuanto a la necesidad de una infraestructura externa para su operación. Este grado de autonomía es conseguido mediante la capacidad de las MANET de auto configurarse y admitir actualizaciones dinámicas en tiempo y espacio.

La diferencia entre los sensores que se conocen y la nueva generación de redes de sensores inalámbricas es que estos últimos son inteligentes, es decir, capaces de poner en marcha una acción según la información que vayan acumulando, y no están limitados geográficamente por un cable fijo.

Previamente en la ESPOCH se ha realizado un trabajo de fin de carrera acerca del diseño e implementación de un sistema de monitorización y detección de anomalías en ambientes cerrados, basado en la tecnología WSN [7].

También se ha utilizado el software NS2 para realizar un estudio comparativo de los protocolos de ruteo en redes Ad-Hoc aplicado a redes móviles. [8]

DIAGRAMA DEL PROTOTIPO DE PRUEBAS PROPUESTO:

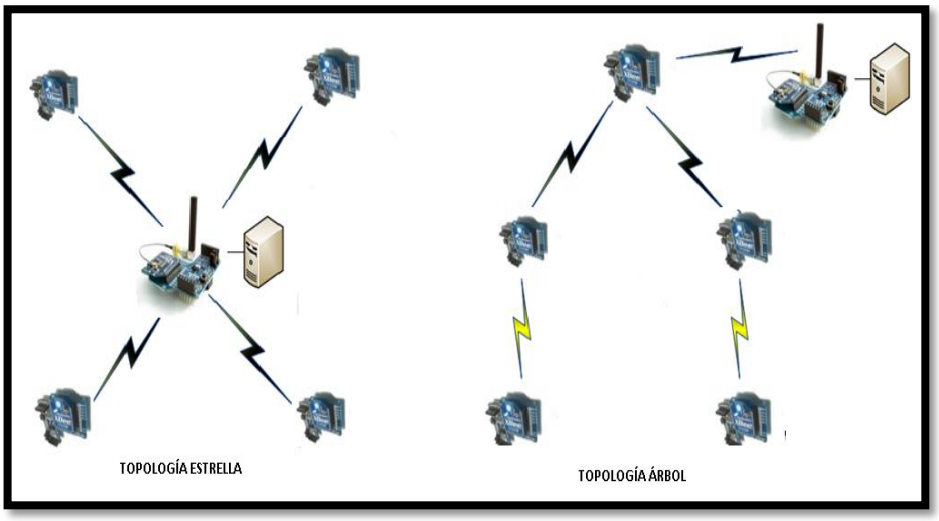

Figura I.1: Topologías propuestas de estudio

Fuente: Elaboración Propia.

Mientras que lo que nosotros se propone en este trabajo es realizar un estudio a profundidad del rendimiento de WSN, mediante el software ns2 se determinará la mejor topología física para este tipo de redes, para esto se toman en cuenta varios factores como: fiabilidad en envió de paquetes, cobertura, durabilidad, etc. Una vez determinada la mejor topología se implementará un prototipo de una red inalámbrica de sensores ambientales, y todos los datos recogidos serán enviados a una base de datos la cual los procesará e interpretará mediante graficas estadísticas.

1.4. OBJETIVOS

1.4.1 OBJETIVO GENERAL:

Evaluar el desempeño de las topologías físicas de una WSN mediante la implementación de un prototipo de medición de temperatura y humedad para el GEAA-ESPOCH.

1.4.2 OBJETIVOS ESPECÍFICOS:

- ✓ Revisar el estado del arte de las redes de sensores inalámbricos, enfocándonos en la plataforma libre Arduino y los módulos de comunicación XBee [9].
- ✓ Estudiar el estándar IEEE 802.15.4 y su importancia en WSN.
- ✓ Analizar la plataforma Open Hardware Arduino, sus especificaciones, estructura, lenguajes de programación y aplicaciones.
- ✓ Evaluar el desempeño y calidad de la Red WSN realizando pruebas de envío de paquetes, de consumo de energía y de cobertura mediante la utilización del simulador NS2.
- ✓ Diseñar e implementar un prototipo para el sistema de comunicación inalámbrico que permita el encaminamiento, recolección y manipulación de los datos recogidos.

1.5. HIPÓTESIS

El estudio del desempeño de una WSN mediante métricas de evaluación (pruebas de envío de paquetes, de consumo de energía y de cobertura) permitirá determinar la topología más adecuada para la implementación de una red de sensores inalámbricos eficiente.

1.6. MÉTODOS Y TÉCNICAS

1.4.3 MÉTODOS

Tipo de investigación científica descriptiva, no experimental, de Laboratorio y transversal, básica abordado en el siguiente protocolo de investigación:

Método a Seguir

1. Investigar todo lo relacionado con las redes de sensores inalámbricas (WSN) enfocándonos en las topologías estrella y árbol.
2. Mediante el software NS2 especializado en simular redes inalámbricas, definir la mejor topología teniendo en cuenta las métricas de desempeño de WSN.
3. Implementar el prototipo de medición de variables ambientales basado en el resultado de NS2.
4. Recoger los datos de cada nodo del prototipo, enviarlos hasta una base de datos e interpretarlos.

1.6.1. TÉCNICAS

Las técnicas a utilizar en la elaboración de este proyecto investigativo, son las siguientes:

- *Observación directa.-* es la inspección que se hace directamente a un fenómeno dentro del medio en que se presenta, a fin de contemplar todos los aspectos inherentes a su comportamiento y características dentro de ese campo.

- *Método Inductivo.*- es el estudio de las pruebas que permiten medir la probabilidad inductiva de los argumentos así como de las reglas para construir argumentos inductivos fuertes.
- *Método estadístico de diseño completamente azar.*- En este modelo, el experimentador asigna las unidades experimentales a los tratamientos al azar, con la única restricción del número de observaciones que se tomarán en cada tratamiento. Es el más sencillo y se origina por la asignación aleatoria de tratamientos a un conjunto de unidades experimentales.se usa para la determinación de las diferencias entre dos medias muéstrales y para la construcción del intervalo de confianza para la diferencia entre las medias de dos poblaciones cuando se desconoce la desviación típica de una población y ésta debe ser estimada a partir de los datos de una muestra.

CAPÍTULO II

MARCO TEÓRICO

2.1 INTRODUCCIÓN

La comunicación sin el uso de cables a través del espectro radioeléctrico ha sido un tema muy interesante desde su descubrimiento debido a los múltiples usos y beneficios que esto nos permite, hoy en día se puede encontrar redes inalámbricas no solo en áreas tecnológicas sino en casi todas lo que se puedan imaginar.

Las redes inalámbricas tienen entre sus principales ventajas la movilidad y el área de cobertura que puede variar entre unos pocos metros hasta alcanzar

inmensas distancias pudiendo realizar una comunicación entre continentes (WWAN), lo que les ha permitido un rápido desarrollo y que se encuentren en auge, es por eso que en el presente capítulo se investiga el mundo de las comunicaciones inalámbricas, específicamente en las redes de sensores inalámbricas (WSN) que hoy por hoy es una de las tecnologías con diversas aplicaciones y muy buenos resultados a bajos costos, se habla sobre su funcionamiento, arquitectura, elementos que las conforman como son placas base Arduino, módulos de comunicación XBee, y los diversos sensores que se puede utilizar, también se menciona sobre los diversos campos en los que actualmente se las utiliza.

2.2 WIRELESS SENSOR NETWOK (WSN)

Las WSN forman parte de las Redes Inalámbricas de Área Personal (WPAN), son un tipo de red inalámbrica ad hoc sin infraestructura física preestablecida ni administración central, formada por numerosos sensores los cuales están distribuidos espacialmente y recogen información de distintos fenómenos físicos y ambientales, con la finalidad de que estos sean procesados y se pueda obtener estadísticas y un control en tiempo real de los mismos.

Este tipo de redes están diseñadas para aplicaciones en las cuales no se cuente con una infraestructura cableada o para lugares donde sea inaccesible el suministro energético tal como se puede apreciar en la Figura II.2. Por esta razón es necesario que los nodos funcionen con la mínima cantidad de energía de sus propias fuentes como pilas, baterías o panel solar y que se comuniquen por medio de canales inalámbricos.

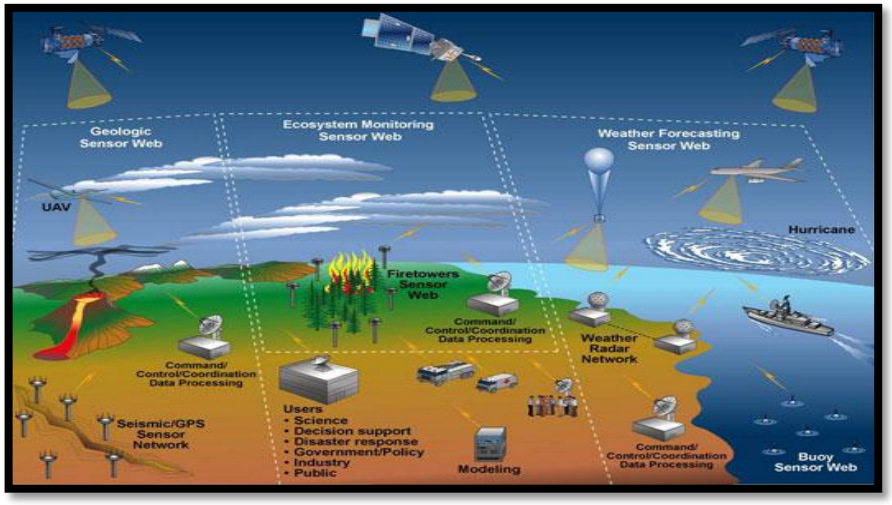

Figura II.2 : Uso de redes WSN

Fuente: http://mspsoft.ir/wp-content/uploads/SensorWebImageForEnewsJuly21.jpg

Con todo esto, una red de sensores puede ser descrita como un grupo de motas que se coordinan para llevar a cabo una aplicación específica. Al contrario que las redes tradicionales, las redes de sensores llevarán con más precisión sus tareas dependiendo de lo denso que sea el despliegue y lo coordinadas que estén.

2.2.1 Descripción de la WSN

Los dispositivos de una WSN, conocidos como nodos sensores o motas (en inglés mote) son unidades autónomas, capaces de realizar algún tipo de procesamiento, recopilación de la información sensorial, y la comunicación con otros nodos conectados en la red. Tienen la habilidad de medir un medio físico con gran detalle. Asimismo, constan de un microprocesador, fuente de energía (pilas o baterías), un radiotransceptor y un elemento sensor como se muestra en la Figura II.3.

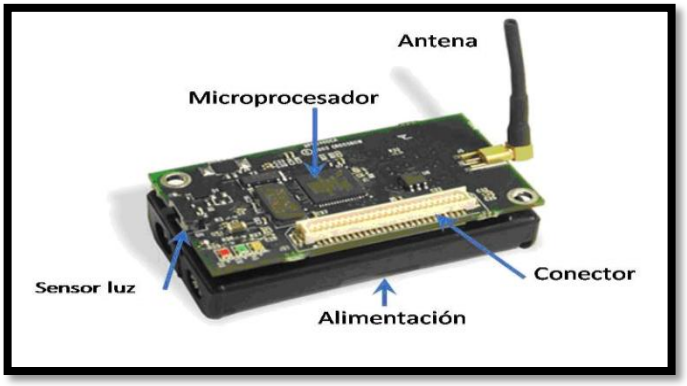

Figura II.3: Nodo

Fuente: http://lasredesconsensores.blogspot.com/2012_07_01_archive.html

Una WSN es una red de diminutos dispositivos, equipados con sensores, que colaboran en una tarea común y están distribuidos en un área geográfica determinada.

Las WSN tienen capacidad de auto-restauración, es decir, si se avería un nodo, la red encontrará nuevas vías para encaminar los paquetes de datos. De esta forma, la red sobrevivirá en su conjunto, aunque haya nodos individuales que pierdan potencia o se destruyan. Las capacidades de autodiagnóstico, autoconfiguración, auto-organización, auto-restauración y reparación, son propiedades que se han desarrollado para este tipo de redes para solventar problemas que no eran posibles con otras tecnologías. Las redes de sensores se caracterizan por ser redes desatendidas (sin intervención humana), habitualmente construidas ad-hoc para resolver un problema muy concreto (es decir, para ejecutar una única aplicación).

Es importante conocer los protocolos más usados en la actualidad para formar redes de sensores inalámbricas (WSN), en la mayor parte de aplicaciones se

utiliza el protocolo Zigbee debido a las múltiples ventajas que presenta. A diferencia de Wi-Fi que permite un máximo de 32 nodos y de Bluetooth que permite un máximo de 7 nodos, Zigbee permite la creación de redes de 65000 nodos y con un tráfico de hasta 250Kbps

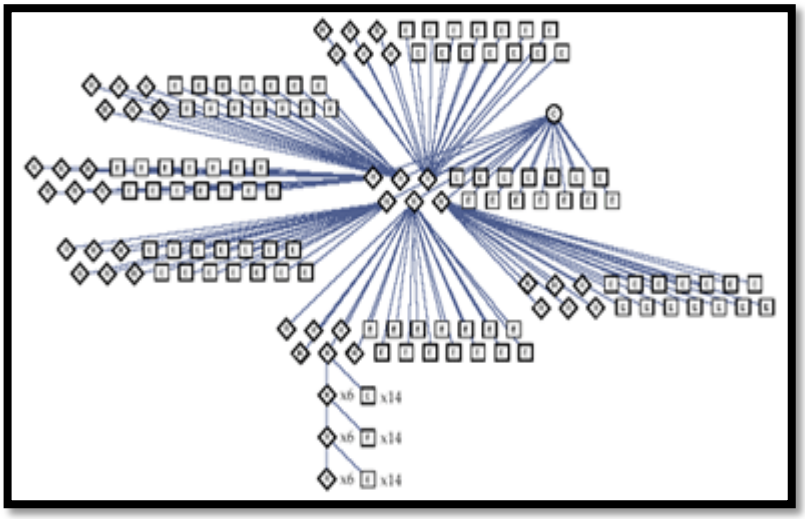

Figura II.4: Wireless Sensor Network con ZigBee

Fuente: http://www.blogelectronica.com/redes-zigbee-i-introduccion/

Las redes basadas en el protocolo Zigbee, en similitud con las que usan Bluetooth, son tecnologías inalámbricas de área personal que buscan un ahorro de energía sacrificando otros recursos del sistema como es el Ancho de Banda. Comparando los distintos grados de ahorro de energía entre estas dos tecnologías, Zigbee tendría un menor consumo energético. Esto se debe a que el Zigbee permanece la mayor parte del tiempo dormido, mientras que en una comunicación Bluetooth siempre se está transmitiendo o recibiendo [10].

2.2.2 Características

- **Topologías Dinámicas:** Un WSN debe adaptarse ante cambios en la Red. Los nodos son auto-configurables, con tolerancia a fallos y presentan una elevada fiabilidad.
- **No se utiliza una infraestructura de red:** Una red de sensores no tiene necesidad alguna de infraestructura para poder operar, ya que sus nodos pueden actuar de emisores, receptores o enrutadores.
- **Tolerancia a errores:** Un nodo dentro de la red debe ser capaz de mantener su funcionamiento a pesar de errores en el sistema.
- **Comunicaciones multisalto o broadcast:** Se utilizan tecnologías inalámbricas de corto alcance, es necesario que la red cuente con un protocolo que permita comunicaciones “multi-hop”.
- **Tamaño reducido:** Estos dispositivos están destinados a ser extremadamente pequeños, variando en tamaño desde una escala del micrómetro a un milímetro (que puede ser comparado con un grano de arena o incluso una partícula de polvo).
- **Bajo consumo energético:** Es uno de los factores más sensibles debido a que dependen de pilas o baterías y tienen una larga autonomía de funcionamiento. Pueden operar sin mantenimiento durante varios meses o años.
- **Limitaciones de Hardware:** Con el fin de ahorrar energía, el hardware debe ser lo más sencillo posible. Esto nos limita a tener una capacidad de procesamiento limitada.

- **Soporta múltiples opciones de conectividad:** Las redes WSN no limitan al uso de la tecnología a utilizar, esto depende del uso y de las condiciones que presente el lugar en donde se las va aplicar, la tecnología más utilizada es Zigbee sin embargo se puede utilizar Wi-Fi, Wimax, Bluetooth, etc.
- **Costo de producción:** Normalmente económicos en comparación a otras tecnologías. [11] [12]

2.2.3 Elementos De Una WSN

Las redes de sensores inalámbricos están compuestas por un conjunto de dispositivos denominados nodos sensores, con capacidad limitada ya sean de almacenamiento o de procesamiento de información y comunicación, cuyo tiempo de vida depende de la fuente de energía instalada en el dispositivo.

Estos dispositivos se encuentran dispersos de manera ad-hoc en una determinada área a monitorizar. En redes de comunicación, dicha expresión hace referencia a una red en la que no hay un nodo central, sino que todos los nodos están en igualdad de condiciones como ya se mencionó en otros capítulos.

Típicamente, el modelo seguido por las aplicaciones es el siguiente: realizar una serie de mediciones sobre el medio, transformar dicha información en digital en el propio nodo y transmitirla fuera de la red de sensores vía un elemento gateway a una estación base, donde la información pueda ser almacenada y tratada temporalmente para acabar finalmente en un servidor con mayor capacidad que permita componer un histórico o realizar análisis de

datos. En la Figura II.5 se puede observar la composición básica de una Wireless Sensor Network (WSN)

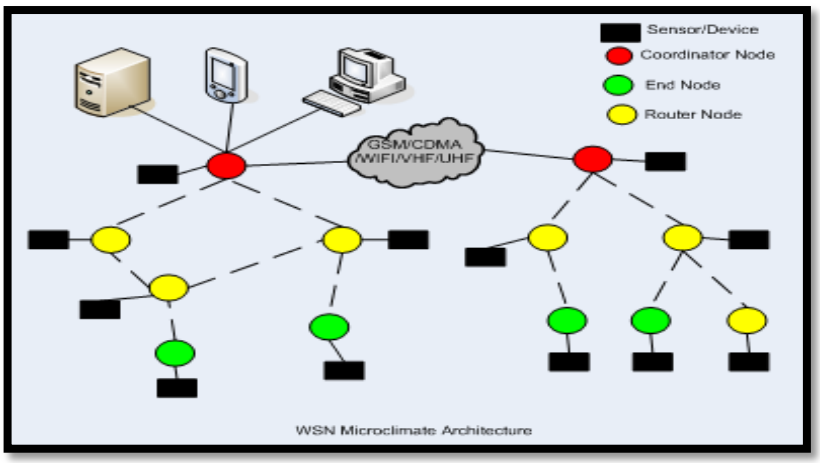

Figura II.5: Elementos de una WSN

Fuente: http://i97.photobucket.com/albums/l209/japmarpaung/architecture.png

A continuación se explicara cada elemento que conforma la estructura de una red WSN.

2.2.3.1 Nodos Inalámbricos

Los nodos inalámbricos se llaman motas, del inglés ‘mote’, por su ligereza y reducido tamaño. Son dispositivos electrónicos capaces de captar información proveniente del entorno en el que se encuentran, procesarla y transmitirla inalámbricamente hacia otro destinatario.

Diseñar un mota no se reduce a miniaturizar una computadora. Hay que tener en cuenta que se quiere un espacio reducido, un consumo muy bajo de energía y un coste de los dispositivos reducido. Y en contraposición a esto una potencia de ejecución de programas elevadas y una transmisión de datos eficaz y con

amplia longitud de emisión. Estas motas son diseñadas y programadas para formar parte de una red con un objetivo particular, lo que quiere decir que una mota aislada tiene muy poca utilidad.

El hardware de cada uno de estos dispositivos tiene varias partes bien diferenciadas: los sensores son de distinta naturaleza y tecnología. Toman del medio la información y la convierten en señales eléctricas. En el mercado existen placas con sensores de medida de muy diversos parámetros, como sensores de presión barométrica, GPS, luz, medida de radiación solar, humedad en suelo, humedad aire, temperatura, sonido, velocidad del viento y un largo etc.

2.2.3.2 Nodos Sensores

Se compone de una mota y una placa de sensores. Mota es la entidad compuesta de un procesador y los dispositivos de radio, para el procesamiento de procesamiento y de comunicación al nodo sensor. Los procesadores de radio, toman los datos del nodo sensor a través de sus puertas de datos, y envían la información a la estación base.

2.2.3.3 Gateway

Son los elementos destinados a la interconexión entre la red de sensores y una red de datos (por ejemplo TCP/IP). Es un nodo especial sin elemento sensor, cuyo objetivo es actuar como puente entre dos redes de diferente tipo. Los dispositivos que realizan la función de interconectar dos redes de diferente

naturaleza se les llama dispositivo puerta de enlace; pero el término más conocido en el ambiente de las redes es gateway. [13]

2.2.4 Estación Base

Recolector de datos basado en un computador o sistema embebido, aquí se procesa e interpreta la información.

2.2.5 Componentes De Un Nodo WSN

Para poder comprender mejor la estructura de un nodo WNS se debe tener presente los elementos que los componen, es decir, su estructura.

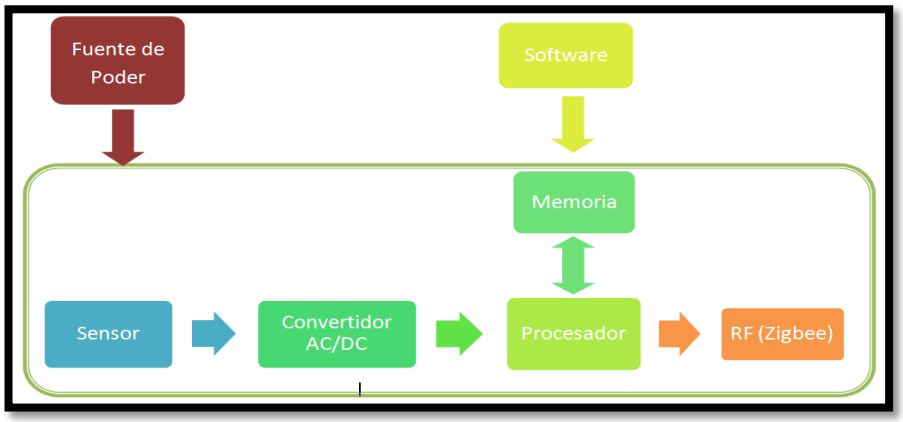

Figura II.6: Arquitectura de un nodo WSN

Fuente: Elaboración propia.

Como se muestra en la Figura II.6, un nodo de una WSN integra detección, transformación, procesamiento, almacenamiento y transmisión inalámbrica con la ayuda de una fuente de poder. A continuación se explica brevemente alguno de estos elementos.

2.2.5.1 Fuente de Poder

Una fuente de poder apropiada y eficiente debe ser capaz de proporcionar energía al nodo un largo periodo de tiempo, ya sean meses o años, dependiendo de la aplicación que vaya a tener nuestro nodo. Aunque es muy común que las fuentes de poder duren por lo menos varios meses, y mucho más si se combina con alguna fuente de recarga como paneles solares, esto brindará perdurabilidad de la baterías por años. En la Figura II.7 se puede observar que el mota está funcionando con 2 pilas AA.

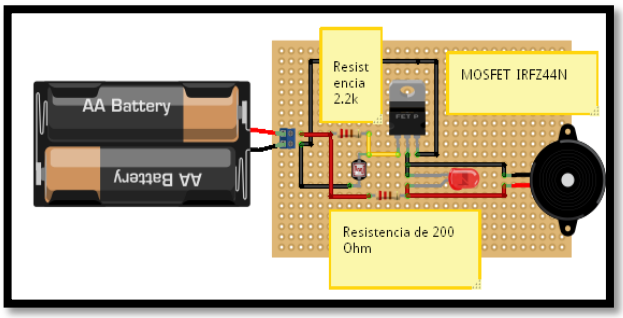

Figura II.7: Fuente de poder

Fuente: http://Arduinolab.files.wordpress.com/2012/05/laser-tripwire-fritzing-blog.png

2.2.5.2 Sensores

Un sensor es un dispositivo que está capacitado para detectar acciones o estímulos externos y responder en consecuencia. Estos aparatos pueden transformar las magnitudes físicas o químicas en magnitudes eléctricas [14].

Un tipo de clasificación muy básico, es diferenciar a los sensores atendiendo a la naturaleza de la señal eléctrica generada, así se tiene a los sensores de tipo PASIVOS o ACTIVOS; los sensores activos generan la señal de salida sin la

necesidad de una fuente de alimentación externa, mientras que los pasivos si requieren de esta alimentación para poder efectuar su función. [15]

En la Figura II.8 se muestra un sensor básico para placas Arduino.

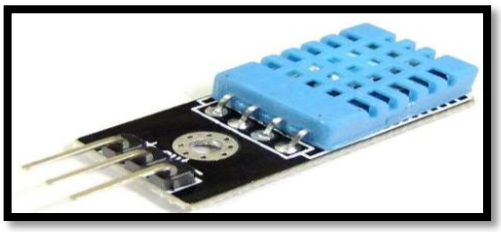

Figura II.8: Sensor de Temperatura y Humedad

Fuente: http://es.paperblog.com/sensor-de-temperatura-y-humedad-con-el-modulo-dht11-y-Arduino-1541392/

2.2.5.3 Procesador

Es aquel que maneja los datos luego de haber sido convertidos para manipularlos según las configuraciones que se hayan realizado, por ejemplo con el almacenamiento, para lo cual necesita una memoria, así como también para tareas como la corrección de errores, el cifrado.

Los requisitos de cómputo y almacenamiento en una WSN dependen de la aplicación y pueden ir desde la utilización de un microcontrolador de 8 bits hasta 64 bits. Los requerimientos de almacenamiento pueden igualmente oscilar entre 0,01 hasta 100 gigabytes (GB).

Además de la memoria proporcionada por el microcontrolador no es extraño encontrar modelos que incluyan memoria externa adicional, por ejemplo en forma de memoria flash [13]. En la Figura II.9 se observa un Arduino Uno que el cual cumplirá con las tareas del procesador en los motas.

Figura II.9 Microprocesador Arduino

Fuente: http://www.unocero.com/wp-content/uploads/2011/12/Arduinouno.jpg

2.2.5.4 Radio

El dispositivo de radio es aquel que brinda la posibilidad de comunicar y transmitir la información inalámbricamente entre todos los nodos que forman la WSN, con el propósito de lograr propiedades específicas de este tipo de comunicación tal como: bajo consumo de energía, velocidad de datos, y distancias cortas. Las WSN usan las frecuencias de comunicación que andan entre 433 MHz y 2.4 GHz.

Los estados de operación son emitir, recibir, dormir e inactividad. En los actuales modelos de transceptor, el modo inactivo consume casi igual que el modo recepción. Por lo que es mejor tener completamente apagado las comunicaciones radio, en el modo Inactivo, cuando no se está emitiendo ni recibiendo. También es significativa la cantidad de energía consumida cuando cambia de modo durmiente a transmisión de datos. En la Figura II.10 se visualiza un dispositivo de radio comunicación XBee, el cual es el más utilizado en este tipo de redes.

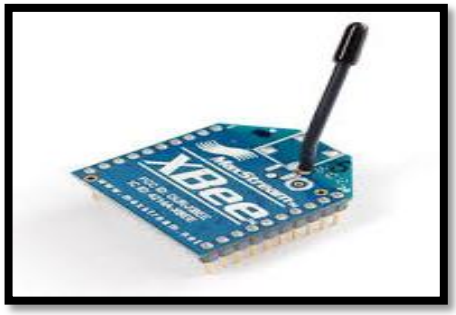

Figura II.10: Antena Wireless XBEE

Fuente: http://www.adafruit.com/images/medium/XBeemodule_MED.jpg

2.2.3 Topologías De WSN

Topología se refiere a la configuración de los componentes hardware y como los datos son transmitidos a través de esa configuración. Cada topología es apropiada bajo ciertas circunstancias y puede ser inapropiada en otras. La idea de una red de sensores surge gracias a las posibilidades que nos da la tecnología.

Existen 3 topologías recogidas por Zigbee: malla (MESH), árbol (TREE) y estrella (STAR).

2.2.3.1 Topología Estrella (Star-Monosalto)

El gateway de la red se sitúa en el centro de la red, es decir, es un sistema donde la información enviada sólo da un salto y donde todos los nodos sensores están en comunicación directa con la puerta de enlace, usualmente a una distancia de 30 a 100 metros. Los nodos finales no intercambian información entre ellos, sino que usan la puerta de enlace para ello, si es

necesario. La puerta de enlace también es usada para transmitir datos al exterior y permitir la monitorización de la red.

La topología en estrella como se puede ver en la Figura II.11 es la que menor gasto de energía desarrolla, pero por el contrario está limitada por la distancia de transmisión vía radio entre cada nodo y la puerta de enlace. Tampoco tiene un camino de comunicación alternativo en caso de que uno de los nodos tenga obstruido el camino de comunicación, lo que lleva a que en este caso la información de ese nodo sea perdida. Teóricamente Se pueden conectar hasta 65536 [16] nodos a la red de este tipo.

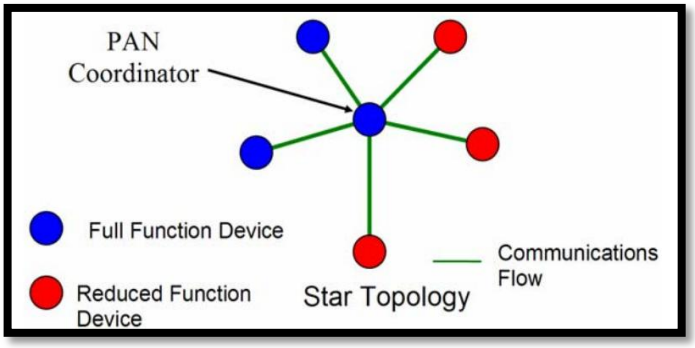

Figura II.11: Tipología Zigbee-Árbol

Fuente: http://clusterfie.epn.edu.ec/ibernal/html/CURSOS/Marzo07Agosto07/Actualizacion/ZigBeeNov06.pdf

2.2.3.2 Topología Malla (Mesh-Multisalto)

La topología en malla es un sistema multisalto, donde todos los nodos son a la vez enrutadores y son idénticos. Cada nodo es capaz de enviar y recibir información de otro nodo y de la puerta de enlace. A diferencia de la topología en estrella, donde los nodos solo pueden hablar con la puerta de enlace, en ésta los nodos pueden enviarse mensajes entre ellos.

La propagación de los datos a través de los nodos hacia la puerta de enlace hace posible, por lo menos en teoría, crear una red con una extensión posible ilimitada. Este tipo, también es altamente tolerante a fallos ya que cada nodo tiene diferentes caminos para comunicarse con la puerta de enlace como se observa en la Figura II.12. Si un nodo falla, la red se reconfigurará alrededor del nodo fallido automáticamente. [17]

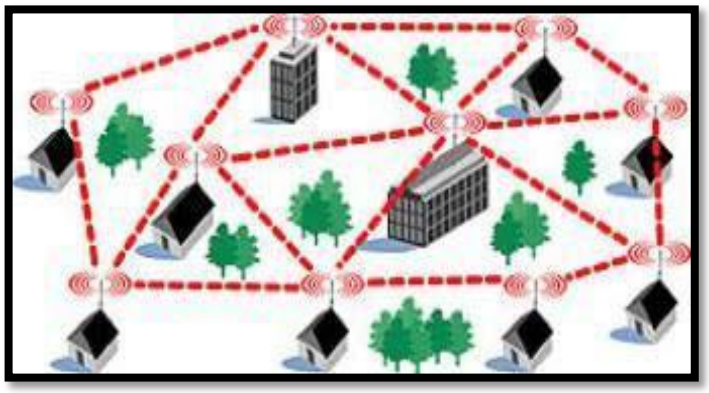

Figura II.12 Topología tipo Malla (Mesh)

Fuente: http://www.poderypolitica.com/portal/images/stories/conectividad/4.jpg

2.2.3.3 Topología Árbol (TREE)

El coordinador de la red se comunica solo con dispositivos enrutadores, los cuales a su vez comunican de forma directa con los nodos finales. Un ejemplo sería un edificio en el que se establece un clúster por planta, los dispositivos finales que haya instalados en dicha planta comunicarán únicamente con el nodo enrutador de dicha planta, que tendrá que encaminar los datos, poniendo en contacto así a dichos dispositivos con el coordinador, que será la raíz del árbol. Como muestra la Figura II.13.

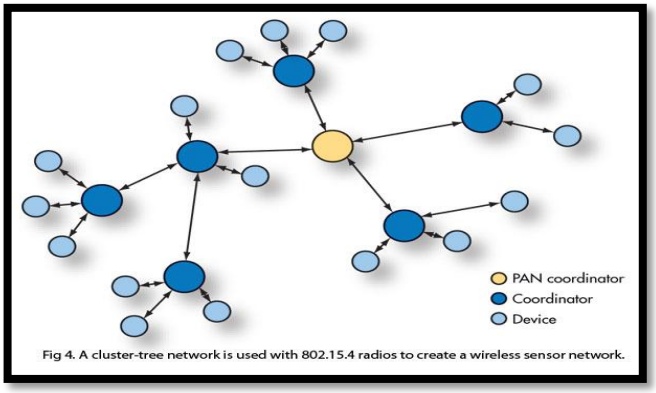

Figura II.13 Topología de tipo Árbol (TREE)

Fuente: http://electronicdesign.com/content/content/62518/62518-fig4.jpg

La Tabla II.I muestra en forma resumida las ventajas y desventajas de las topologías.

Tabla II.I: Ventajas y Desventajas de las topologías WSN

Topología	Ventajas	Inconvenientes
Estrella	· Baja latencia · Topología muy robusta. Fiabilidad muy alta. · Sencillez y rapidez en el desarrollo. · Fácil de desplegar: comportamiento en cliente similar al obtenido en laboratorio. · Gasto energético homogéneo.	· No siempre es posible desplegar una topología de este tipo. · Escalabilidad baja. Posibles problemas de colisiones cuando aumenta el número de ZEDs. · Si falla el nodo central cae toda la red.
Árbol	· Mantiene todas las ventajas de la topología Estrella. · Alta escalabilidad. · Menor porcentaje de colisiones.	· La inclusión de routers puede encarecer significativamente la solución final. · Puede caer una parte significativa de la red al caer un router. · Costoso y difícil de desarrollar el algoritmo de enrutado dinámico. · Baja fiabilidad de los algoritmos de enrutamiento.
Malla	· Menor coste: no es necesaria una cantidad tan grande de routers para alcanzar una gran escalabilidad.	· Complejidad del sistema alta. · Alta cantidad de colisiones. · Empeora la latencia de la red. · Significativas diferencias entre

	▪ Pueden caer uno o varios nodos que la información seguirá obteniéndose, al existir rutas alternativas.	las pruebas en laboratorio y el despliegue real, con lo que se obtienen despliegues muy costosos.

Fuente: http://www.javierlongares.com/arte-en-8-bits/introduccion-a-zigbee-y-las-redes-de-sensores-inalambricas/

2.2.3 Aplicación De Las Redes De Sensores Inalámbricos

Las aplicaciones que ofrecen las WSN, debido a su estructura sin ataduras, es decir sin cables que limiten su ubicación además de su bajo consumo de energía, hacen posible que se las utilice en todos o en la mayoría de áreas en las que se requiera monitorizar, estudiar y almacenar datos recogidos desde un sensor. Aquí se presentan diferentes áreas de aplicación de las WSN, para dar una idea las características de las mismas, haciendo más común su desarrollo así como también uso.

2.2.3.1 Aplicaciones Militares

El apoyo militar fue el primer propósito por el cual empezó a investigarse esta área. Tener conocimiento en tiempo real del campo de batalla es esencial para el control, las comunicaciones y la toma de decisiones.

Las WSNs pueden ser parte integral de sistemas militares que llevan las órdenes, el control, comunicaciones, procesamiento, inteligencia, vigilancia, reconocimientos y objetivos militares, como se ve en la Figura II.14.

Figura II.14 Aplicaciones Militares

Fuente: http://i41.tinypic.com/2sb5xsp.jpg

El rápido y denso despliegue de las redes de sensores, su auto-organización y tolerancia a fallos las hace una buena solución para aplicaciones militares. Ofrecen una solución de bajo coste y fiable para éstas ya que la pérdida de un nodo no pone en riesgo el éxito de las operaciones.

2.2.3.2 Aplicaciones en la Agricultura

La agricultura es una de las áreas donde las redes de sensores tienen una gran repercusión ya que se necesita la recolección de datos que quizá para el hombre sea imperceptible como por ejemplo la cantidad de CO_2 en el medio o los radiación que esté presente en determinada área de sembrío como lo muestra la Figura II.15, esto se logra con el uso de redes de sensores.

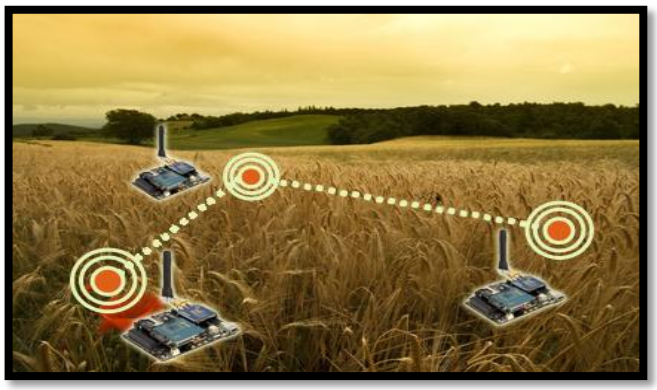

Figura II.15: Aplicaciones en Agricultura

Fuente: http://www.probesrl.net/eng/wp-content/uploads/2010/06/campo.jpg

Otros factores importantes de monitoreo son:

- Control de la cantidad de agua, fertilizante o pesticida que las plantas necesitan.
- Medida de la humedad del suelo.
- Decisión del momento óptimo para realizar la cosecha.
- Optimización de la producción y la calidad de una cosecha.
- Gestión de alarmas por intrusión de animales o daños provocados por heladas.

Como ejemplo, los viñedos Camalie, en Estados Unidos, tienen uno de los sistemas más avanzados de medida de la humedad del suelo. Usan una tecnología inalámbrica desarrollada por la Universidad de Berckeley en colaboración con Intel y comercializada por Crossbow. La aplicación consiste en optimizar la irrigación, reduciendo el consumo de agua y mejorando la calidad de la uva. [18]

2.2.3.3 Aplicaciones en Medio ambiente

En esta área se aprecian aplicaciones como el seguimiento de aves, animales e insectos; monitorización de condiciones ambientales que afectan al ganado y las cosechas; irrigación; macroinstrumentos para la monitorización planetaria de gran escala; detección química o biológica; agricultura de precisión (monitorización de niveles de pesticidas, polución y erosión del terreno); detección de incendios; investigación meteorológica o geofísica; detección de inundaciones; mapeado de la biocomplejidad del entorno; y estudios de la polución.

Un ejemplo en nuestro país es un proyecto multidisciplinario que está investigando el uso de redes de sensores inalámbricos para monitorear erupciones de volcanes activos y peligrosos tal como se muestra en la Figura II.16. Además de que estas redes tienen una tremenda ventaja sobre instrumentación existente utilizado en estudios de campo volcánicas. Esta tecnología permitirá a conjuntos de sensores de mayor resolución espacial y aberturas más grandes que las estaciones de monitoreo con cables existentes.

Se han desplegado tres redes de sensores inalámbricos en los volcanes activos. La implementación inicia en el volcán Tungurahua, Ecuador, en julio de 2004 sirvió como una prueba de concepto y consistía en un pequeño conjunto de nodos inalámbricos captura de datos de infrasonido continua. El segundo despliegue en el volcán Reventador, Ecuador, en julio-agosto de 2005 constaba de 16 nodos desplegados sobre una abertura de 3 kilometros en los flancos superiores del volcán, y se mide tanto las señales sísmicas y de infrasonidos con alta resolución (24 bits por canal a 100 Hz). [19].

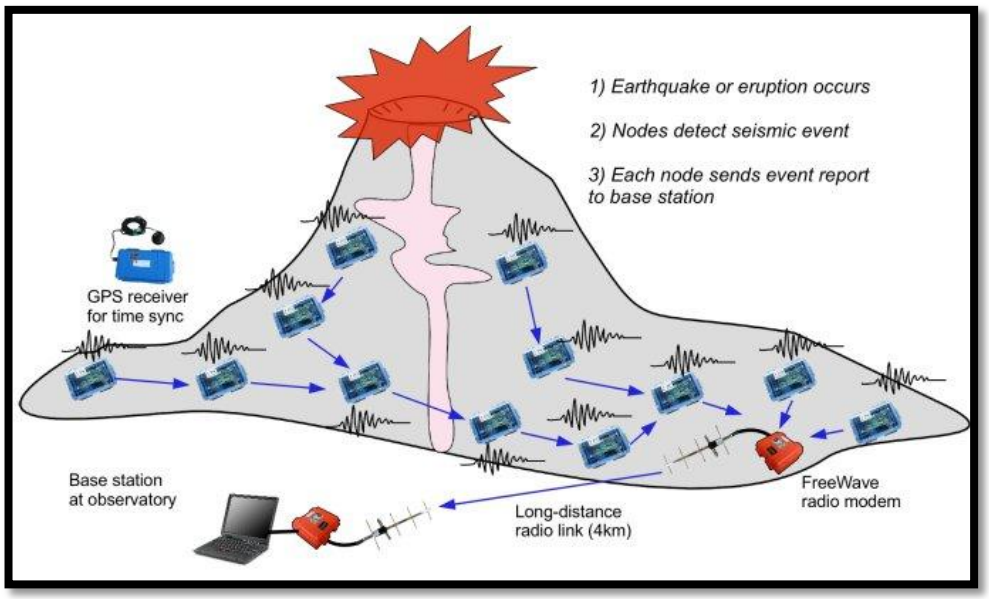

Figura II.16: Aplicación de WSN en Volcanes del Ecuador.
Fuente: http://fiji.eecs.harvard.edu/Volcano

2.2.3.4 Aplicaciones en la Automoción

El abanico de posibilidades que ofrecen los sensores en el ámbito vehicular, y las posibles topologías de red que se pueden encontrar son muy numerosos. Sin sensores es casi imposible realizar la gran cantidad de funciones de monitoreo y control en sistemas modernos de gestión del motor, sistemas de seguridad y confort (ASR, ABS, airbag, ajuste del cinturón de seguridad, aire acondicionado, etc.).

Como muestra el ejemplo de la Figura II.17 las redes de sensores son el complemento ideal a las cámaras de tráfico, ya que pueden informar de la situación del tráfico en ángulos muertos que no cubren las cámaras y también pueden informar a conductores de la situación, en caso de atasco o accidente, con lo que estos tienen capacidad de reacción para tomar rutas alternativas. [20].

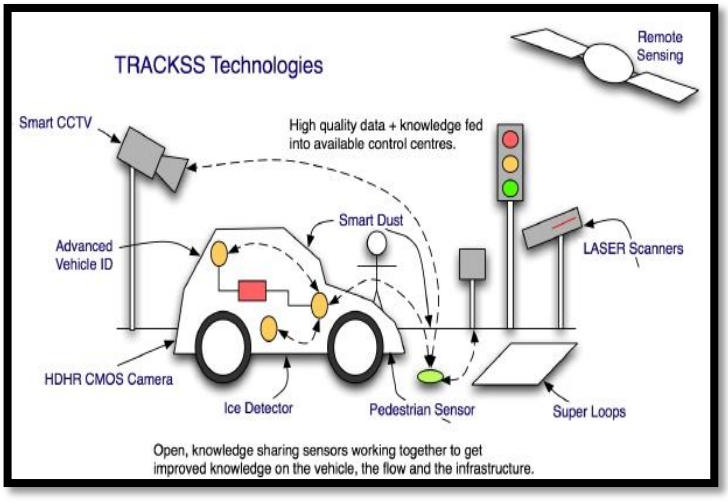

Figura II.17 Aplicaciones de Automoción

Fuente: http://www.trackss.net/imagenes/trackss-technologies.jpg

Los sensores se usan para registrar de manera precisa los estados reales del motor en funcionamiento, tales como la presión de aceite del motor, la temperatura del motor o el número de revoluciones. Cada vez se van incorporando un mayor número de sensores en el vehículo, ofreciendo más sistemas de seguridad y confort al conductor.

Existen numerosas iniciativas y consorcios de investigación, que pretenden desarrollar servicios futuristas de muy diversa índole, algunos proyectos de investigación como Trackss [21] hacen uso de sensores para construir las aplicaciones del futuro.

2.2.3.5 Aplicaciones en la Domótica

En los años 80 nace, principalmente en Estados Unidos, el término de casas inteligentes (smart house) luego en esta misma década se comienza a hablar en Europa del término Domótica, el cual no es más que la integración de servicios y tecnologías, aplicadas a hogares y pequeños edificios para automatizarlos como se puede observar en la Figura II.18. En la actualidad a partir de las ventajas que aporta la utilización de las WSN en esta área se han desarrollado una gran variedad de aplicaciones en este ámbito.

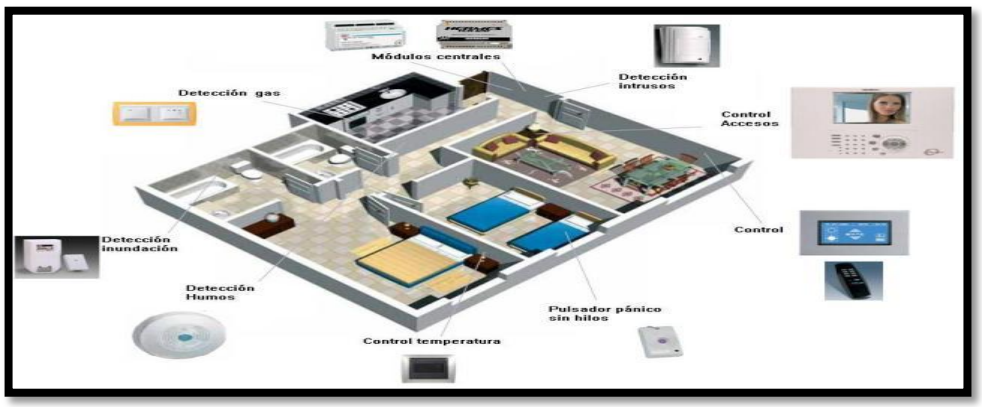

Figura II.18 Aplicación domótica

Fuente: http://t0.gstatic.com/images?q=tbn:ANd9GcQDs26Vajc3EfqkkA-MAh2NRmtt8KbMZ-Rn4_WaW_6NY814AytZ

El objetivo de esta aplicación es programar y configurar sensores para adaptarlos a entornos del hogar, posibilitando aplicaciones que favorezcan los objetivos de la domótica.

Los nodos sensores pueden ser introducidos en aparatos domésticos como aspiradoras, microondas, hornos, frigoríficos y VCRs. Esto permite que sean manejados remotamente por los usuarios finales mediante una comunicación que se realizaría vía satélite o Internet. Se crea lo que se llama un entorno inteligente, cuyo diseño puede tener dos enfoques:

- Desde el punto de vista humano: el entorno se adapta a necesidades del usuario final en términos de capacidades de entrada-salida.
- Desde el punto de vista tecnológico: hay que desarrollar nuevas tecnologías hardware, soluciones de redes y servicios middleware. [22]

2.2.3.6 Aplicaciones en el Monitoreo de Estructuras

Las estructuras físicas como puentes, edificios y construcciones en general experimentan vibraciones, ya sea ocasionada por actividades normales para las que fueron construidas o por fenómenos naturales. Las variaciones en los comportamientos indican fatiga u otros cambios mecánicos En este sentido algunos países como Estados Unidos y Canadá han invertido grandes cantidades de dinero, se estima que como 25 trillones de dólares en estructuras civiles, y ante tanta inversión se desea tener un control de las estructuras realizadas. La tecnología utilizada se llama SHM (Structural Health Monitoring), para la identificación y monitoreo de comportamientos extraños de

dichas estructuras como puentes, edificios y otras estructuras que fueron edificadas en los años noventa.

Un proyecto que se lleva actualmente en San Francisco, California, es el de evaluar las vibraciones del puente Golden Gate. Una placa de sensores se conecta a una Mica Mote y se coloca en el puente para realizar el análisis de movimiento debido al paso de los automóviles. En la Figura II.19 se muestra la colocación de los sensores en el puente:

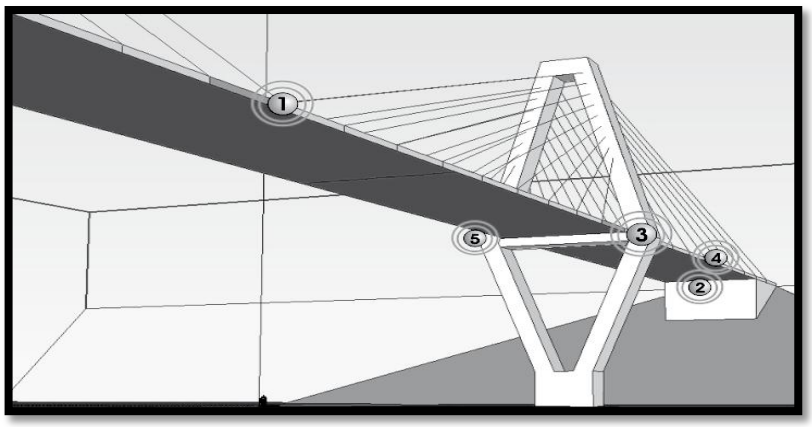

Figura II.19: Aplicación Estructuras.

Fuente: http://t0.gstatic.com/images?q=tbn:ANd9GcQDs26Vajc3EfqkkA-MAh2NRmtt8KbMZ-Rn4_WaW_6NY814AytZ

2.2.3.7 Aplicaciones en la Salud

En el ámbito de la salud, algunas de las posibilidades que brindan las WSN son: la monitorización de pacientes, como se ve en la Figura II.20, la administración de medicinas, el seguimiento de la enfermedad y la localización de pacientes.

Figura II.20: Aplicaciones en la Salud

Fuente: http://t2.gstatic.com/images?q=tbn:ANd9GcT2wLdJ3pE_5EzDIh9a3sfKlDApEGlA85ASIaOoFNaCEBDghf6h

Estas tecnologías proporcionan a los pacientes mayor confianza y mejoran su calidad de vida ya que se puede dar una detección más rápida de estados anormales y una supervisión continua.

2.3 MÓDULOS DE COMUNICACIÓN XBEE

Estos módulos son pequeños chip azules capaces de comunicarse de forma inalámbrica unos con otros. Pueden hacer cosas simples, como reemplazar un par de cables en una comunicación serial, lo cual es relevante cuando se desea diseñar por ejemplo una Red de Sensores Inalámbricos.

Existen muchos tipos diferentes de módulos XBee, pero una de las ventajas de estos XBee, es que todos, independiente del modelo o serie, tienen los pines similares. Alimentación, tierra y los pines de comunicación (TX/RX) se encuentran en el mismo lugar, haciendo que los chip sean totalmente intercambiables, para la mayoría de las aplicaciones más simples. Algunas de las características más avanzadas no son siempre compatibles.

Para entender mejor de que se trata se dará a conocer los detalles de estos dispositivos así como los protocolos que se manejan en el mismo.

Los módulos XBee son módulos de radio frecuencia que trabajan en la banda de 2.4 GHz con protocolo de comunicación 802.15.4.Son utilizados en automatización de casas, sistemas de seguridad, monitoreo de sistemas remotos, aparatos domésticos, alarmas contra incendio, plantas tratadoras de agua, etc., A continuación en la Figura II.21 se ven los modelos de módulos XBee.

Figura II.21 Ejemplos de módulos XBee

Fuente: http://webdelcire.com/wordpress/wp-content/uploads/2012/02/moduloszb.jpg

2.3.1 Circuito básico para XBEE

La Figura II.22 muestra las conexiones mínimas que necesita el módulo XBee para poder ser utilizado. Luego de esto, se debe configurar según el modo de operación que se desea para la aplicación requerida por el usuario.

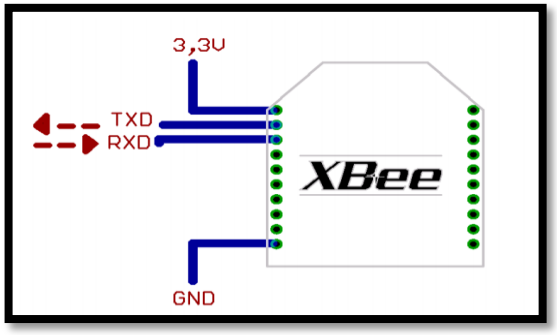

Figura II.22: Conexiones mínimas requeridas para módulo XBEE

Fuente: http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf

El módulo requiere una alimentación desde 2.8 a 3.4 V, la conexión a tierra y las líneas de transmisión de datos por medio del UART (TXD y RXD) para comunicarse con un microcontrolador, o directamente a un puerto serial utilizando algún conversor adecuado para los niveles de voltaje.

Esta configuración, no permite el uso de Control de Flujo (RTS & CTS), por lo que ésta opción debe estar desactivada en el terminal y en el módulo XBEE. En caso de que se envíe una gran cantidad de información, el buffer del módulo se puede sobrepasar. Para evitar existen dos alternativas:

- Bajar la tasa de transmisión.
- Activar e control lde flujo.

2.3.2. Modos de Operación

Los módulos XBee pueden operar en los siguientes modos como se muestra en la Figura II.23. Cada uno de estos modos tiene funciones específicas. Así los modos de operación son:

- Modo de Transmisión
- Modo de Recepción
- Modo de comando
- Modo de bajo consumo (Sleep)

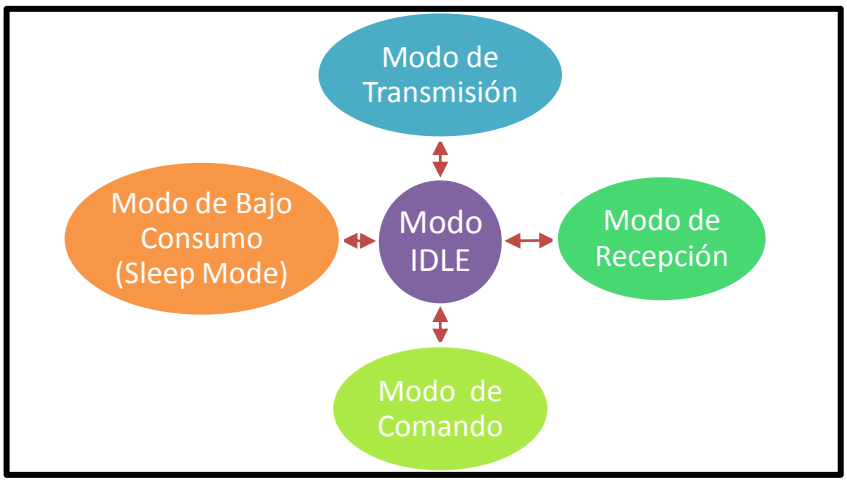

Figura II.23: Modos de Operación del módulo XBEE

Fuente: Elaboración Propia

2.3.1.1 Modo RECIBIR/TRANSMITIR.

Se encuentra en estos modos cuando al módulo le llega algún paquete RF a través de la antena(modo Receive) o cuando se manda información serial al buffer del pin 3 (UART Data in) que luego será transmitida (modo Transmit).

La información transmitida puede ser Directa o Indirecta. En el modo directo la información se envía inmediatamente a la dirección de destino. En el modo Indirecto la información es retenida durante un período de tiempo y es enviada sólo cuando la dirección de destino la solicita.

Además es posible enviar información por dos modos. Unicast y Broadcast. Por el primero, la comunicación es desde un punto a otro, y es el único modo que permite respuesta de quien recibe el paquete RF, es decir, quien recibe debe

enviar un ACK (paquete llamado así, y que indica que recibió el paquete, el usuario no puede verlo, es interno de los módulos) a la dirección de origen. Quien envió el paquete, espera recibir un ACK, en caso de que no le llegue, reenviará el paquete hasta 3 veces o hasta que reciba el ACK. En el modo Broadcast la comunicación es entre un nodo y a todos los nodos de la red. En este modo, no hay confirmación por ACK.

2.3.1.2 Modo de Bajo Consumo (Sleep Mode).

El modo sleep hace posible que el módulo RF entre en un modo de bajo consumo de energía cuando no se encuentra en uso.

Para poder entrar en modo sleep, se debe cumplir una de las siguientes condiciones:

- Sleep_RQ(pin 9) está en alto y el módulo está en pin sleep mode (SM= 1,2 o 5)
- El módulo está en reposo (no hay transmisión ni recepción de datos) por la cantidad de tiempo definido por ST (Time before Sleep). [ST sólo está activado cuando SM=4,5]

La configuración de los ciclos sleep se realiza principalmente con el comando SM. Por defecto, los modos de Sleep están deshabilitados (SM=0), permaneciendo el módulo en estado de reposo/recepción. En este estado el módulo está siempre preparado para responder a un comando, ya sea, por el puerto serial o la interfaz RF.

A continuación se presenta en la Tabla II.II un resumen, considerando los

modos de operación, consumos, voltajes y condiciones para entrar en modo de bajo consumo.

Tabla II.II: Modo Sleep y consumos de corriente.

Modo	Consumo alimentación			Modo Sleep	Modo Wake -up
	2.8 – 3 V	3.2 V	3.4V		
SM=1	<3 uA	32 uA	255 uA	Sleep_RQ	Sleep_RQ
SM=2	<35 uA	48 uA	170 uA	Sleep_RQ	Sleep_RQ
SM=3		(reservado)		(reservado)	(reservado)
SM=4	<34 uA	49 uA	240 uA	Comando ST	Comando SP
SM=5	<34 uA	49 uA	240 uA	Comando ST	Sleep RQ

Fuente: http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario

2.3.1.3 Modo de Comando.

Este modo permite ingresar comandos AT al módulo XBee, para configurar, ajustar o modificar parámetros. Permite ajustar parámetros como la dirección propia o la de destino, así como su modo de operación entre otras cosas. Para poder ingresar los comandos AT es necesario utilizar el Hyperterminal de Windows, el programa X-CTU 3 o algún microcontrolador que maneje UART y tenga los comandos guardados en memoria o los adquiera de alguna otra forma.

Para ingresar a este modo se debe esperar un tiempo dado por el comando GT (Guard Time, por defecto ATGT=0x3E84 que equivalen a 1000ms) luego ingresar +++ y luego esperar otro tiempo GT. Como respuesta el módulo entregará un OK. El módulo XBee viene por defecto con una velocidad de 9600bps.

En caso de no poder ingresar al modo de comandos, es posible que sea debido a la diferencia de velocidades entre el módulo y la interfaz que se comunica vía serial como se muestra en la Figura II.24. [23]

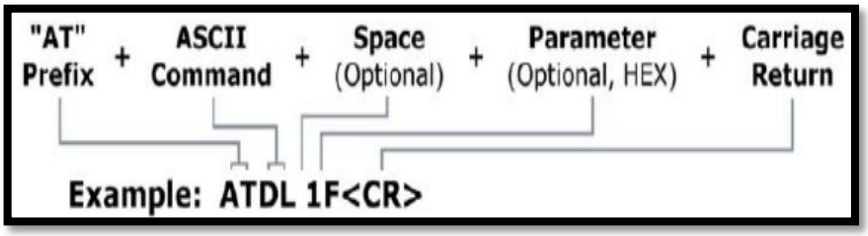

Figura II.24: Ejemplo de comandos AT

Fuente: http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf

2.3.1.4 Modo Idle

Cuando el módulo no se está en ninguno de los otros modos, se encuentra en éste. Es decir, si no está ni transmitiendo ni recibiendo, ni ahorrando energía ni en el modo de comandos, entonces se dice que se encuentra en un estado al que se le llama IDLE.

2.3.2 Modelos de módulos XBEE

Actualmente MaxStream, que ahora forma parte de Digi International, es una empresa líder mundial en el desarrollo de módems de conexión a redes inalámbricas para dispositivos electrónicos. MaxStream dispone de módulos de módems inalámbricos, módems por radio independientes, servicios de diseño de radiofrecuencia y el software correspondiente [24], es importante saber cada una de las características de estos módulos antes de decidirnos por comprar determinado modelo ya que el uso va estrictamente ligado a la aplicación en la que se vaya a trabajar.

2.3.2.1 Series Módulos XBEE

Hay varias versiones de módulos XBee por lo que se va a dar un breve resumen de cada una de las versiones de los módulos XBEE así se tiene:

- **XBee Series 1 (también llamados XBee 802.15.4**): Son la serie más fácil para trabajar, no necesitan ser configurados, pero incluso así se pueden obtener beneficios. Para comunicaciones Punto-a-Punto, estos módulos trabajan tan bien como los de la Serie 2, pero sin todo el trabajo de pre configuración previa. Un punto clave antes de decidirse por determinado hardware es que las Series 1 y las Series 2/2.5/ZB *no son compatibles*.

- **XBee Znet 2.5 (Formalmente Series 2)** Retirado - Estos son los divertidos. Los módulos Serie 2 deben ser configurados antes de ser usados, pues pueden funcionar en los diferentes modos antes mencionados, por ejemplo en modo Transparente o por medio de comandos API, pero todo esto depende de cual firmware se configure en los módulos. También pueden funcionar en una red mesh, creando unos módulos totalmente configurables e increíbles. Pero también son más difíciles que usar que los de la Serie 1. Los módulos Znet 2.5 ya no se venden, pero han sido reemplazados con módulos ZB más compatibles.

- **ZB (módulo Series2)** - Básicamente es el módulo Znet 2.5, pero con un nuevo firmware. Esto significa que también funcionan en modo transparente o por medio de comandos API. También funcionan en redes mesh. Puedes tomar el nuevo firmware y actualizarlo por sí mismo. El firmware entre ambos no es compatible (pero es fácilmente intercambiable) por lo que debes elegir cuál

firmware deseas usar, y apegarte a éste para crear toda la red. Estos a menudo son llamados módulos de Serie 2.

- **2B (Series2)** - Son nuevos módulos que poseen mejoras en el hardware respecto de los de la Serie 2, mejorando por ejemplo el uso de la potencia. Funcionan con el Firmware del módulo ZB, pero debido al cambio de hardware, ya no pueden funcionar con el firmware del módulo Znet 2.5. Por lo que se debe tener cuidado se agrega uno de estos módulos a una red ya existente que utilice módulos Znet 2.5. Actualmente algunas de nuestras tarjetas son 2B y otras son ZB.

- **900MHz** - Técnicamente no es una Serie, pero sí es una familia tal como los otros. Estos módulos pueden funcionar con dos diferentes tipos de firmware, el firmware DigiMesh y el firmware Point-to-Multipoint. Digi actualmente vende ambos módulos, el hardware es el mismo, pero con diferentes firmware. OLIMEX solamente vende la versión Point-to-Multipoint, pero el firmware se puede cambiar fácilmente. Estos módulos son más o menos Plug and Play pero por supuesto no se podría tener beneficio de todas las grandes características que en el módulo se pueden configurar.

- **XSC** - Son, básicamente, módulos de 900 MHz, pero sacrifican velocidad de datos por el alcance. Los módulos 900 MHz tienen una velocidad de datos de aproximadamente 156 Kbps (los otros de aproximadamente 250 Kbps), mientras que en los módulos XSC es de alrededor de 10 Kbps. Por otro lado, si colocas una antena de alta ganancia puedes tener un alcance de alrededor de 24 Km y de 9,6 Km con una antena regular. Estos módulos no requieren

configuración externa y tienen otras diferencias incluyendo un set de comandos diferente a los anteriores por lo que se recomienda revisar con anterioridad la hoja de comandos. [25]

Otras características de los módulos se explican a continuación:

- **Regular vs Pro -** Hay pocas diferencias entre un XBee regular y un XBee PRO. Los XBee PRO son un poco más largos, consumen más potencia y cuestan un poco más de dinero. El mayor consumo de potencia quiere decir también mayor alcance 1,6 Km en vez de 91,5 m. Los dos modelos se pueden mezclar dentro de la misma red.

- **900 vs 2.4** - La mayoría de los módulos XBee operan a 2.4 GHz, pero hay unos pocos que operan a 900 MHz. Básicamente los de 900 MHz pueden llegar muy lejos con una antena de alta ganancia (hasta casi 24 Km). Además a menor frecuencia, la señal posee mayor penetración. Otro punto importante es que los módulos de 900 MHz no están permitidos en algunos países, incluso Digi tiene versiones de 868 MHz que sí está permitido en la mayoría de los países. Estas dos versiones no se pueden mezclar en la misma red.

2.3.2.2 Tipo de Antenas módulos XBEE

Se hablará de las diferentes antenas que componen un módulo XBEE las cuales se pueden visualizar en la Figura II.25, entre las cuales se tienen:

- *Chip Antenna***:** Básicamente es un pequeño chip que actúa como antena. Rápido, sencillo y barato.

- *Wire Antenna (Whip Antenna)*: Es un pequeño cable que sobresale del módulo.

- *u.FL Antenna*: Es un conector pequeño para conectar tu propia antena. Esto es perfecto si tienes tu equipo en una caja y deseas la antena afuera de ésta.

- *RPSMA Antenna***:** Un conector más grande para conectar una antena externa. Nuevamente, esto es perfecto si tienes tu equipo en una caja y deseas la antena afuera de ésta.

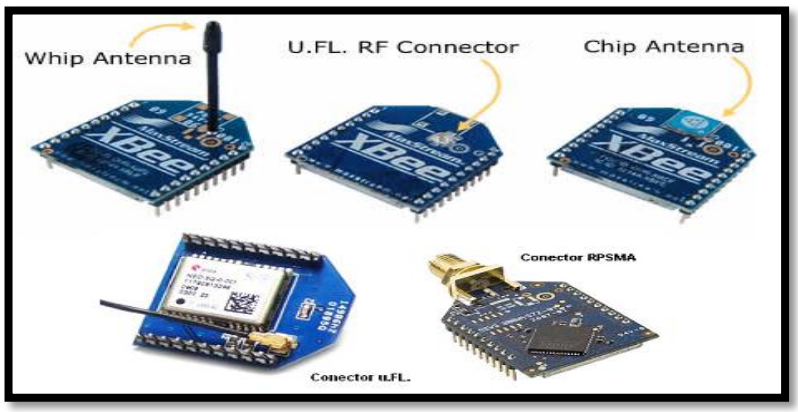

Figura II.25: Tipos de antenas en módulos XBEE

Fuente: http://www.olimex.cl/images/tutorial-XBee/tipos%20de%20antena.JPG

MaxStream fabrica más de 70 tipos de módulos XBee con diferentes antenas, potencia y capacidades. Muchas de las características de los módulos XBee tales como velocidad de transmisión y canales por ejemplo pueden ser configurados utilizando el software X-CTU o directamente desde tu microcontrolador. A continuación se muestra en la Tabla II.III de resumen con las características esenciales de cada uno de los diferentes módulos.

Tabla II.III: Comparativa de módulos XBEE

XBee	Max data rate	Frequency band	Transmit power	Antenna	IO pins digital	ADC inputs	Range
XBee 1mW PCB Antenna	115.2 kbP	2.4 GHz	1 mW (+0 dBm)	Built-in	8	(7) 10-bit	300ft (100m)
XBee 1mW Wire Antenna	250kbps	2.4 GHz	1mW output (+0dBm)	Built-in	8	(6) 10-bit	300ft (100m)
XBee 2mW Chip Antenna - Series 2	250kbps	2.4 GHz	2mW output (+3dBm)	Built-in	8	(6) 10-bit	400ft (120m)
XBee 2mW PCB Antenna - Series 2 (ZigBee Mesh)	250kbps	2.4 GHz	2mW output (+3dBm).	Built-in	8	(6) 10-bit	400ft (120m)
XBee 2mW RPSMA - Series 2	250kbps	2.4 GHz	2mW output (+3dBm)	RPSMA	8	(6) 10-bit	400ft (120m)
XBee 2mW Wire Antenna - Series 2 (Mesh)	250kbps	2.4 GHz	2mW output (+3dBm)	Built-in	8	(6) 10-bit	400ft (120m)
XBee Pro 50mW RPSMA - Series 2 (Mesh)	250kbps	2.4 GHz	50mW output (+17dBm).	RPSMA	8	(6) 10-bit	1 mile (1600m)
XBee Pro 50mW Serie 2.5 Wire Antena	250kbps	2.4 GHz	50mW output (+17dBm).	Built-in	8	(4) 10-bit	1 mile (1600m)
XBee Pro 60mW serie 1 PCB Antena	250kbps	2.4 GHz	60mW output (+18dBm).	Built-in	8	(6) 10-bit	1 mile (1600m)
XBee Pro 60mW Wire Antenna	250kbps	2.4 GHz	60mW output (+18dBm)	Built-in	8	(6) 10-bit	1 mile (1600m)
XBee Pro 900 RPSMA	156 Kbps	ISM de 900MHz	50 mW (+17 dBm)	RPSMA	10	(6) 10-bit	6 miles (10 km)
XBee Pro 900 XSC RPSMA	9.6kbps	ISM de 900MHz	100 mW power output	RPSMA	none	none	15 miles

Fuente: http://www.XBee.cl/caracteristicas.html

2.3.3 Componentes Adicionales

Para poder integrar los módulos en nuestros proyectos sin necesidad de incómodos cables se necesita hardware adicional tal como: XBee Explorer, XBee Explorer Regulado, XBee shield, XBee breakouts.

Se debe conocer la función de estos dispositivos ya que puede que sirva o no para nuestros proyectos, por lo que se debe establecer desde un comienzo qué es lo que se necesita.

- **XBee Explorer**

Se usa en los módulos de Series 2/2.5/ZB para conectarlos a un PC e instalar el firmware. También son útiles para cambiar la configuración, y te permite enviar y recibir información en tu PC. Viene tanto en una versión microB USB como en una para conectar directamente al puerto USB como se ve en la Figura II.26.

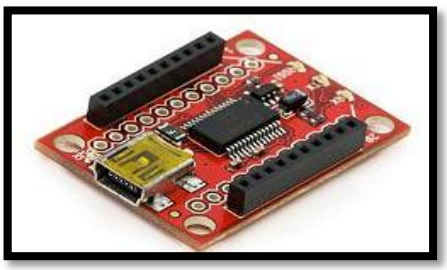

Figura II.26: XBee Explorer USB

Fuente: http://webdelcire.com/wordpress/wp-content/uploads/2012/02/XBeeexplorerusb.jpg

- **XBee Explorer Regulated**

Es ideal para incluir tu XBee a tu propio circuito de 5V. Es básicamente un Breakout con un regulador de 3.3 V incluido mostrado en la Figura II.27.

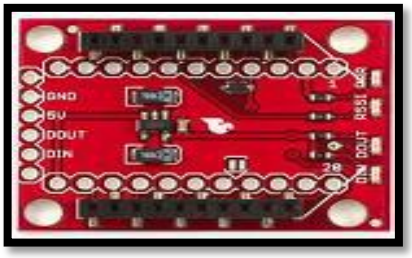

Figura II.27: XBee Explorer Regulated

Fuente: http://www.olimex.cl/images/tutorial-XBee/XBeexplorerregulated.JPG

- **XBee Shield**

Esta tarjeta es ideal para incluir un módulo Arduino que se verá en la Figura II.28 en conjunto con un módulo XBee en el circuito. Tiene un regulador de 3.3 Volt incluido. La versión actual posee un switch que permite conectar el XBee a los pines UART del Arduino, o a otro diferente.

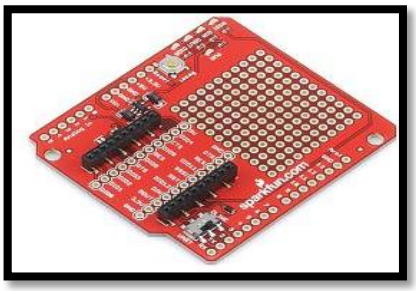

Figura II.28: XBee Shield

Fuente: http://www.olimex.cl/images/tutorial-XBee/XBeeshield.jpg

2.3.4 Recursos básicos necesarios para módulo XBee.

Se mencionará algunos recursos para configurar un módulo, entre los principales se cuentan con:

- **X-CTU software** - Este programa permite configurar los módulos XBee. El programa sólo está disponible en Windows, en la Figura II.29 se observa una ventana de software mencionado.

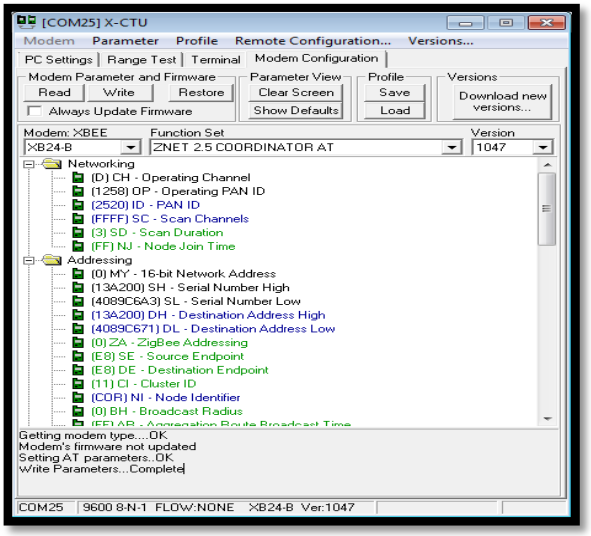

Figura II.29: X-CTU

Fuente: http://tdrobotica.co/images/stories/tutoriales/comunicaciones/T04COM/COR.png

- **Regulaciones Gubenamentales** - La comunicación inalámbrica tiene diferentes restricciones dependiendo del país. Los que veremos, no tienen problemas para usarse en el Ecuador, pero para una mayor información, verificar la información de Digi sobre cuál XBee es aceptable dependiendo del país.
- **Bibliografía** - Información de los módulos que cubra todo, desde la configuración de los módulos hasta el uso de los pines de Entrada/Salida (I/O) y funciones Sleep para ahorro de energía.

2.3.5 Tipos de Redes que soporta XBee.

Los módulos XBee, pueden ser ajustados para usarse en redes de configuración punto-a-punto o punto-a-multipunto. Teniendo en cuenta que los módulos XBee Serie 2 permiten hacer redes *mesh*, y la serie 1 no, que es la de la que se dispone en este proyecto.

2.3.5.1 Conexión Punto a Punto.

Un módulo se conecta con otro como se muestra en la Figura II.30.

Figura II.30: Conexión punto a punto

Fuente:http://wiki.ubnt.com/images/thumb/e/e2/Prouctselection_4.png/360pxProductselection_3.png

Para ello se utilizan los comandos MY y el DL para configurar la dirección. Este tipo de conexión ha sido el utilizado durante el desarrollo de la primera parte de este proyecto hasta conseguir una familiarización con los dispositivos XBee y su funcionamiento.

2.3.5.2 Conexión Punto a Multipunto.

En la Figura II.31 se observa una red multipunto que permite transmitir información desde la entrada serie de un módulo a uno o varios módulos conectados a la misma red de manera más controlada, ya que se necesitan las direcciones de los otros módulos, por lo que existe mayor seguridad. Para esto se necesitan dos comandos más aparte de MY y DL. Se utilizara el direccionamiento de 16 bits.

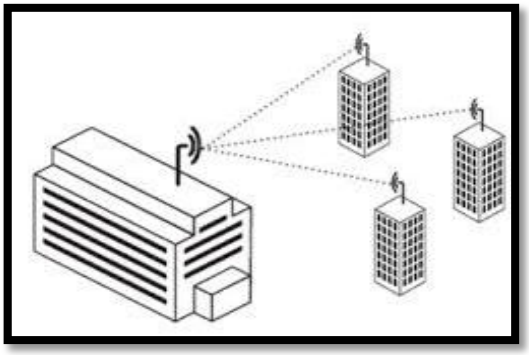

Figura II.31: Conexión Multipunto

Fuente:http://wiki.ubnt.com/images/thumb/e/e2/Prouctselection_4.png/360pxProductselection_4.png

El primer comando es el ID de la PAN (*Personal Area Network*- Red de Área Personal). Todos los módulos que tengan idéntico PAN ID, pertenecerán a la misma red. El comando para configurar este valor es ID, es decir, ATID, y su rango va entre 0x0 y 0xFFFF. La Figura II.31 representa el tipo de red punto a multipunto y como se configura. La elección del canal debe ser cuidadosa, ya que otras tecnologías como WiFi o bluetooth utilizan el mismo espectro de frecuencias, por lo que se podría producir interferencia.

2.3.5.3 Conexión NonBeacon. *Peer to Peer*.

Una red *peer-to-peer* permite que todos los módulos se conecten con todos, es decir, se crea una conexión de par en par con cada uno de los módulos de la red. El modo de conexión *Non-Beacon* es la configuración por defecto y permite establecer una red *peer-to-peer* donde cada módulo puede hacer las funciones de maestro o esclavo.

La configuración de red *Non-Beacon*, se refiere a que cada nodo primario (o nodo central maestro) se mantiene despierto siempre. Por lo que los demás dispositivos que se conectan a él, pueden entrar en modo *sleep*, y solo

despertarse cuando sea necesario para enviar datos. En una red *Beacon*, los dispositivos enrutadores (o nodos primarios) están siempre en modo *SLEEP*, y envían señales de su existencia (llamadas *Beacon*) cada cierto intervalo al resto de la red. Así para poder comunicarse, deben estar todos los dispositivos totalmente organizados, ya que de no ser así, existe la posibilidad de perder la señal *Beacon* y no poder enviar hasta la próxima entrega. La ventaja de las redes *Beacon*, es el ahorro de energía. Por este motivo las redes *Non-Beacon* están pensadas para dispositivos que posean una alimentación segura, mientras las *Beacon*, para alimentación autónoma, como baterías. Los módulos XBee Series 1, solo soportan redes *Non-Beacon.*

2.3.5.4 Conexión *Non-Beacon* con coordinador.

Es básicamente lo mismo que una red punto multipunto con la diferencia de que existe un módulo central que posee ciertas propiedades y características que le permiten administrar la red. En esta red, el módulo central es llamado coordinador, mientras que el resto de módulos son llamados Dispositivos Terminales (*End Device*) como se representa en la Figura II.32. Un mismo módulo XBee puede ser configurado para funcionar como coordinador o como Dispositivo Terminal. Para configurar esta red, todos los módulos deben tener el mismo canal y la misma PAN. El modulo coordinador se configura como ATCE=1 (ATCE1), mientras que todos los demás, los cuales serán llamados Dispositivos Terminales, se configuran como ATCE=0 (ATCE0).

Este tipo de configuración se usa cuando se necesita una unidad central para enviar mensajes a varios módulos, o juntar información proveniente de varios Dispositivos Terminales, asignar canales o ID de redes PAN.

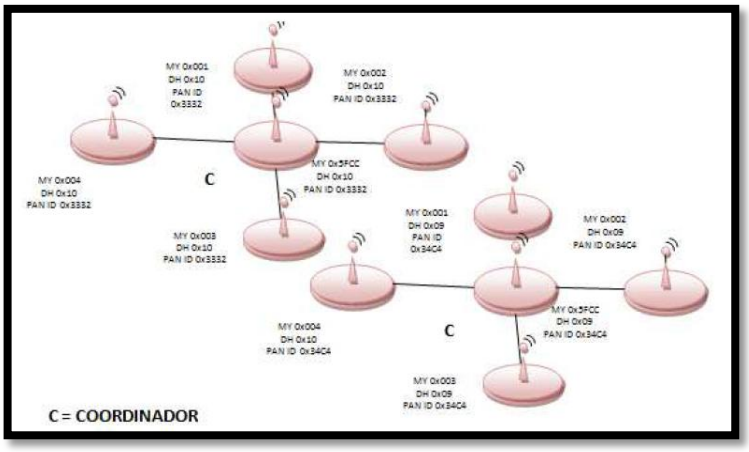

Figura II.32: Conexión Non-Beacon

Fuente: http://www.ni.com/cms/images/devzone/tut/clip_image004_20080328130917.jpg

2.4 ESTANDAR IEEE 802.15.4 Y ZIGBEE

La base de todas las normas y especificaciones de ZigBee es el potente estándar IEEE 802.15.4 que operan en bandas sin licencia en todo el mundo a 2,4 GHz (global), 915Mhz (América) y 868Mhz (Europa). Ofrece velocidades de datos de rendimiento de 250Kbs a 2,4 GHz (16 canales), 40Kbs a 915Mhz (10 canales) y 20Kbs a 868Mhz (1 canal). Distancias de transmisión son notables por una solución de bajo consumo, que van desde 10 a 1.600 metros, dependiendo de la potencia y las condiciones ambientales, tales como edificios, tipos de paredes interiores y la topología geográfica.

Todo sistema u organización de elementos se debe regir según normas que reglamenten su funcionamiento y aplicación. Las Redes de Sensores Inalámbricos no son la excepción. Este tipo de sistema se comunica a través de señales de radio, por lo cual tiene asignado un espectro de la señal electromagnética para conseguir la comunicación entre los dispositivos de la red. El estándar que fija las condiciones para que este enlace se produzca es el IEEE 802.15.4, existen más pero éste es el estándar más empleado. El IEEE

802.15.4 sirve de base para otras especificaciones como ZigBee cuyo propósito es ofrecer una solución completa para este tipo de redes definiendo los niveles superiores de la pila de protocolos que el estándar 802.15.4 no cubre.

2.4.1 Estándar IEEE 802.15.4

El estándar IEEE 802.15.4 [26] [27], cuya última revisión se aprobó en 2006, define una capa de comunicación que se encuentra en el nivel 2 (Enlace de datos) del modelo OSI. Aquí las unidades de la información digital (bits) son gestionados y organizados para convertirse en impulsos electromagnéticos (ondas) en el nivel inferior, el físico. Su objetivo principal es permitir la comunicación entre dos dispositivos. La característica más importante de este estándar es su flexibilidad de red, bajo coste, bajo consumo de energía.

Este estándar fue creado para llenar el hueco existente en el campo de estándares inalámbricos de baja tasa para aplicaciones en redes de sensores. Los estándares existentes hasta el momento en el mercado estaban destinados a aplicaciones con mayores requisitos en cuanto a ancho de banda se refiere, como pueden ser videoconferencias o redes domésticas.

Los ejemplos más representativos de estas tendencias son:

- IEEE 802.11 [28], también conocido como Wi-Fi

- IEEE 802.15.1 [29]conocido como Bluetooth, que es una tecnología de red inalámbrica de baja potencia y baja tasa para comunicaciones punto a punto.

- IEEE 802.15.3: [30]WPAN (Wireless Personal Area Network) de alta tasa de datos. Se utiliza en aplicaciones que requieren alta tasa de datos o una gran cobertura, lo que supone soluciones complejas con elevado consumo de potencia.

La dificultad que surgía al emplear cualquiera de éstos estándares, era su gran consumo de energía y ancho de banda frente a la baja tasa y bajos requisitos de energía necesaria para las redes de sensores. En el caso de Bluetooth no está diseñado para soportar la comunicación entre redes de varios nodos, por tanto, se necesita un nuevo estándar (IEEE 802.15.4) que cumpla con los criterios que se mencionaron anteriormente.

El requisito fundamental del estándar IEEE 802.15.4 es un consumo de potencia extremadamente bajo. Su eficiencia energética de este protocolo reside fundamentalmente en el uso de las tramas “Beacon”, que permiten sincronizar los dispositivos de la red para que puedan permanecer en modo ahorro de energía el mayor tiempo posible, esto supone una gran ventaja para el desarrollo WSN que realicen tanto tareas de monitorización como de control. El inconveniente es que, debido al bajo consumo de potencia, el radio de cobertura se ve reducido

2.4.2 Estándar Zigbee

Zigbee es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE_802.15.4. Creado por Zigbee Alliance, una organización, teóricamente sin ánimo de lucro, de más de 200

grandes empresas (destacan Mitsubishi, Honeywell, Philips, ODEM, Invensys, entre otras), muchas de ellas fabricantes de semiconductores. Zigbee permite que dispositivos electrónicos de bajo consumo puedan realizar sus comunicaciones inalámbricas. Es especialmente útil para redes de sensores en entornos industriales, médicos y, sobre todo domóticas.

Las comunicaciones Zigbee se realizan en la banda libre de 2.4GHz, este protocolo realiza las comunicaciones a través de una única frecuencia, es decir, de un canal. Normalmente puede escogerse un canal de entre 16 posibles. El alcance depende de la potencia de transmisión del dispositivo así como también del tipo de antenas utilizadas (cerámicas, dipolos, etc.) El alcance normal con antena dipolo en línea vista es de aproximadamente, tomando en cuenta módulos con antenas de 1mW de potencia es de 100m y en interiores de unos 30m. La velocidad de transmisión de datos de una red Zigbee es de hasta 256kbps. Una red Zigbee la pueden formar, teóricamente, hasta 65535 equipos, es decir, el protocolo está preparado para poder controlar en la misma red esta cantidad enorme de dispositivos. Entre las necesidades que satisface el módulo se encuentran:

- Bajo costo.
- Ultra-bajo consumo de potencia.
- Uso de bandas de radio libres y sin necesidad de licencias.
- Instalación barata y simple.
- Redes flexibles y extensibles.

El uso del protocolo Zigbee reemplaza un cable por una comunicación serial inalámbrica, hasta el desarrollo de configuraciones punto a punto, multipunto, peer-to-pee, mesh. Una conexión típica se muestra en la Figura II.33, donde se observa que cada nodo tiene una función específica, los cuales entregan los datos para ser enviados a través de la red a un coordinador que administre la información.

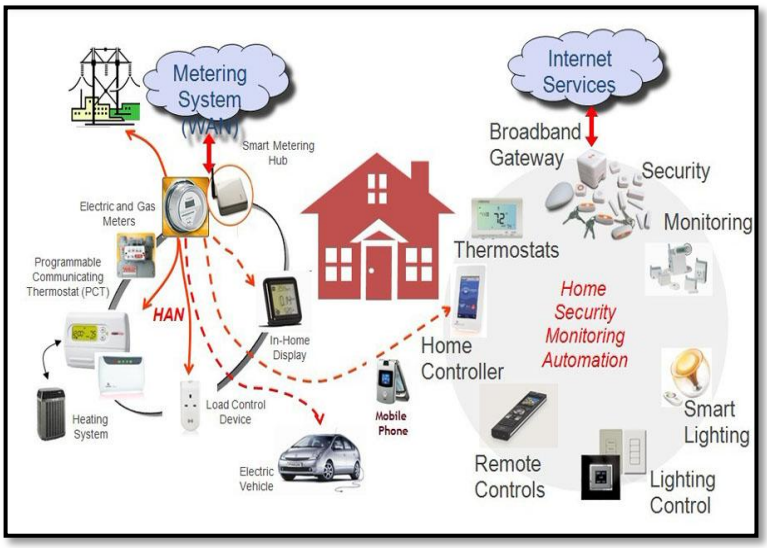

Figura II.33: Uso del protocolo de zigbee
Fuente: http://www.hoperf.com/upload/solutionimg/home_automation.jpg

Los beneficios del uso de estándares de ZigBee en los productos y servicios son convincentes:

- La interoperabilidad de dispositivos estandarizados permite a los fabricantes de productos para centrarse en la innovación de los productos frente a la creación de sus propios protocolos de red propietarios
- Los fabricantes de productos se pueden actualizar los productos existentes y añadir nuevas características innovadoras

- Las características innovadoras permiten nuevos productos para resolver problemas previamente irresolubles.

Desde sus anuncios ZigBee ha gozado de gran expectativa, incluso corrían los rumores que se trataba del reemplazo de Bluetooth, y no es para menos pues por ejemplo, el nodo ZigBee más completo requiere en teoría cerca del 10% del software de un nodo de Bluetooth o Wi-Fi típico; esta cifra baja al 2% para los nodos más sencillos, no obstante, el tamaño de código en sí es bastante mayor y se acerca al 50% del tamaño del de Bluetooth; no obstante como se observa en la Figura II.34, ZigBee ha surgido no para reemplazar a Bluetooth, pues sus campos de acción son distintos.

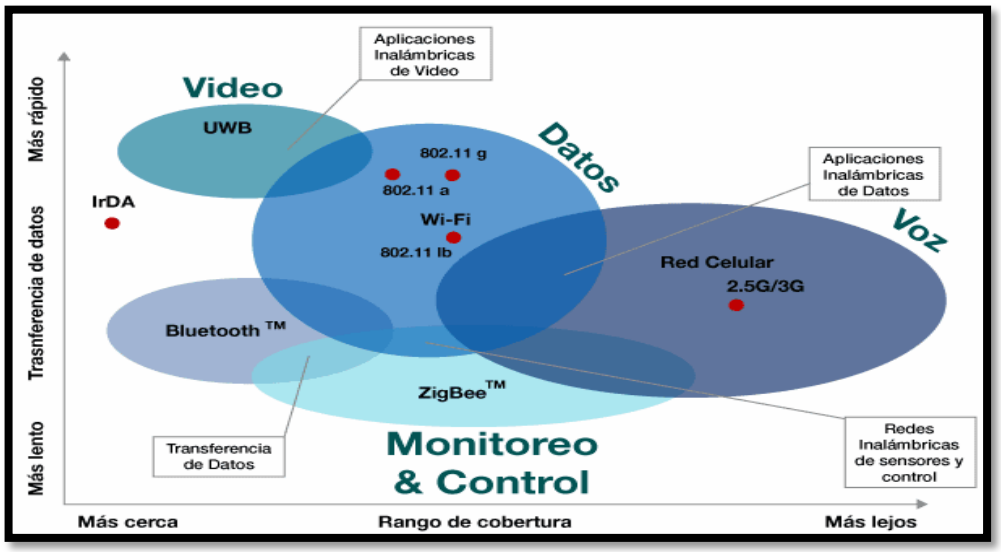

Figura II.34: Rango de Cobertura

Fuente. http://www.seccperu.org/files/ZigBee.pdf

2.4.2.1 Características De Zigbee

Algunas de las características más importantes son:

- ZigBee, también conocido como "HomeRF Lite", es una tecnología inalámbrica con velocidades comprendidas entre 20 kB/s y 250 kB/s.

- Los rangos de alcance son de 10 m a 75 m.
- Puede usar las bandas libres ISM (Industrial, Scientific and Medical) de 2,4 GHz (Mundial), 868 MHz (Europa) y 915 MHz (EEUU).
- Una red ZigBee puede estar formada por hasta 255 nodos los cuales tienen la mayor parte del tiempo el transceiver ZigBee dormido con objeto de consumir menos que otras tecnologías inalámbricas.
- Un sensor equipado con un transceiver ZigBee pueda ser alimentado con dos pilas AA durante al menos 6 meses y hasta 2 años.
- A pesar de coexistir en la misma frecuencia con otro tipo de redes como WiFi o Bluetooth su desempeño no se ve afectado, esto debido a su baja tasa de transmisión y, a características propias del estándar IEEE 802.15.4
- La fabricación de un transmisor ZigBee consta de menos circuitos analógicos de los que se necesitan habitualmente.
- Diferentes tipos de topologías como estrella, punto a punto, malla, árbol.
- Acceso de canal mediante CSMA/CA (acceso múltiple por detección de portadora con evasión de colisiones).
- Escalabilidad de red.- Un mejor soporte para las redes más grandes, ofreciendo más opciones de gestión, flexibilidad y desempeño.
- Fragmentación.- Nueva capacidad para dividir mensajes más largos y permitir la interacción con otros protocolos y sistemas.
- Agilidad de frecuencia.- Redes cambian los canales en forma dinámica en caso que ocurran interferencias.

- Gestión automatizada de direcciones de dispositivos.- El conjunto fue optimizado para grandes redes con gestión de red agregada y herramientas de configuración.
- Localización grupal.- Ofrece una optimización adicional de tráfico necesaria para las grandes redes.
- Puesta de servicio inalámbrico.- El conjunto fue mejorado con capacidades seguras para poner en marcha el servicio inalámbrico.
- Recolección centralizada de datos.- El conjunto fue sintonizado específicamente para optimizar el flujo de información en las grandes redes.
- Capacidad de operar en redes de gran densidad, esta característica ayuda a aumentar la confiabilidad de la comunicación, ya que entre más nodos existan dentro de una red, entonces, mayor número de rutas alternas existirán para garantizar que un paquete llegue a su destino.
- Cada red ZigBee tiene un identificador de red único, lo que permita que coexistan varias redes en un mismo canal de comunicación sin ningún problema. Teóricamente pueden existir hasta 16 000 redes diferentes en un mismo canal y cada red puede estar constituida por hasta 65 000 nodos, obviamente estos límites se ven truncados por algunas restricciones físicas (memoria disponible, ancho de banda, etc.).
- Es un protocolo de comunicación multi-salto, es decir, que se puede establecer comunicación entre dos nodos aún cuando estos se encuentren fuera del rango de transmisión, siempre y cuando existan otros nodos intermedios que los interconecten, de esta manera, se incrementa el área de cobertura de la red.

- Su topología de malla (MESH) permite a la red auto recuperarse de problemas en la comunicación aumentando su confiabilidad [4].

2.4.2.2 Áreas De Aplicación

En la actualidad existen muchos campos que han optado por utilizar esta tecnología, un gran número de las compañías que forman parte de la ZigBee Alliance, en la Figura II.35 donde se encuentran desarrollando productos impulsando los mercados de destino que más les interesa, entre ellos se menciona:

- Gestión de edificios comerciales
- Electrónica de consumo
- Gestión de la energía
- Cuidado de la salud y de la aptitud
- La administración del hogar
- Gestión de venta
- Telecomunicaciones

Figura II.35: Aplicaciones ZigBee

Fuente: Fuente: http://www.zigbee.org/About/AboutTechnology/Standards.aspx

Hay que tener en cuenta que ZigBee está diseñado para aplicaciones que transmiten unos cuantos bytes esporádicamente, que es el caso de una aplicación para automatizar el hogar (domótica). Al usar esta tecnología no habría la necesidad de cablear los interruptores, los cuales podrían ser cambiados de un lugar a otro con plena libertad, pudiendo por ejemplo, prender o apagar las luces de tu casa a través de Internet o utilizando tu teléfono celular en cualquier momento como se ve en la Figura II.36.

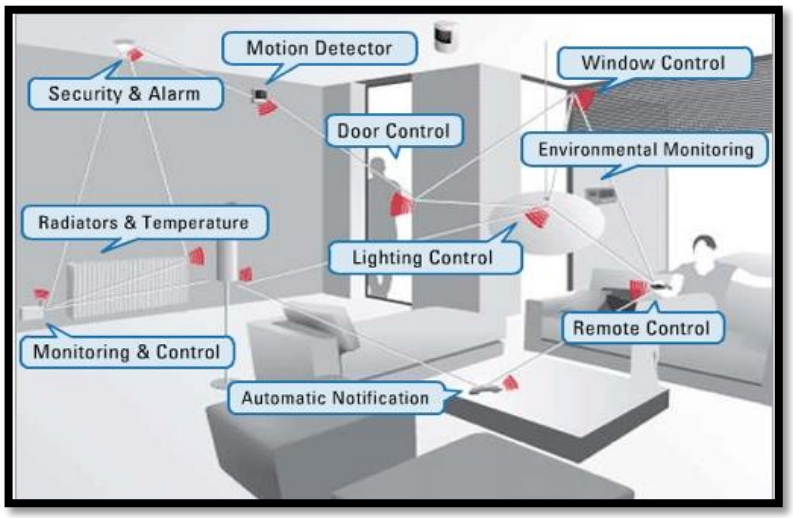

Figura II.36: Aplicaciones en Domótica

Fuente: http://sx-de-tx.wikispaces.com/ZIGBEE

Una de las áreas de aplicación que ha tomado fuerza, es la de los sistemas de medición avanzada, medidores de agua, luz y gas que forman parte de una red con otros dispositivos como displays ubicados dentro de las casas, que pueden monitorear el consumo de energía y no sólo eso, sino que también pueden interactuar con electrodomésticos o cualquier otro sistema eléctrico como bombas de agua o calefacción, con la finalidad de aprovechar mejor la energía. Zigbee goza de un importante respaldo para la gestión energética y para las soluciones de consumo eficiente por parte de la industria de los servicios

públicos; y por parte de los patrocinadores de las redes energéticas inteligentes en varios países.

En general, ZigBee resulta ideal para redes estáticas, escalables y con muchos dispositivos, pocos requisitos de ancho de banda y uso infrecuente, y dónde se requiera una duración muy prolongada de la batería.

2.5 MÓDULOS OPEN HARDWARE ARDUINO

Arduino es una herramienta para hacer que los procesadores puedan sentir y controlar el mundo físico a través de su computadora. Es una plataforma de desarrollo de computación física (physical computing) de código abierto, basada en una placa con un sencillo microcontrolador y un entorno de desarrollo para crear software (programas) para la placa.

Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces, motores y otros actuadores. Los proyectos de Arduino pueden ser autónomos o comunicarse con un programa (software) que se ejecute en la computadora (ej. Flash, Processing, MaxMSP). La placa puedes montarla tú mismo o comprarla ya lista para usar, y el software de desarrollo es abierto y lo puedes descargar gratis.

El lenguaje de programación de Arduino es una implementación de Wiring, una plataforma de computación física parecida, que a su vez se basa en Processing, un entorno de programación multimedia.

Arduino fue elegido para esta tesis debido a los múltiples beneficios que presta, se va a explicar básicamente todo lo que tiene que ver con el uso, los diferentes tipos de módulos, las características, los beneficios, de esta plataforma open hardware, la cual se utiliza para la creación de prototipos basada en software y hardware flexibles y fáciles de usar.

2.5.1 Antecedentes

El proyecto comenzó en Ivrea, Italia, en el 2005 con el fin de crear un dispositivo para estudiantes para el control integrado de proyectos de diseño e interacción, con la finalidad de que fuera más barato que los sistemas de creación de prototipos disponibles en ese entonces. A partir de mayo de 2011, más de 300.000 unida-des de Arduino han sido distribuidas. Los fundadores Massimo Banzi y David Cuartielles nombraron el proyecto como Arduino de Ivrea, un protagonista histórico de la ciudad. En primer lugar "Arduino" es un término masculino italiano, que significa "gran amigo" [31].

2.5.2 ¿Por qué Arduino?

Hay muchos otros microcontroladores y plataformas con microcontroladores disponibles para la computación física. Parallax Basic Stamp, BX-24 de Netmedia, Phidgets, Handyboard del MIT, y muchos otros ofrecen funcionalidades similares. Todas estas herramientas organizan el complicado trabajo de programar un microcontrolador en paquetes fáciles de usar. Arduino, además de simplificar el proceso de trabajar con microcontroladores, ofrece algunas ventajas respecto a otros sistemas a profesores, estudiantes y amateurs:

- **Asequible** - Las placas Arduino son más asequibles comparadas con otras plataformas de microcontroladores. La versión más cara de un módulo de Arduino puede ser montada a mano, e incluso ya montada cuesta bastante conveniente.

- **Multi-Plataforma** - El software de Arduino funciona en los sistemas operativos Windows, Macintosh OSX y Linux. La mayoría de los entornos para microcontroladores están limitados a Windows.

- **Entorno de programación simple y directo** - El entorno de programación de Arduino es fácil de usar para principiantes y lo suficientemente flexible para los usuarios avanzados. Pensando en los profesores, Arduino está basado en el entorno de programación de Procesing con lo que el estudiante que aprenda a programar en este entorno se sentirá familiarizado con el entorno de desarrollo Arduino.

- **Software ampliable y de código abierto** - El software Arduino está publicado bajo una licencia libre y preparado para ser ampliado por programadores experimentados. El lenguaje puede ampliarse a través de librerías de C++, y si se está interesado en profundizar en los detalles técnicos, se puede dar el salto a la programación en el lenguaje AVR C en el que está basado. De igual modo se puede añadir directamente código en AVR C en tus programas si así lo deseas.

- **Hardware ampliable y de Código abierto** - Arduino está basado en los microcontroladores ATMEGA168, ATMEGA328 y ATMEGA1280. Los planos de los módulos están publicados bajo licencia Creative Commons, por lo que

diseñadores de circuitos con experiencia pueden hacer su propia versión del módulo, ampliándolo u optimizándolo. Incluso usuarios relativamente inexpertos pueden construir la versión para placa de desarrollo para entender cómo funciona y ahorrar algo de dinero.

2.5.3 Tipos De Placas Arduino

A continuación se presentan una breve descripción de algunas placas Arduino que se pueden ver en la Figura II.37.

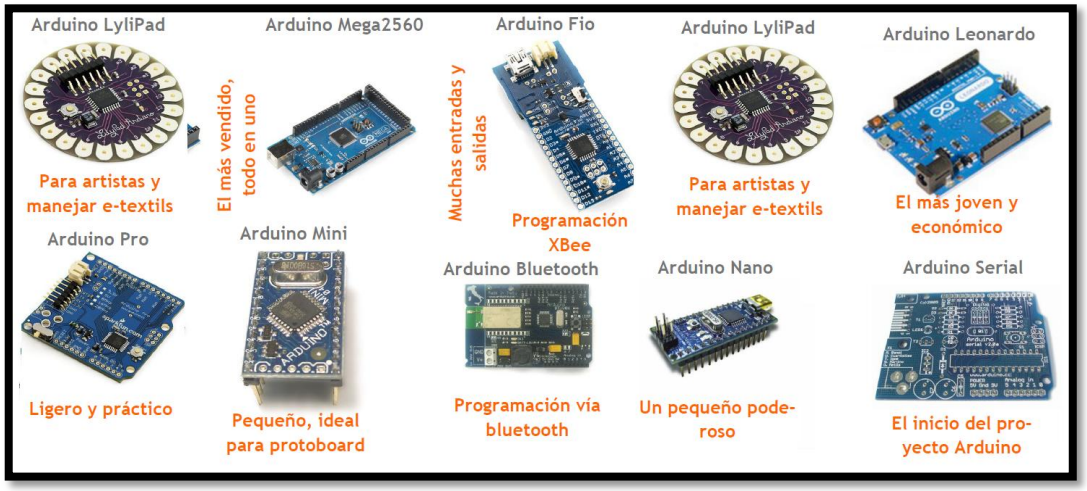

Figura II.37: Aspecto de Arduino UNO

Fuente: http://tienda.tdrobotica.co/download/Libro_kit_Basico

2.5.4 Prestaciones

- ***Arduino Mega*** es con mucha diferencia el más potente y el que más i/o tiene, apto para trabajos ya algo más complejos aunque se tenga que sacrificar un poco el espacio, cuenta con el microcontrolador Atmega1280 con más memoria para el programa, más RAM y más pines que el resto de los modelos .

- ***Arduino Bluetooth*** incorpora un módulo para la transmisión de datos de hasta 100 metros, con esta placa se puede programar sin cables así como también realizar comunicaciones serie con cualquier dispositivo bluetooth.

- ***Arduino Pro*** más robusto y mejor acabado final; incorpora funcionalidades interesantes tales como un conector especial para conectar una batería LiPo y realizar un montaje portátil.

- ***Arduino Nano*** su principal ventaja es que puede ser pinchado directamente sobre una protoboard haciendo muy cómodo el prototipo al igual que el Arduino mini.

- ***Arduino UNO*** es la placa estándar Arduino Diecimila, a pesar de ser el mismo modelo que el duemilanove pero en su versión anterior lo cito porque aún hay algunas tiendas con él en stock; la principal desventaja es que trae el chip atmega168 frente al atmega328 del duemilanove que es más potente aunque perfectamente compatibles respecto a patillaje y programación

- ***Arduino Mini*** versión miniaturizada de la placa Arduino. Mide tan sólo 30x18mm y permite ahorrar espacio en los proyectos que lo requieran. Las funcionalidades son las mismas que Arduino Duemilanove. [31]

2.5.5 Características Generales

La placa que se escogido para este trabajo es Arduino UNO como se mencionó anteriormente se trata de una placa open hardware por lo que su diseño es de libre distribución y fácil utilización, que incluso se puede construir, en la Figura II.38 se observa el aspecto de la placa.

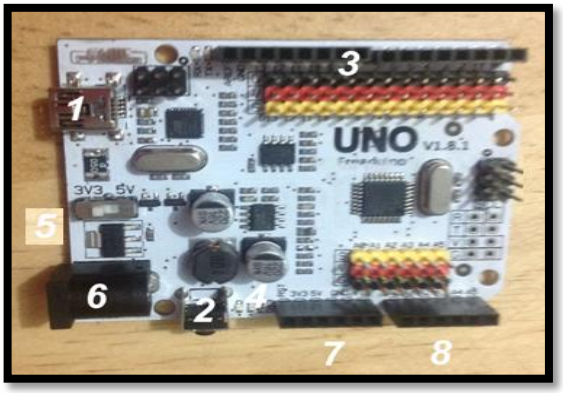

Figura II.38: Aspecto de Arduino UNO

Fuente: Elaboración propia

Las partes principales de un Arduino UNO se numeran en la Figura II.38 y estas son:

1. Conector USB para el cable
2. Pulsador de Reset
3. Pines de E/S digitales y PWM
4. LED verde de placa encendida
5. Regulador de voltaje
6. Conector hembra 2.1mm con centro positivo
7. Pines de voltaje y tierra
8. Entradas análogas

2.5.6 Entorno De Desarrollo

Para programar la placa es necesario descargarse de la página web de Arduino el entorno de desarrollo (IDE). Se dispone de versiones para Windows y para MAC, así como las fuentes para compilarlas en LINUX. En la Figura II.39 se muestra el aspecto del entorno de programación. En el caso de disponer de una placa USB es necesario instalar los drivers FTDI. Estos drivers vienen incluidos en el paquete de Arduino mencionado anteriormente. Existen en la web, versiones para distintos sistemas operativos.

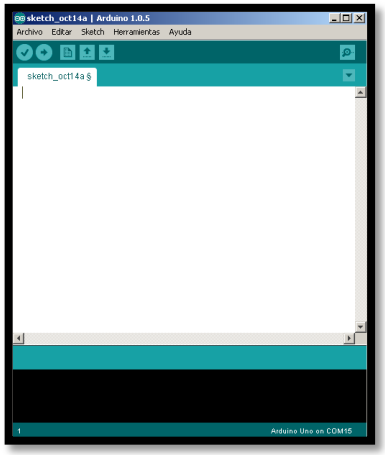

Figura II.39: Entorno de de Desarrollo Arduino
Fuente: Elaboración propia

Lo primero que se debe hacer para comenzar a trabajar con el entorno de desarrollo de Arduino es configurar las comunicaciones entre la placa Arduino y el PC. Para ello se debe abrir en el menú "*Tools"* la opción "*Serial Port"*. En esta opción se debe seleccionar el puerto serie al que está conectada nuestra placa, para posteriormente empezar la programación del código. [32]

2.5.7 Estructura básica de un programa

La estructura básica de programación de Arduino es bastante simple y divide la ejecución en dos partes: setup y loop.

- Setup() constituye la preparación del programa. Se incluye la declaración de variables y se trata de la primera función que se ejecuta en el programa. Esta función se ejecuta una única vez y es empleada para configurar el pinMode (p. ej. Si un determinado pin digital es de entrada o salida) e inicializar la comunicación serie.
- loop() es la ejecución. Además que incluye el código a ser ejecutado continuamente (leyendo las entradas de la placa, salidas, etc.).

En la Figura II.40 se muestra la configuración básica, se observa en este bloque de código cada instrucción acaba con ; y los comentarios se indican con //. Al igual que en C se pueden introducir bloques de comentarios con /* … */.

```
void setup() {
   inMode(pin, OUTPUT);      // Establece 'pin' como salida
}
void loop() {
   digitalWrite(pin, HIGH);  // Activa 'pin'
   delay(1000);              // Pausa un segundo
   digitalWrite(pin, LOW);   // Desactiva 'pin'
   delay(1000);
}
```

Figura II.40: Programa básico Arduino

Fuente: Elaboración propia

2.5.8 Funciones

Una función es un bloque de código identificado por un nombre y que es ejecutado cuando la función es llamada. La declaración de una función incluye en primer lugar el tipo de datos que devuelve la función. Después del tipo de datos se especifica el nombre de la función y los parámetros de la misma.

La siguiente figura se muestra una función es empleada para realizar un retardo en el programa leyendo el valor de un potenciómetro como se muestra en la Figura II.41:

sketch_oct14a §

```
int delayVal() {
int v; // crea una variable temporal 'v'
v = analogRead(pot); // lee el valor del potenciómetro
v /= 4; // convierte los valores 0-1023 a 0-255
return v; // devuelve el valor final de la variable
}
```

Figura II.41: Funciones en Arduino

Fuente: Elaboración propia

2.5.8.1 Funciones Básicas

Se va explicar funciones básicas que ayudan al proceso de datos externos.

- **Función pinMode(pin, mode)**

Función usada en la function setup() para configurar un pin dado para comportarse como INPUT u OUTPUT. Ej. pinMode(pin, OUTPUT); configura el pin número 'pin' como de salida. Los pines de Arduino funcionan por defecto como entradas, de forma que no necesitan declararse explícitamente como entradas empleando pinMode().[32]

- **Función digitalRead(pin)**

Lee el valor desde un pin digital específico. Devuelve un valor HIGH o LOW. El pin puede ser especificado con una variable o una constante (0-13). Ej. V=digitalRead (Pin);

- **Función digitalWrite(pin, value)**

Introduce un nivel alto (HIGH) o bajo (LOW) en el pin digital especificado. De nuevo, el pin puede ser especificado con una variable o una constante 0-13. Ej. digitalWrite(pin, HIGH);

- **Función analogRead(pin)**

Lee el valor desde el pin analógico especificado con una resolución de 10 bits. Esta función solo funciona en los pines analógicos (0-5). El valor resultante es

un entero de 0 a 1023. Los pines analógicos, a diferencia de los digitales no necesitan declararse previamente como INPUT u OUTPUT.

- **Función analogWrite(pin, value)**

Escribe un valor pseudo-analógico usando modulación por ancho de pulso (PWM) en un pin de salida marcado como PWM. Esta función está activa para los pines 3, 5, 6, 9, 10, 11. Ej analogWrite(pin, v); // escribe 'v' en el 'pin' analógico. Puede especificarse un valor de 0 – 255.

Un valor 0 genera 0 V en el pin especificado y 255 genera 5 V. Para valores de 0 a 255, el pin alterna rápidamente entre 0 V y 5 V, cuanto mayor sea el valor, más a menudo el pin se encuentra en HIGH (5 V). Por ejemplo, un valor de 64 será 0 V tres cuartas partes del tiempo y 5 V una cuarta parte. Un valor de 128 será 0 V la mitad del tiempo y 5 V la otra mitad. Un valor de 192 será 0 V una cuarta parte del tiempo y 5 V tres cuartas partes.

2.5.8.2 Funciones de Tiempo y Matemáticas

- **delay(ms).** Realiza una pausa en el programa la cantidad de tiempo en milisegundos especificada en el parámetro (máximo 1000, mínimo 1).
- **millis().** Devuelve la cantidad de milisegundos que lleva la placa Arduino ejecutando el programa actual como un valor long unsigned. Después de 9 horas el contador vuelve a 0.
- **min(x,y). max(x,y).** Devuelve el mínimo y el máximo respectivamente de entre sus parámetros.

2.5.8.3 Funciones Para el Puerto Serie

- **Serial.begin(rate).** Abre un Puerto serie y especifica la velocidad de transmisión. La velocidad típica para comunicación con el puerto de la computadora es de 9600 aunque se pueden soportar otras velocidades.
- **Serial.println(data).** Imprime datos al puerto serie seguido por un retorno de línea automático. Este comando tiene la misma forma que Serial.print() pero este último sin el salto de línea al final. Este comando puede emplearse para realizar la depuración de programas. Para ello puede mandarse mensajes de depuración y valores de variables por el puerto serie. Posteriormente, desde el entorno de programación de Arduino, activando el “Serial Monitor” se puede observar el contenido del puerto serie, y, por lo tanto, los mensajes de depuración. Para observar correctamente el contenido del puerto serie se debe tener en cuenta que el “Serial Monitor” y el puerto serie han de estar configurados a la misma velocidad (Para configurar la velocidad del puerto serie se hará con el comando Serial.begin(rate)).
- **Serial.read().** Lee o captura un byte (un aplicación) desde el puerto serie. Devuelve -1 si no hay ningún carácter en el puerto serie.
- **Serial.available().** Devuelve el número de caracteres disponibles para leer desde el puerto serie.

Como se nota es una programación bastante similar a C, por lo que cualquier programador va a tener la intuición de crear código en Arduino, además con los ejemplos propios de este software se tiene una guía para alguna aplicación, además que internet es la mayor fuente de información en la que también se puedes descargar librerías dedicadas al manejo de dispositivos específicos, en este trabajo se necesitó la librería adicional para el sensor DHT11.

CAPÍTULO III

EVALUACIÓN DE LAS TOPOLOGÍAS WSN

3.1. INTRODUCCIÓN

En este capítulo se evaluará los parámetros más importantes que se debe tener en consideración al momento de implementar una red WSN, la red estará formada por 10 nodos y 1 coordinador basándose en los archivos de configuración instalados adjunto al software NS2, los cuales están organizados en dos topologías físicas con las mismas características. Topología estrella, cada nodo se conecta directamente al nodo coordinador. Topología árbol, cada

nodo se conecta a un nodo enrutador y después al nodo coordinador, los datos son ruteados desde el nodo de menor jerarquía en el árbol hasta el coordinador.

No se evaluó las características de la topología malla ya que esta se usa para áreas que requieren mayor seguridad, redundancia, tolerancia a fallas, y son aplicadas en campos médicos, en donde la vida de las personas depende del óptimo funcionamiento de estas redes, mientras que el propósito de este trabajo de investigación está orientado a la medición de variables ambientales.

Para evaluar se utilizará NS2 [33] que es el simulador de redes en código abierto más extendido en áreas de investigación, siendo aplicado para simular estructuras y protocolos de redes de todo tipo, además de desarrollar nuevos protocolos, algoritmos y comprobar su funcionamiento, así como también comparar distintas características de cada topología.

Para comprobar la hipótesis planteada, se ha empleado el método estadístico de diseño completamente al azar (DCA) [34], el cual determina la topología con mejores características para que la red WSN sea eficiente.

3.2. SIMULADOR NS2

Network Simulator 2 (NS2) es un software con el cual se puede simular eventos discretos y fue ideado para la ayuda a la investigación de redes. NS proporciona soporte para la simulación de multitud de protocolos de las capas de aplicación (http, ftp, cbr, etc.), transporte (TCP, UDP, RTP, SRM), protocolos de enrutamiento unicast, multicast, broadcast, etc., tanto para redes cableadas

como inalámbricas o vía satélite, además de simular topologías complejas con un gran número de generadores de tráfico.

NS comenzó en 1989 como una variante al ya existente REAL Network Simulator y ha evolucionado sustancialmente en los últimos años, habiendo sido objeto de desarrollo por DARPA con ayuda de varias instituciones de investigación en redes, como LBL, Xerox PARC, UCB, USC/ISI,etc.

También contempla mecanismos referentes a la capa de Enlace de Datos en Redes de Área Local (LAN), tales como protocolos MAC (Control de Acceso al Medio) del tipo CSMA/CD (Acceso Múltiple con Detección de Portadora y Detección de Colisiones). Así mismo, incluye diversos algoritmos para la planificación de Colas: FQ (Encolamiento Justo), SFQ (Encolamiento Estocástico Justo), DRR (Deficit Round Robin), FIFO (Primero en Entrar es Primero en Salir), etc. Posee herramientas para graficar como NAM y XGRAPH. La utilidad de NAM es que puede representar gráficamente la red, que previamente se haya construido mediante comandos escritos en un lenguaje de programación llamado Tcl y posteriormente compilados por NS. Así mismo, NAM puede visualizar dinámicamente el desplazamiento de los paquetes de la simulación y que NS ha almacenado en un archivo junto con la propia topología de la red. Estos resultados que NS produce a partir de los archivos que contienen los comandos escritos en lenguaje Tcl o Scripts, lógicamente, dependerán de la topología, protocolos, parámetros, y demás actividad que en ellos se definan. Por otra parte, con las gráficas bidimensionales de XGRAPH se pueden hacer análisis de paquetes recibidos

en la comunicación, paquetes perdidos, ancho de banda y de retardos, aunque necesita un archivo con un formato específico para poder interpretar los datos y así generar tráfico por lo cual existes otra herramienta denominada Tracegraph que con la ayuda de Trace Converter hacer posible manejar los datos del archivo tal como se obtiene del simulador.

3.2.1. Definición

NS (Network Simulator) es básicamente un simulador orientado a objetos, escrito en **C++**, cuya interfaz de usuario se presenta como un intérprete de lenguaje Tcl orientado a objetos o, en otras palabras, de lenguaje Otcl. El simulador soporta una jerarquía de clases escrita en C++, también llamada jerarquía compilada, y otra jerarquía de clases similar a la anterior pero dentro del intérprete que se presenta al programador en Otcl, y que también se conoce como jerarquía interpretada. Estas dos jerarquías están estrechamente relacionadas entre sí, de modo que, desde la perspectiva del usuario, hay una correspondencia uno a uno entre una clase de la jerarquía interpretada y otra de la compilada. La raíz de esta jerarquía es una clase que se llama TclObject. Cuando el usuario crea un objeto simulador desde el intérprete, cosa con la que generalmente se comienza un script, este objeto es creado dentro del intérprete y se crea una estrecha relación con otro objeto idéntico, pero dentro de la jerarquía compilada. La jerarquía de clases interpretada se establece automáticamente a través de métodos definidos dentro de la clase llamada TclClass. Además, se crea otras jerarquías al gusto en los scripts, escritas en

Otcl, y que no estén desdobladas en la jerarquía compilada como se puede ver en la Figura III.42.

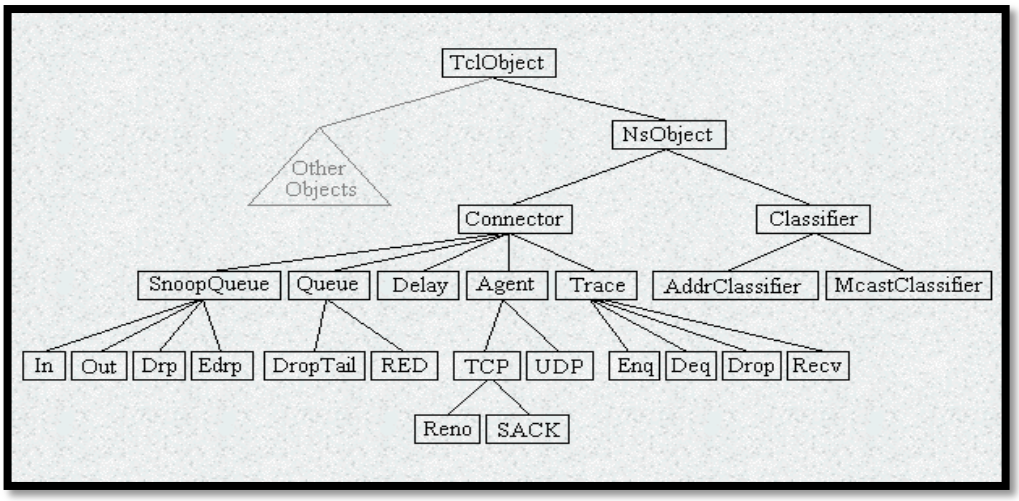

Figura III.42: Jerarquía Otcl

Fuente: http://nile.wpi.edu/NS/

Es muy importante hacer notar que, para trabajar a fondo con NS, usted debe construirse sus propias clases, así como sus propios protocolos y modificar según sus necesidades los ya existentes. Todo ello conlleva hacerlo en el programa fuente del propio simulador NS, escrito en C++, y luego volver a compilar y vincular todos los módulos para crear un nuevo ejecutable de NS.

También se puede ampliar la funcionalidad de NS a través de una segunda posibilidad: mediante Otcl, dentro de los scripts. Dependiendo de qué se necesite hacer, será más apropiado ampliar NS haciéndolo en C++, o bien en Otcl. Por una parte, las simulaciones detalladas de protocolos requieren el uso de C++, al ser un lenguaje de programación que puede manipular eficientemente bytes, paquetes, cabeceras, e implementar algoritmos que manejen grandes cantidades de datos. Para estas tareas la velocidad de ejecución es más importante que el tiempo que pueda invertirse en su

desarrollo. Por otro lado, una gran parte del trabajo de investigación de redes conlleva una modificación continua de parámetros de configuración, la exploración de muchos escenarios de simulación, etc. En estos casos, el tiempo de iteración es muy importante, y en estos casos sólo cobra sentido hacerlo en Otcl.

3.2.2. Estructura básica de simulación

El usuario define mediante un script la pila de protocolos y otros aspectos básicos del tipo de red a simular, y proporciona a ns diversos datos, algunos en forma de archivos acerca del escenario a simular y del tipo y características del tráfico se va a utilizar. Conforme avanza la simulación, se generan un conjunto de datos de salida que se almacenan en un archivo de traza. A partir de las trazas de simulación se pueden utilizar lenguajes como Perl y AWK para filtrar la traza y obtener los índices de prestaciones que se deseen evaluar. Finalmente, herramientas tales como Network Animator (nam) permiten realizar un análisis visual del envío y recepción de paquetes de datos y control a medida que avanza la simulación. La Figura III.43 muestra un esquema del proceso general de simulación.

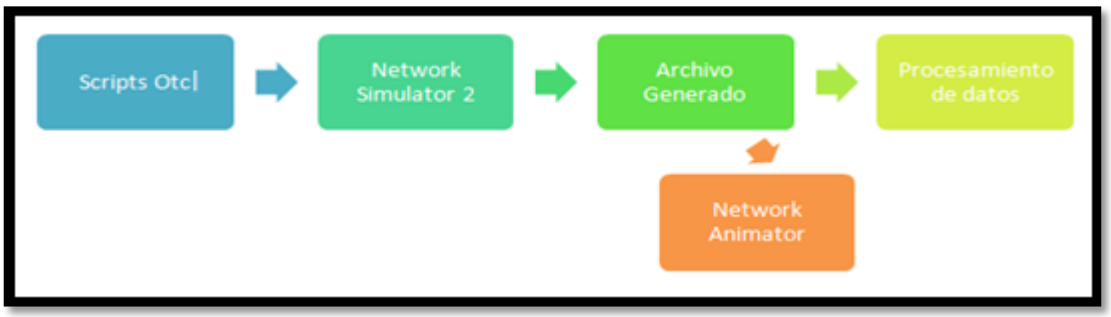

Figura III.43: Estructura básica de una simulación

Fuente: Elaboración Propia

3.2.3. Instalación NS2

Antes de instalar NS2, se debe:

1. Instalar algunos programas esenciales:

- sudo apt-get install tcl8.5-dev tk8.5-dev
- sudo apt-get install build-essential autoconf automake
- sudo apt-get install perl xgraph libxt-dev libx11-dev libxmu-dev

2. Se descarga el archivo fuente NS2 desde la página oficial

- http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.34
- Se descargará un archivo llamado “ns-allinone-2.35.tar.gz“

3. Se desempaqueta ns-allinone-2.35.tar.gz en el directorio “home”.

- tar –zxvf ns-allinone-2.35.tar.gz –C /home/vero

4. Se instala NS2

- cd /home/vero/ns-allinone-2.35
- sudo ./install

5. Se debe poner en el PATH environment. Esto hará que se ejecute itm/tclsh/wish/xgraph.

- /home/vero/ns-allinone-2.35/bin:/home/vero/ns-allinone-2.35/tcl8.5.10/unix:/home/vero/ns-allinone-2.35/tk8.5.10/unix

6. Se modifica .bahrc

- vi /home/vero/.bashrc

Se agrega al final del archivo las siguientes líneas

- export PATH=$PATH:/home/stan/ns-allinone-2.35/bin:/home/vero/ns-allinoe-2.35/tcl8.5.10/unix:/home/vero/ns-allinone-2.35/tk8.5.10/unix
- export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/vero/ns-allinone-2.35/otcl-1.14:/home/vero/ns-allinone-2.35/lib
- export TCL_LIBRARY=$TCL_LIBRARY:/home/vero/ns-allinone-2.35/tcl8.5.10/library

3.2.4. Funcionamiento Básico

Hay dos maneras distintas de ejecutar el simulador, la primera consiste en ejecutar el comando ns seguido de un argumento (en general un archivo tcl a ejecutar) y la segunda ejecutando sólo el comando ns y entrando en el modo interactivo.

- *Con argumentos*:

NS-2 abre y ejecuta el archivo que se pasa como argumento como se muestra en la Figura III.44, al acabar la simulación vuelve a la línea de comandos. El archivo pasado como argumento tiene que tener extensión .tcl y en él deben estar todas las instrucciones necesarias para realizar la simulación.

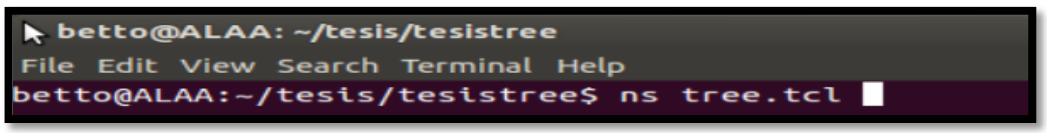

Figura III.44: NS2 con argumentos

Fuente: Elaboración Propia

- *Sin argumentos*

NS-2 entra en modo interactivo y se comporta como cualquier intérprete de comandos, es decir, analiza y ejecuta cada instrucción una a una. En este modo aparece el símbolo % en la consola de comandos, para salir de éste hay que introducir exit. Se verá un ejemplo en la Figura III.45:

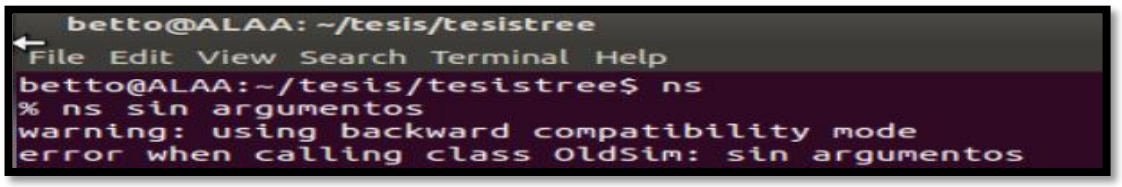

Figura III.45: NS2 sin argumentos

Fuente: Elaboración propia

Las simulaciones realizadas en esta tesis se han creado usando un archivo con la extensión *.tcl* como argumento tal como se ve en la Figura III.46, ya que introducir línea a línea los programas no sería muy eficiente.

Figura III.46: Archivo de configuración (.tcl).

Fuente: Elaboración propia

Además de pasar el archivo como argumento también se pueden pasar más argumentos definiéndolos en el archivo. De esta manera se pueden cambiar algunos parámetros de las simulaciones sin necesidad de cambiar el código del programa.

3.2.5. Complementos

Los resultados obtenidos de las simulaciones se almacenan en archivos de trazas *(.tr).* Es muy importante interpretar los archivos de trazas, ya que de esto depende la evaluación de las topologías en este trabajo de investigación, estos archivos son difíciles de entender, debido a que se presentan como archivos de gran tamaño y multitud de datos sin explicación alguna. Es por ello que se precisa de otros programas para entender e interpretar los resultados obtenidos, algunos son de representación gráfica, como NAM (network animator) [35], xgraph, y otros son programas de filtrado como awk que se utilizará como filtros para recolectar la información necesaria para entender y analizar los resultados.

3.2.5.1. NAM (Network AniMator)

NAM es una de las aplicaciones que se instalan junto al NS-2. Se puede obtener un archivo de salida (.nam) añadiendo unas instrucciones al programa .tcl. Este archivo lo interpreta y ejecuta el NAM presentando una reproducción visual bastante agradable de la simulación realizada. Lo malo es que no se pueden analizar muchos parámetros con este programa.

3.2.5.2. Archivos AWK

AWK es un programa de UNIX que cogiendo como entrada el archivo de trazas (.tr) nos permite operar con los datos o mostrar solo la información deseada para una mejor comprensión. Más adelante se detallan y explican más cosas acerca de los archivos AWK.

3.3. ESTRUCTURA DE LA SIMULACIÓN

Se creó un archivo de simulación para cada topología de los cuales para un mejor resultado, se tomaran datos veinte veces lo que asegura una muestra estadística más valedera como lo indica el método del diseño completamente al azar [36] con la cuales se hará un promedio, en las que en cada una se cambian el tipo de tráfico ya que se genera tráfico UDP aleatorio que produce diferentes muestras para la evaluación, la Figura III.47 muestra la estructura de la simulación.

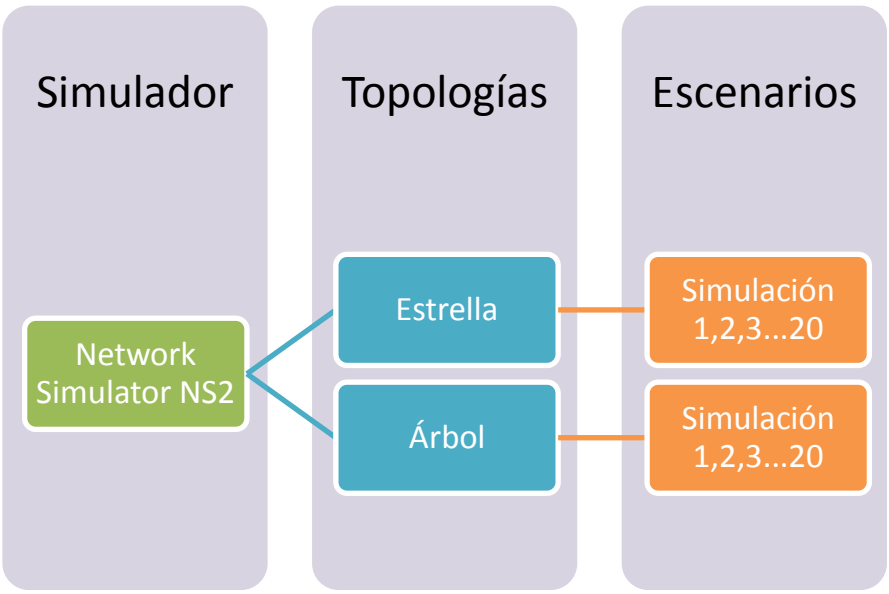

Figura III.47: Estructura de la simulación

Fuente: elaboración Propia

3.3.1. Parámetros de la simulación

La Figura III.48 describe los parámetros constantes y variables en las simulaciones realizadas para cada topología, así se puede observar que las únicas variables relativas al momento de la simulación son el tráfico generado aleatoriamente, el tamaño de los paquetes, mientras que parámetros como el tiempo de simulación, protocolos de enrutamiento, número y cobertura de nodos permanecen constantes. De esta manera se tendrán resultados más confiables evitando caer en muestras que podrían darse por casualidades y excepciones esporádicas.

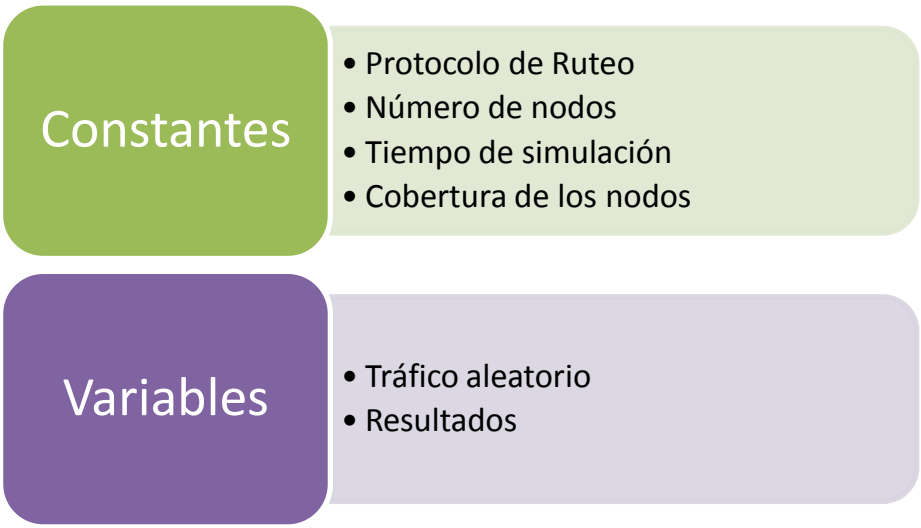

Figura III.48: Parámetros de simulación

Fuente: elaboración Propia

Entre los distintos parámetros de simulación se debe tomar en cuenta los más representativos de la red como:

- Número de nodos

La simulación se realizó con una cantidad tráfico CBR, dado que este tráfico al ser más simple nos permite simular escenarios a mayor escala, haciendo la conexión más rápida ya que este tráfico funciona bajo UDP y cumple con los

requerimientos de envío de datos de una red de variables ambientales que es el objetivo de la investigación.

- Tiempo de simulación

Esta constante permite ver la evolución de la red desde su inicio en el que los nodos buscan establecer conexión con el coordinador, luego en el tiempo de simulación se observa el comportamiento con el envío aleatorio de datos se producen durante la simulación así como se observa el tipo de tráfico generado representado cada uno por un color en particular. Para este efecto se simulo durante 100 segundos, debido a que en la escala de tiempo de ns2 1 segundo en la realidad, representa apenas 2ms dentro de la simulación, dándonos un tiempo suficiente para recoger datos de la red.

- Tipo de trafico

Esta variable es fundamental en una simulación, ya que se utilizó un tráfico lo más cercano posible a la realidad en la que se transmite para tener perspectiva del funcionamiento de la red se utiliza trafico simple para estudiar el desempeño del protocolo de red como es el trafico CBR que funciona sobre UDP, CBR envía bits a una rata constante sin variaciones de ningún tipo, llegando a explotar la conexión y sin confirmaciones de entrega que se asemeja a los datos que se van a recoger en el prototipo a implementar.

- Cobertura de los nodos

Al tener una red WSN, para alcanzar nodos finales y mantener la conectividad, la cobertura es indispensable, por ello se asignó la cobertura que se tienen con

las antenas de los módulos XBEE s1 que es de cien metros para cada nodo al tener en la antena omnidireccional 0,1mW de potencia.

- Tamaño del escenario

El tamaño del escenario se relaciona directamente a la cantidad de nodos que se posea en la red como se puede ver en las Figura III.49-50 los once nodos organizados de distinta manera para cada topología por ello para la simulación de la topología estrella se seleccionó un escenario de trescientos metros por trescientos metros y para la topología árbol se seleccionó un escenario de quinientos metros por quinientos metros.

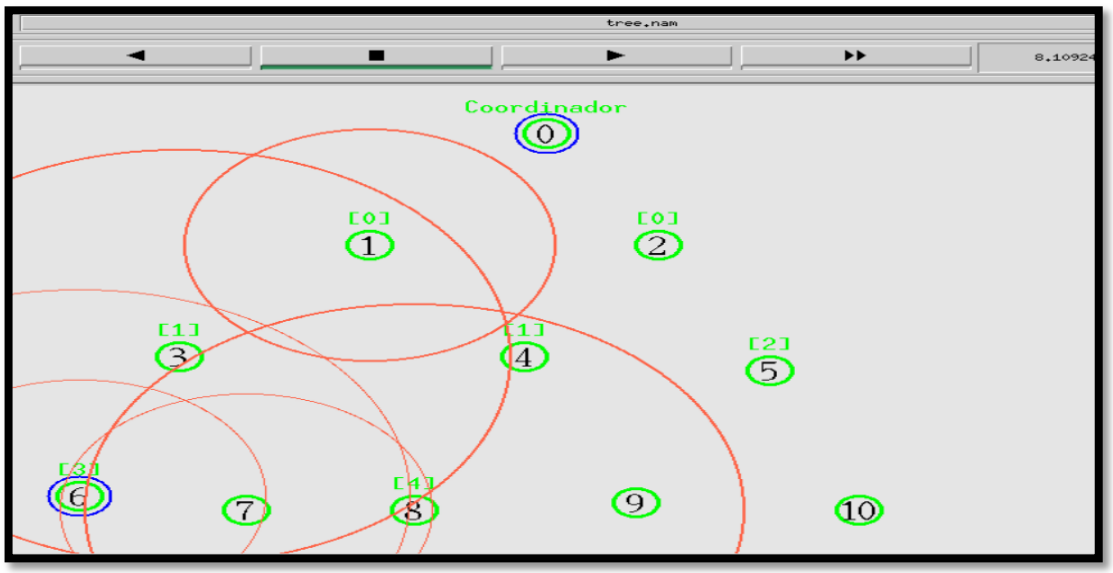

Figura III.49: Escenario de simulación árbol

Fuente: Elaboración propia

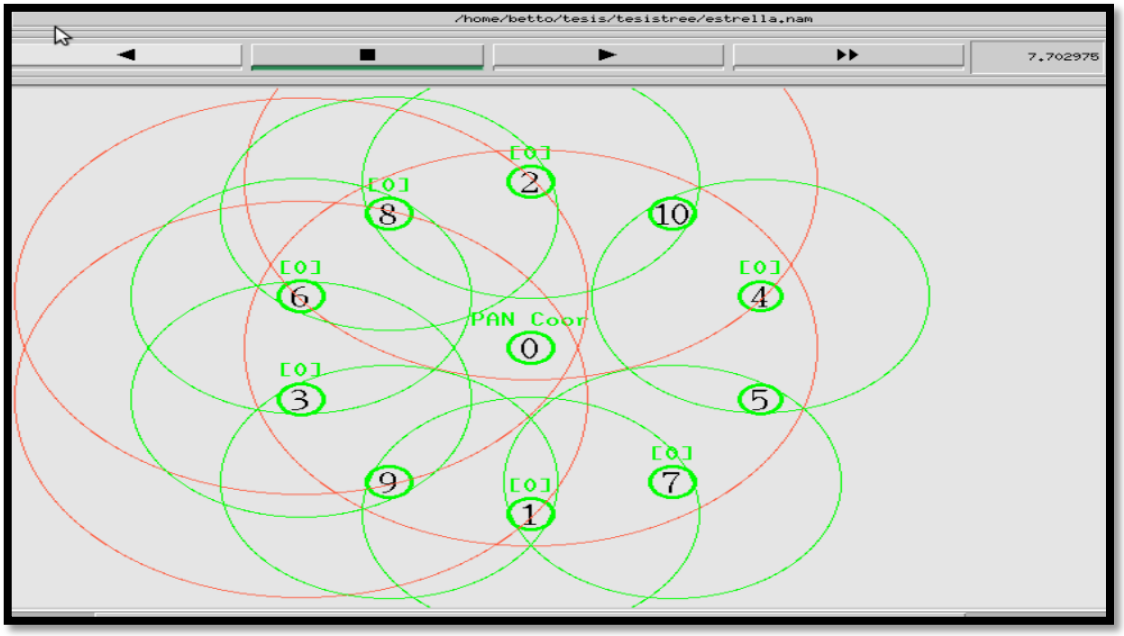

Figura III.50: Escenario de simulación estrella

Fuente: Elaboración propia

3.3.2. Variables de análisis

Para el estudio se toma en cuenta las variables que se consideren más relevantes en análisis de eficiencia, consumo de energía y cobertura de las topologías de una red WSN. Para dicho efecto se analizan los siguientes puntos:

- Cantidad de paquetes enviados, recibidos, desechados y reenviados en la red.
- Retardo de los paquetes que son enviados por los nodos finales al nodo coordinador.
- Throughput de la red WSN.
- Energía consumida por la red en total.
- Cobertura sensada

En la Figura III.51 se observa la información que será analizada en los resultados obtenidos de las simulaciones.

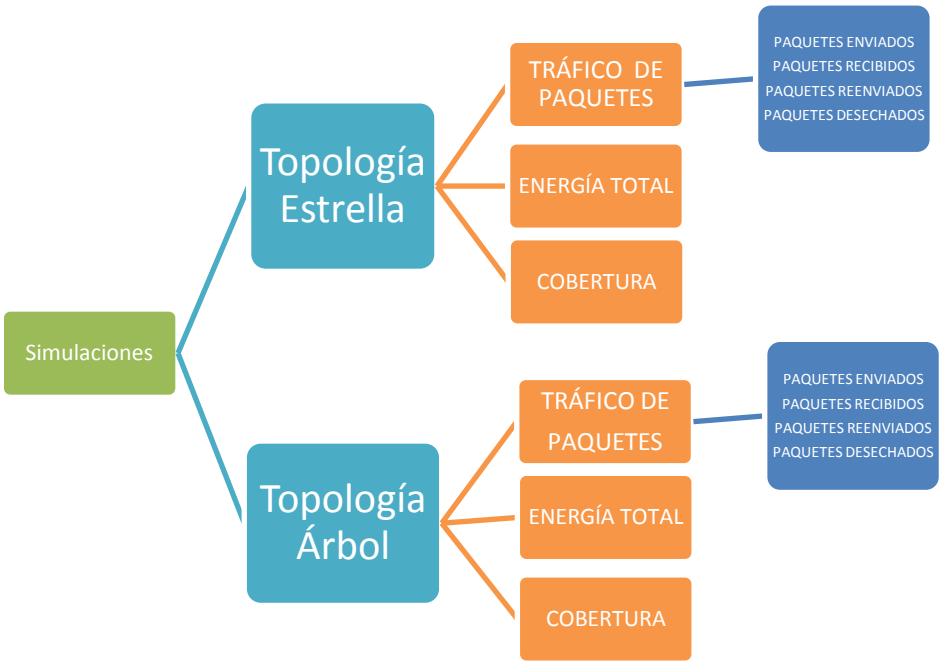

Figura III.51: Variables de simulación

Fuente: Elaboración Propia

Como se observa en la Figura III.54 los resultados se clasifican en tres grupos principales que son:

- Tráfico de paquetes
- Energía consumida
- Cobertura

3.3.2.1. Tráfico de paquetes

Dentro de la simulación se obtuvieron datos generando trafico CBR mediante una conexión UDP. Este tráfico es el que se generan en los nodos finales, los sensores que recolectan datos para intercambiar información y transmitirla hacia el nodo coordinador así se tiene:

- **Paquetes enviados**

Son los paquetes generados por el nodo emisor para crear un enlace de comunicación con el nodo coordinador. Estos paquetes son de tipo CBR sobre una conexión UDP.

- **Paquetes recibidos**

Respecto a los paquetes recibidos estos son aquellos que lograron transmitirse exitosamente entre los nodos finales y el coordinador dependiendo de cada topología, por ejemplo en la topología estrella se transmitirá directo a nodo coordinador, en cambio que en la topología árbol el paquete tendrá que pasar por nodos enrutadores para así llegar al nodo coordinador. Cabe especificar que el número de paquetes recibidos es proporcional al número de paquetes enviados tomando en cuenta la diferencia con los paquetes que se desechan en la red.

- **Paquetes reenviados**

Los paquetes reenviados son aquellos que para alcanzar su destino realizaron saltos a través de la red entre diferentes tipo de nodos que integran una red WSN, de esta manera los nodos enrutadores reenvían paquetes que en este caso se contabilizan como reenviados teniendo la misma estructura que un paquete emitido por el emisor UDP. De este número de paquetes se pueden concluir el número de saltos que utiliza cada conexión para enviar la información ya que mientras más paquetes se reenvíen significa que el número de saltos es mayor.

- **Paquetes desechados**

Los paquetes desechados o caídos en la red son aquellos que por distintas razones como cambios en la topología de la red, destinos inalcanzables u otros no lograron llegar a su destino y en algún punto de la red fueron desechados.

3.3.2.2. Energía Total consumida

Se extrae la información del consumo de energía que se encuentra directamente relacionado con la cobertura de la red por parte de cada nodo para la transmisión de paquetes y un valor nominal para la recepción y modo espera de los nodos, variando el consumo de energía de la red en función de la cantidad de trafico generada e influenciada directamente por el protocolo de ruteo.

El resultado de la energía consumida en la red viene dado por dos parámetros fundamentales que son la cobertura de cada nodo y el número de paquetes transmitidos en la red.

- Energía consumida total

Al hablar de energía se precisa que cada nodo en su inicio cuenta con una capacidad inicial, para la simulación si asigno un valor inicial de energía expresado en Julios de 5760J para cada nodo, de este valor se procede a restar la energía que se va consumiendo en el transcurso de la simulación, de esta manera se puede obtener un valor de la energía consumida expresado en Julios.

3.4. CREACIÓN DEL ESCENARIO

Para la simulación de las topologías de redes de sensores inalámbricos se crean los escenarios que permita obtener las variables para poder analizarlos y compararlos de acuerdo a cada protocolo, para ello se crean múltiples simulaciones con la misma configuración variando las características antes explicadas.

3.4.1. Aspectos Importantes

Al momento de simular, como se muestra en la Figura III.52, se toma en cuenta los elementos que conformaran nuestro sistema entre ellos se tiene la red inalámbrica, el área de cobertura, la generación de tráfico, los niveles de energía y la configuración de los nodos.

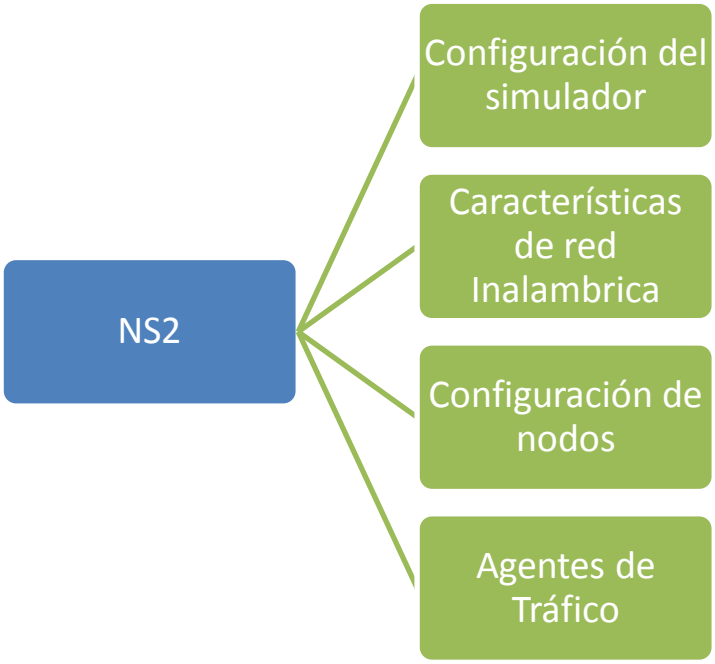

Figura III.52: Variables de simulación

Fuente: Elaboración Propia

3.4.2. Configuración del simulador

- **Inicialización de variables globales de simulación**

Para empezar la simulación lo primero es crear una nueva simulación, para definir variables se utiliza el comando set seguido del nombre de la variable y el valor inicial. Si se trata de definir un objeto de una clase hay que usar el método new. Entonces se creó una nueva simulación ns_ y cualquier procedimiento referido a este objeto se realizará empezando con $ns_.

```
# Crear un nueva instancia de simulación
set ns_      [new Simulator]
```

El siguiente paso consiste en crear un espacio en el que se desarrolle la simulación, esto se consigue adquiriendo una instancia a la clase Topography, también se asigna a esta topología un plano x y en donde se ejecutaran los eventos, como se muestra a continuación:

```
# Configuración de objeto  topography
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
```

Se crea el objeto god encargado de gestionar los enlaces entre nodos, el que se le indica el número de nodos:

```
# Creación God
set god_ [create-god $val(nn)]
```

En esta instrucción permite que se utilice la nueva traza para comunicación wireless, este paso es muy importante e indispensable ya que con la activación

de esta característica podrán realizar filtros .awk para la recolección de la muestras, al ponerlas en un formato adecuado.

```
# Configuración nuevo formato de trazas
$ns_ set WirelessNewTrace_ ON
```

- **Definición archivos de trazas (NS-2 y NAM)**

Se abre un archivo en el cual se genera la traza de los paquetes, es decir el resultado de la simulación. En la siguiente línea se dirige toda la salida de las trazas al archivo que se abrió para el efecto llamado estrella.tr.

```
set tracefd     [open ./estrella.tr w]
$ns_ trace-all $tracefd
```

De igual manera se crea un archivo para guardar las trazas que se utilizaron en la animación de la simulación con el complemento NAM, se especifica que la salida de datos de traza para .nam se guarden en el archive abierto en un plano x y, además en la última línea se indica que se utilice el nuevo formato de trazas.

```
set namtrace     [open ./$val(nam) w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
$ns_ use-newtrace
```

- **Finalización del simulador**

Se detienen la simulación llamando al proceso:

```
$ns_ at $stopTime “stop”
$ns_ at $stopTime “puts \”NS Saliendo...\n\””
$ns_ at $stopTime “$ns_ halt”
```

En el proceso de finalización se cierra los archivos y se termina las trazas de datos, como se muestra a continuación:

```
proc stop {} {
    global ns_ tracefd appTime1 val env
    $ns_ flush-trace
    close $tracefd
}
```

3.4.3. Características de la red inalámbrica

Son variables que se crearon para caracterizar la simulación, donde se asignó valores acorde con la simulación que se desea Así se tiene:

```
# Definición de variables
set val(chan)          Channel/WirelessChannel    ;# Tipo de canal
set val(prop)          Propagation/TwoRayGround   ;# Modelo de radio Propagación
set val(netif)         Phy/WirelessPhy/802_15_4   ;# Interfaz wireless de comuniación
set val(mac)           Mac/802_15_4               ;#Definición de capa MAC
set val(ifq)           Queue/DropTail/PriQueue    ;# Tipo de interfaz queue
set val(ll)            LL                         ;# Tipo de capa de enlace
set val(ant)           Antenna/OmniAntenna        ;# Modelo de antena
set val(ifqlen)        150                        ;# Máximo de paquetes en ifq
set val(nn)            11                         ;# Numero de nodos
set val(rp)            AODV                       ;# Protocolo de enrutamiento
set val(x)        200                             ;#Área de cobertura
set val(y)        200
set val(Mncoverage) 200.0
set val(nam)       estrella.nam
set val(traffic)  cbr                             ;# Trafico UDP/CBR
```

Entre los parámetros de generales se encuentra el modelo de radio propagación para este trabajo se elige el modelo de dos rayos (Two-Ray Ground), esta es una versión mejorada de modelo de espacio libre, no solo toma en cuenta la línea de vista directa del enlace, sino que también el rayo reflejado del suelo que llega de los nodos, lo que proporciona datos más precisos.

3.4.4. Configuración de los nodos

Para la creación de los nodos se especifica una configuración general en la que se asigna los valores de las variables a la configuración del nodo:

```
# Configuración de características del nodo

$ns_ node-config –adhocRouting $val(rp) \
        -llType $val(ll) \
        -macType $val(mac) \
        -ifqType $val(ifq) \
        -ifqLen $val(ifqlen) \
        -antType $val(ant) \
        -propType $val(prop) \
        -phyType $val(netif) \
        -topoInstance $topo \
```

Se especifica los eventos que generaran trazas, estos pueden ser del agente, de ruteo, de acceso al medio y de trazas, al ser una red WSN con variables ambientales no se necesita activar el movimiento en los nodos.

```
        -agentTrace ON \
        -routerTrace ON \
        -macTrace ON \
        -movementTrace OFF \
```

En cuanto a la energía se inicia un modelo del escenario con los parámetros que se desean medir:

```
        -energyModel “EnergyModel” \
        -initialEnergy 5760\
        -rxPower 0.5319 \
        -txPower $pottrans \
        -sleepPower \
        -channel $chan_1_
```

El parámetro $pottrans es la potencia que se va a calcular con los parámetro que tiene la antena de una módulos XBEE s1.

Para realizar el cálculo de energia inicial del nodo wsn, se sabe que la unidad para medir la energía acumulada en una batería es el Julio; sin embargo, para fines prácticos, y dado que el voltaje de la batería del nodo es fijo, se utiliza el [Ah] como unidad de carga, haciendo referencia al tiempo de carga y descarga de la batería.

La equivalencia de energía máxima que se acumula en una batería viene dada por la cantidad de [Ah] de la batería, multiplicada por 3600 y por el voltaje.

En este trabajo la batería tiene 175 mAh (0,175 Ah) de capacidad de carga, y proporciona un voltaje de 9 V, eso quiere decir que puede acumular 15760 Julios:

$$0.175\ Ah * 3600 * 9 = 5760\ J$$

Para la creación de los nodos una vez que ya se han definido y configurado las características de los nodos, éstos pueden ser creados, automáticamente utilizando un bucle “for”, ya que los nodos tendrán las características definidas anteriormente como se usa a continuación.

```
For {set I 0} {$i < $val(nn) } {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0      ;# disable random motion
}
```

Para definir la posición inicial de los nodos hay que situarlos en configurar manualmente donde se desea que estén configurados mediante las coordenadas cartesianas (x, y, z) va a ocupar cada nodo. Así se tiene:

```
$node_(0) set X_ 25
$node_(0) set Y_ 25
$node_(0) set Z_ 0
$node_(1) set X_ 25
$node_(1) set Y_ 14
$node_(1) set Z_ 0
…
```

3.4.5. Creación de los agentes de tráfico

Los nodos por si solos no generan ni reciben tráfico, para hacerlo hay que crear los agentes. Una vez creados estos agentes se deben asociar a distintos nodos. De la implementación de estos agentes dependerán los protocolos de las capas de transporte y superiores. El NS-2 permite implementar una gran cantidad de agentes, de los cuales se eligió crear los siguientes agentes:

```
# Configuración de flujo de tráfico entre nodos

proc cbrtraffic { src dst interval starttime } {
  global ns_ node_

  set udp_($src) [new Agent/UDP]
  eval $ns_ attach-agent \$node_($src) \$udp_($src)

  set null_($dst) [new Agent/Null]
  eval $ns_ attach-agent \$node_($dst) \$null_($dst)

  set cbr_($src) [new Application/Traffic/CBR]
  eval \$cbr_($src) set packetSize_ 32
  eval \$cbr_($src) set interval_ $interval
  eval \$cbr_($src) set random_ 0
  eval \$cbr_($src) set maxpkts_ 10000
  eval \$cbr_($src) attach-agent \$udp_($src)
  eval $ns_ connect \$udp_($src) \$null_($dst)
  $ns_ at $starttime "$cbr_($src) start"
}
```

Se nota que primero se crea el agente UDP, luego el destino Null y después se establece la conexión entre ambos. A continuación se establece el protocolo a nivel de aplicación y se asigna al agente UDP en este caso. Hay una serie de características del tráfico a nivel de aplicación por lo que asigno un tamaño de paquetes de 32 bytes que es el tamaño típico de una trama de variables medio ambientales, un intervalo de aleatorio propio del generador de tráfico y con un máximo de 10000 paquetes recomendado en las simulaciones.

Otra de las características que se configuró en esta simulación fue el flujo de tráfico que se iba a dar para esto se configuraron las siguientes líneas:

```
$val(traffic)traffic 9  0 0.6  $appTime9
  $val(traffic)traffic 10 0 0.6  $appTime10


  $ns_ at $appTime1   "$ns_ trace-annotate \"(at $appTime1)   $val(traffic) tráfico desde nodo 1  al nodo 0\""
  $ns_ at $appTime2   "$ns_ trace-annotate \"(at $appTime2)   $val(traffic) tráfico desde nodo 2  al nodo 0\""
```

3.4.6. Establecer la Cobertura de los nodos

Es importante establecer el radio de cobertura de los nodos. En NS2 se asigna por defecto parámetros en la configuración de los nodos, por esto se va a definir parámetros según las necesidades de este trabajo investigativo. Así para el cálculo de la potencia con la que los nodos deberán transmitir apegándonos a las características reales de la antena del módulo XBEE se realiza las siguientes configuraciones.

Lo primero que hay que decir al respecto de la cobertura es que no existe un parámetro configurable que establezca directamente la cobertura, pero sí se puede configurar la potencia emitida por los nodos, así que se toman en cuenta la fórmula de la potencia recibida por una antena:

$$Pr = \frac{Pt \times Gt \times Gr \times ht^2 \times hr^2}{d^4 \times L}$$

Los parámetros que aparecen en esta fórmula corresponden a:

- *Pt* es la potencia transmitida por la antena transmisora
- *Pr* es la potencia recibida en la antena receptora
- *Gt* es la ganancia de la antena transmisora
- *Gr* es la ganancia de la antena receptora
- *ht* es la altura de la antena transmisora
- *hr* es la altura de la antena receptora
- *d* es la distancia entre ambas antenas
- *L* son las pérdidas

Despejando la potencia transmitida de la fórmula anterior se obtienen:

$$Pt = \frac{Prd^4 \times L}{Gt \times Gr \times ht^2 \times hr^2}$$

Como se muestra la potencia transmitida se puede resumir como una función de la distancia entre antenas, además de otros valores que se comportarán como constantes. De esta forma, sabiendo cual es la cobertura máxima que deben tener los nodos, se ajustó a la potencia máxima que podrán transmitir. Pero todo esto se debe programarlo en el NS-2 como una función, ésta quedará de la forma siguiente:

```
#Se calcula la potencia de transmisión
proc SetPt { coverage } {
set Gt [Antenna/OmniAntenna set Gt_]
set Gr [Antenna/OmniAntenna set Gr_]
set ht [Antenna/OmniAntenna set Z_]
set hr [Antenna/OmniAntenna set Z_]
set RXThresh [Phy/WirelessPhy set RXThresh_]
set d4 [expr pow($coverage,4)]
set Pt [expr ($RXThresh*$d4)/($Gt*$Gr*$ht*$ht*$hr*$hr)]
return $Pt
```

```
}

#Se configura los parámetros básicos de las antenas Omnidireccionales
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 0.002
Antenna/OmniAntenna set Gr_ 0.002
set val(Mncoverage) 200.0
Phy/WirelessPhy set Pt_ [SetPt $val(Mncoverage)] ;# asigna la potencia calculada
Phy/WirelessPhy set CSThresh_ $dist(15m)
Phy/WirelessPhy set RXThresh_ $dist(15m)
Phy/WirelessPhy set CPThresh_ 10
Phy/WirelessPhy set L_ 1.0 ;# Pérdida por Trayectoria = 1.0
Phy/WirelessPhy set freq_ 2.4e+6
Phy/WirelessPhy set L_ 1.0
Phy/WirelessPhy set Pt_ [SetPt $val(Mncoverage)] ;# asigna la potencia transmitida
set pottrans [SetPt $val(Mncoverage)]
```

3.4.7. Archivos de Trazas

El archivo de trazas es el resultado final de la simulación. Éste contiene gran cantidad de información con la se puede calcular la eficiencia, adaptabilidad, tiempos de transmisión, número de ruta, tipo de tráfico, energía consumida, etc. También se ha comprobado que estos archivos de trazas son poco amigables y son difíciles de entender. Así que se va a analizar cómo son estos archivos.

Lo primero que hay que conocer es como se estructura el archivo de trazas del NS-2. Cada fila del archivo contiene un evento diferente, puede ser un paquete enviado, un paquete recibido, un paquete perdido, etc. Dentro de cada fila hay una gran cantidad de campos separados por espacios en blanco, el número de campos puede ser distinto en función del tipo de evento. En general, los primeros campos de cada evento son los mismos y son los últimos los que más difieren. Algo muy importante como se mencionó anteriormente es que NS-2 ofrece distintos formatos de trazas. Por defecto viene un formato de traza más

antiguo con menos información, pero en estas simulaciones se ha usado el nuevo formato de traza. (Figura III.53) explicado en la Tabla III.IV.

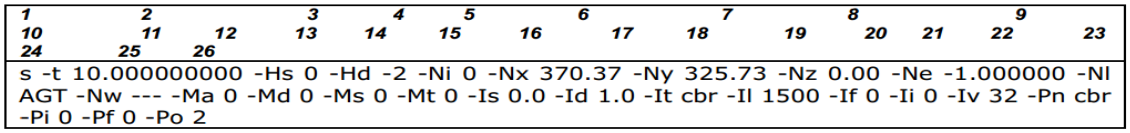

Figura III.53: Campos de archivo .tr

Fuente: NS2 v.2.35

Tabla III.IV: Parámetros de modulación

Campo	Marca	Descripción
1	-	Indica el tipo de evento. Puede tener cinco valores distintos: • s → send, indica que se trata de un paquete enviado • r → receive, es un paquete recibido • f → forward, es un paquete reenviado por un nodo intermedio • d → drop, se trata de un paquete perdido
2	-t	Tiempo en qué ocurre el evento.
-	-H...	Los campos que empiecen con esta marca se refieren a información acerca del siguiente salto.
3	-Hs	Número de nodo dónde acontece el evento.
4	-Hd	Número del siguiente nodo hacia el destino.
-	-N...	Los campos que empiecen con esta marca se refieren a información acerca de las propiedades del nodo.
5	-Ni	Identificador del nodo.
6	-Nx	Coordenada x en la que se encuentra el nodo.
7	-Ny	Coordenada y en la que se encuentra el nodo.
8	-Nz	Coordenada z en la que se encuentra el nodo.
9	-Ne	Nivel de energía del nodo.

10	**-Nl**	El valor de este campo depende de qué capa estamos teniendo en cuenta: • MAC → paquete correspondiente a la capa MAC • AGT → capa de transporte (tcp ó udp) • RTR → paquete enrutado • IFQ → referente a la cola salida del nodo PHY → capa física
11	**-Nw**	Motivo por el cual se descarta el paquete: • --- → el paquete no se descarta • END → final de la simulación • COL → colisión a nivel MAC Existen muchas más posibilidades, no hace falta enumerarlas todas.
-	**-M…**	Los campos que empiecen con esta marca se refieren a información acerca de los paquetes a nivel de capa MAC.
12	**-Ma**	Tiempo esperado en segundos para enviar el paquete a través del canal *wireless*.
13	**-Md**	Dirección MAC del nodo destino.
14	**-Ms**	Dirección MAC del nodo fuente.
15	**-Mt**	Tipo de capa MAC.
-	**-I…**	Los campos que empiecen con esta marca se refieren a información acerca de los paquetes a nivel de IP.
16	**-Is**	Dirección IP del nodo fuente seguido de "." Seguido de el número de puerto del nodo fuente.
17	**-Id**	Dirección IP del nodo destino seguido de "." Seguido del número de puerto del nodo destino.
18	**-It**	Tipo de paquete. Puede ser: cbr, DSR, ack, udp, tcp, ARP…
19	**-Il**	Tamaño del paquete en *bytes*.
20	**-If**	Identificador flowstate.
21	**-Ii**	Identificador de secuencia global del paquete. Identifica el paquete en el fichero de trazas.
22	**-Iv**	Valor del TTL (Time To Live).
-	**-P…**	Los campos que empiecen con esta marca se refieren a información acerca de los paquetes a nivel de "aplicación".
23	**-P**	En este campo se indica el tipo de "aplicación" que puede ser: • arp → Address Resolution Protocol • dsr → Dynamic Source Routing • cbr → Constant Bit Rate • tcp → Transport Control Protocol En función de este campo, el número y tipo de campos siguientes serán distintos. Este ejemplo está hecho a partir de un paquete cbr así que seguiremos con los campos de este tipo de paquete.
24	**-Pi**	Número de secuencia cbr.
25	**-Pf**	Número de veces que el paquete ha sido reenviado.
26	**-Po**	Número óptimo de reenvíos necesarios.

Fuente: http://upcommons.upc.edu/pfc/bitstream/2099.1/8374/1/PFC_Jordi_Chalmeta.pdf

3.5. SIMULACIÓN DE LAS TOPOLOGÍAS

3.5.1 Topología Estrella

En la Figura III.54 se muestra en resultado del archivo .nam para la topología estrella.

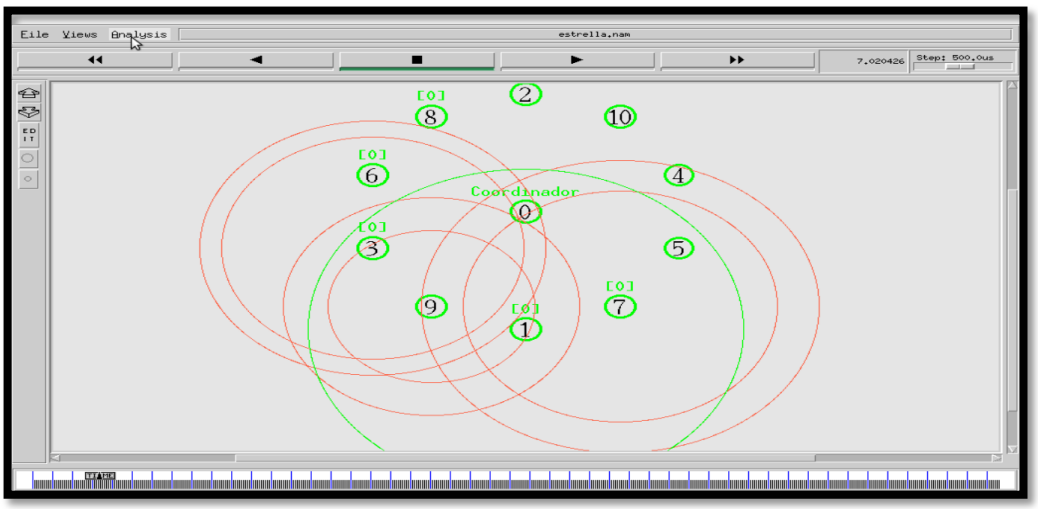

Figura III.54: Topología Estrella

Fuente: Elaboración propia

Para la obtención de datos se utilizó filtro en lenguaje AWK creando los siguientes programas para extraer la información de los archivos de simulación:

- Paquetes recibidos y enviados
- Paquetes caídos y reenviados
- Consumo Total de energía

El código de estos filtros se encontrará en los anexos de este trabajo de investigación.

A continuación se muestran los resultados de una simulación, ya que el procedimiento será repetitivo en las demás simulaciones. Además que los

cálculos que se van a realizar están configurados en los filtros y se generan automáticamente.

- *Tráfico de paquetes.-* los cuales se obtuvieron mediante un filtro en lenguaje AWK el cual se adjunta en los anexos.

Para el análisis de los datos sea decidido utilizar variables estadísticas típicas de en el simulador NS2 y en la redes como son:

- *Packet Delivery Ratio* (Relación de distribución de paquetes): la relación entre el número de paquete de datos entregado al destino. Esto ilustra el nivel de los datos entregados al destino.

$$PDR = \frac{\sum Paquetes\ recividos}{\sum Paquetes\ enviados}$$

- *Throughput* al volumen de trabajo o de información que fluye a través de un sistema, en un determinado tiempo, en este caso tiempo de simulación.

$$Throughput = \frac{números\ de\ bits}{tiempo\ de\ simulación}$$

- *End-to-End Delay (*Retardo de extremo a extremo): el tiempo medio de un paquete de datos para llegar a su destino. También incluye la demora causada por el proceso de descubrimiento de ruta y la cola en la transmisión de paquetes de datos. Sólo los paquetes de datos que se recibieron con éxito a los destinos que cuentan.

$$End - to - End\ Delay = \frac{\sum(tiempo\ de\ llegado\ de\ paquete - tiempo\ de\ envío)}{\sum numero\ de\ coxiones}$$

- *Consumo de Energía:* Se realiza con una simple resta del consumo de energía en cada suceso de la red.

Así la Figura III.55 muestra los datos obtenidos en la primera simulación:

Estadística de las Topología Árbol	
Total paquetes recibidos	1548
Total paquetes enviados	1538
Total paquetes desechados	224
Total paquetes reenviados	541
Packet Delivery Ratio (%)	99
Throughput (Kbps)	0,7690
Average End to End Delay (ms)	0.016213630
Total de energía consumida	46662,835839

Figura III.55: Datos Estadísticos topología estrella

Fuente: Elaboración propia

3.5.2 Topología Árbol

La Figura III.56 muestra el resultado del archivo .nam para la topología estrella.

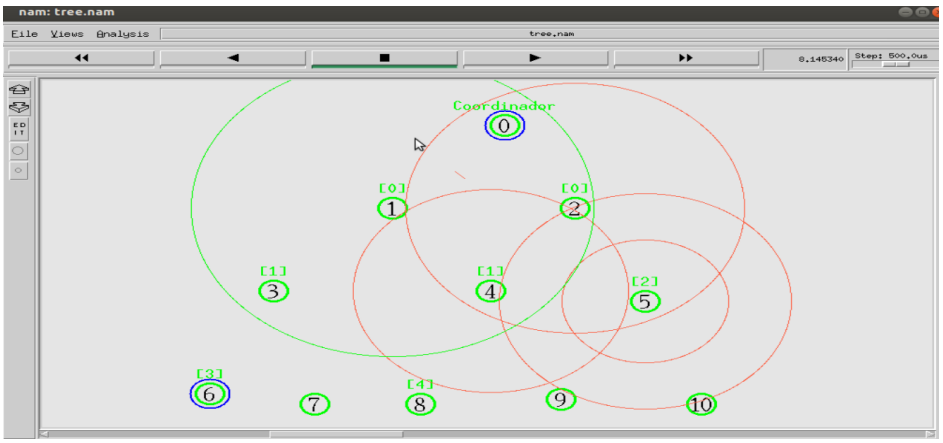

Figura III.56: Topología Árbol

Fuente: Elaboración propia

De la misma manera se utilizó filtros en lenguaje AWK creando los siguientes programas para extraer la información de los archivos de simulación.

A continuación se muestran los resultados de total de paquetes a evaluar de una sola simulación, ya que el procedimiento será repetitivo en las demás simulaciones.

- *PDR*
- *Throughput*
- *End-toEnd Delay*
- *Consumo de Energía.*

Así para la primera simulación se tomaron los siguientes datos, como se muestra en la Figura III.57.

Estadística de las Topología Árbol	
Total paquetes recibidos	2290
Total paquetes enviados	1029
Total paquetes desechados	1893
Total paquetes reenviados	2224
Packet Delivery Ratio (%)	44
Throughput (Kbps)	0,5145
Average End to End Delay (ms)	0,341449245
Total de energía consumida	46645,452310

Figura III.57: Datos Estadísticos topología árbol

Fuente: Elaboración propia

de esta manera en este capítulo se valorará de manera cualitativa a los protocolos para más tarde comprobar la hipótesis mediante el método estadístico de diseño completamente al azar (DCA), el cual se utiliza cuando se obtienen datos de manera aleatoria.

3.6. ANÁLISIS DE RESULTADOS Y COMPROBACIÓN DE HIPÓTESIS

Una vez que se han obtenido datos cuantitativos a través de las simulaciones, se procede a comparar los parámetros de cada topología.

3.6.1 Métodos a Utilizar

Se utilizó el *Método Estadístico De Diseño Completamente Al Azar, se* basa en que el investigador asigna las unidades experimentales a los tratamientos al azar, con la única restricción de que el número de observaciones que se tomarán en cada tratamiento deben ser iguales y mínimo 20. Es el más sencillo y se origina por la asignación aleatoria de tratamientos a un conjunto de unidades experimentales.

Este método sirve para determinar cómo se comportan un conjunto de condiciones experimentales que deben imponerse a una unidad experimental dentro de los confines del diseño seleccionado llamados tratamientos. Como resultado se puede obtener dos opciones que sean iguales o uno de ellos sea estadísticamente superior dejando al experimentador la potestad de elegir la importancia de cada uno de los parámetros a medir y finalmente determinar cuál es la mejor opción de acuerdo a los requerimientos que el investigador necesite.

Para realizar el estudio comparativos con este método se deben seguir estos pasos:

- **Determinar el número de unidades experimentales (n) y numerar.**

Es posible obtener “n” al multiplicar el número de tratamientos (t) por el número de repeticionesaplicación. $n = (t \times r)$

- **Emplear el Modelo estadístico de diseño completamente al azar**

$$Y_{ij} = \mu + \propto_i + \varepsilon_{ij}$$

Dónde:

μ= Efecto de la media general
$\propto_i$= Efecto del i-ésimo tratamiento
ε_{ij}= Error aleatorio asociado a la i-ésimo observación.

- **Análisis de varianza**

Hipótesis Nula Ho ($\mu_1 = \mu_2$): Los tratamientos tienen el mismo efecto sobre la variable en estudio.

Hipótesis Alterna Ha ($\mu_1 \neq \mu_2$): Los tratamientos no tienen el mismo efecto sobre la variable en estudio.

- **Fuentes de variación y grados de libertad.**

Para el análisis de varianza se construye una tabla de análisis de varianza y se completan los datos.

Tabla III.V: Análisis de Varianza

Fuentes de Variación	Grados de Libertad	Suma de Cuadrados	Cuadrados Medios	Fc Calculada	Ft Requerida
Tratamientos	t – 1	$\sum[(Y^2../r\ Y^2..)/(tr)]$	Sctrat/gltotal	Scmedios/Cmerror	
Error	t (r – 1)	Sctotal – Sctrat	Scerror/glerror		
Total	t r -1	$\Sigma\Sigma[\ (Y_{ij} - Y^2..)/(tr)\ ]$			

Fuente: http://reyesestadistica.blogspot.com/2011/07/diseno-de-experimentos-al-completo-azar.html

Grados de Libertad.

- De tratamientos (Gltrat) = t – 1

  Los grados de libertad son uno menos que el número de tratamientos para cada fuente de variación.

- Grados de libertad total (Gltot) = r * t – 1

  Los grados de libertad total están dados por el número de observaciones totales menos 1.

- Grados de libertad del error (Gle) = Gltot – Gltrat

  Los grados de libertar del error están dados por el total de grados de libertad del experimento (Gltot) menos los grados de libertad de los tratamientos (Gltrat).

Suma de cuadrados y Cuadrados Medios

- Cuadrados de Tratamientos

  $Sctrat = \Sigma(Y_i)^2 / r$

  Donde “Yi” totales de tratamientos y “r” el número de repeticiones en cada tratamiento.

- Suma de Cuadrados Totales (Sctot)
- Suma de cuadrados del error (Sce)

  Sce = Sctrat – Sctot

- Cuadrados Medios de Tratamientos (Cmtrat)

  Los cuadrados medios de los tratamientos están dados por la suma de cuadrados del error (Cce) dividido entre el número de repeticiones aplicación.

- Cuadrados medios del error (Cme)

  Los cuadrados medios del error están dados por la suma de cuadrados del error (Sce) dividido entre del error.

Cme = Sce/Gle

F Calculada

La Fc se calcula dividiendo los cuadrados medios de los tratamientos entre el cuadrado medio del error.

Fc = Cmtrat/Cme

Cmtrat = Sce /r

F Requerida

La Ft se encuentra en una tabla diseñada y elaborada para el efecto. La forma de encontrar el valor es dependiendo del nivel de significancia.

El valor de significancia en este estudio es de 0,05, ya que este análisis se realizó con un 95% de nivel de confianza.

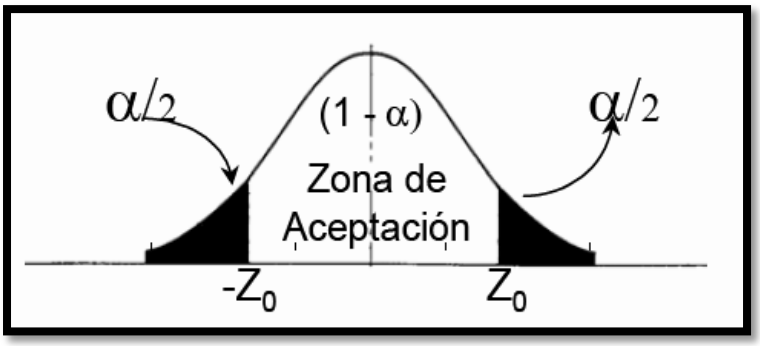

Figura III.58: Nivel de Significancia (95%)

Fuente: http://www.bioestadistica.uma.es/libro/node108.htm

- **Crear la Tabla ANDEVA Ó ANOVA y finalmente interpretar los datos.**

La tabla anova permite rechazar o no la hipótesis Ho mediante el uso del valor "P" que es el valor de la probabilidad del tratamiento.

Donde:

$$Si\ P < 0.05\ Se\ rechaza\ Ho$$
$$Si\ P > 0.05\ No\ se\ rechaza\ Ho$$

Población y Muestra

Población

En este caso la población serán las topologías estrella y árbol.

Muestra

Para la elección de la mejor topología física de una WSN la muestra estuvo constituida por los criterios de evaluación típicos de una red Ad-Hoc, que son los siguientes:

- Desempeño del envío de paquetes para esto se considerará el parámetro *Packet Delivery Ratio* el cual es la relación entre paquetes de datos enviados y entregados al destino, aquí se toma en cuenta los paquetes enviados, recibidos, reenviados y desechados.
- El *Througput* se define como la cantidad de datos por unidad de tiempo que se entregan, a través del medio en un nodo de la red.
- El End-to-End Delay es el tiempo en el que un paquete de datos tarda para llegar a su destino, incluye la demora causada por el proceso de descubrimiento de ruta y la cola en la transmisión de paquetes.
- La energía consumida por toda la red en el tiempo de simulación.
- El área de cobertura que es el espacio físico que se va a sensar.

3.6.2 Estudio Comparativo

Para determinar cuál es la mejor topología se utiliza el método DCA con un nivel de confianza del 95%, además se utiliza el software Minitab que es

diseñado para cálculos estadísticos y permite ver los resultados de forma gráfica.

Se realizó 20 tomas de muestras de cada parámetro con ambas topologías. Donde cada topología representa un tratamiento y cada toma de muestras es una repetición, por tanto para calcular n:

$$n = t * r \rightarrow n = 2 \times 20 \rightarrow n = 40$$

A continuación se puede observar la tabla de las muestras tomadas completamente al azar de las dos topologías y posteriormente se ve la tabla con su respectivas medias generales.

Tabla III.VI: Muestreo de Datos

	N° Muestra	Packet Delivery Ratio (%)	Throughput of the network (Kbps)	Average end to end Delay (ms)	Energía Consumida (J)
TOPOLOGÍA ESTRELLA	1	99	0,7690	0,01621453	46662,835839
	2	98	0,6895	0,01521363	46662,416622
	3	97	0,7641	0,01676364	46661,132841
	4	99	0,6552	0,01491389	46660,536797
	5	97	0,7445	0,01841263	46660,735789
	6	95	0,7690	0,01629461	46662,945839
	7	98	0,7658	0,01631363	46661,835765
	8	99	0,7379	0,00992715	46664,835635
	9	98	0,7289	0,01521663	46661,945887
	10	97	0,7689	0,01621963	46663,173859
	11	98	0,7695	0,01622463	46661,835765
	12	99	0,7695	0,01621363	46664,835635
	13	96	0,7685	0,01621353	46661,945887
	14	98	0,7673	0,02002715	46663,173859

	15	96	0,7694	0,01821263	46662,835839
	16	99	0,7721	0,01629461	46659,416622
	17	99	0,7586	0,01631363	46662,945839
	18	98	0,7684	0,01731353	46661,835765
	19	99	0,7691	0,01521663	46659,835765
	20	99	0,7789	0,01631363	46662,416622
TOPOLOGÍA ÁRBOL	1	44	0,5145	0,341449245	46645,452310
	2	43	0,5132	0,34146909	46650,294276
	3	45	0,5167	0,342449234	46649,761417
	4	44	0,5234	0,341449968	46643,277881
	5	44	0,5147	0,350029245	46645,307219
	6	44	0,5119	0,341049245	46643,245839
	7	46	0,5137	0,348002198	46644,835765
	8	44	0,5149	0,338532198	46643,835635
	9	44	0,5142	0,341450191	46645,452310
	10	44	0,5139	0,349234185	46640,294245
	11	44	0,5151	0,341449245	46645,761324
	12	42	0,5142	0,342342456	46646,272425
	13	44	0,5138	0,354356779	46644,307843
	14	43	0,5140	0,331434252	46643,079859
	15	45	0,5148	0,329032412	46649,027439
	16	45	0,5149	0,341449245	46644,425622
	17	44	0,5137	0,334649212	46642,542339
	18	44	0,5145	0,341449245	46648,135765
	19	43	0,5145	0,338532198	46647,277881
	20	43	0,5143	0,341465245	46649,080905

Fuente: Software NS2

Tabla III.VII: Medias de Datos

	Packet Delivery Ratio (%)	Throughput of the network (Kbps)	Average end to end Delay (ms)	Energía Consumida (J)
Promedio de topología estrella	97,9	0,7542	0,0162	46662,1736
Promedio de topología árbol	43,95	0,5147	0,3416	46645,5834

Fuente: Elaboración propia

Una vez obtenida la media general de cada parámetro, se procede a realizar el análisis de la varianza e interpretación de resultados.

Packet Delivery Ratio (%)

Planteamiento de hipótesis

H0: La media de Packet Delivery Ratio en la topología estrella es igual a la media de Packet Delivery Ratio en la topología árbol.

$$H_0: \mu_1 = \mu_2$$

H1: La media de Packet Delivery Ratio en la topología estrella no es igual a la media de Packet Delivery Ratio en la topología árbol.

$$H_1: \mu_1 \neq \mu_2$$

Calculo de varianza

Tabla III.VIII: Varianza para Packet Delivery Ratio

Topologías	*Muestras*	*Suma*	*Promedio*	*Varianza*
ESTRELLA	20	1958	97,9	1,46315789
ARBOL	20	879	43,95	0,78684211

Fuente: Elaboración propia

Análisis de Resultados (ANOVA)

Tabla III.IX: ANOVA de Packet Delivery Ratio

Origen de las variaciones	*Suma de cuadrados*	*Grados de libertad*	*Promedio de los cuadrados*	*F*	*Probabilidad*	*Valor crítico para F*
Entre grupos	29106,025	1	29106,025	25872,0222	1,85992E-55	4,098171731
Dentro de grupos	42,75	38	1,125			
Total	29148,775	39				

Fuente: Elaboración propia

Gráfica

Mediante el software Mintab, se puede observar de manera gráfica la variación que existe entre los datos de Packet Delivery Ratio recogidos en cada topología.

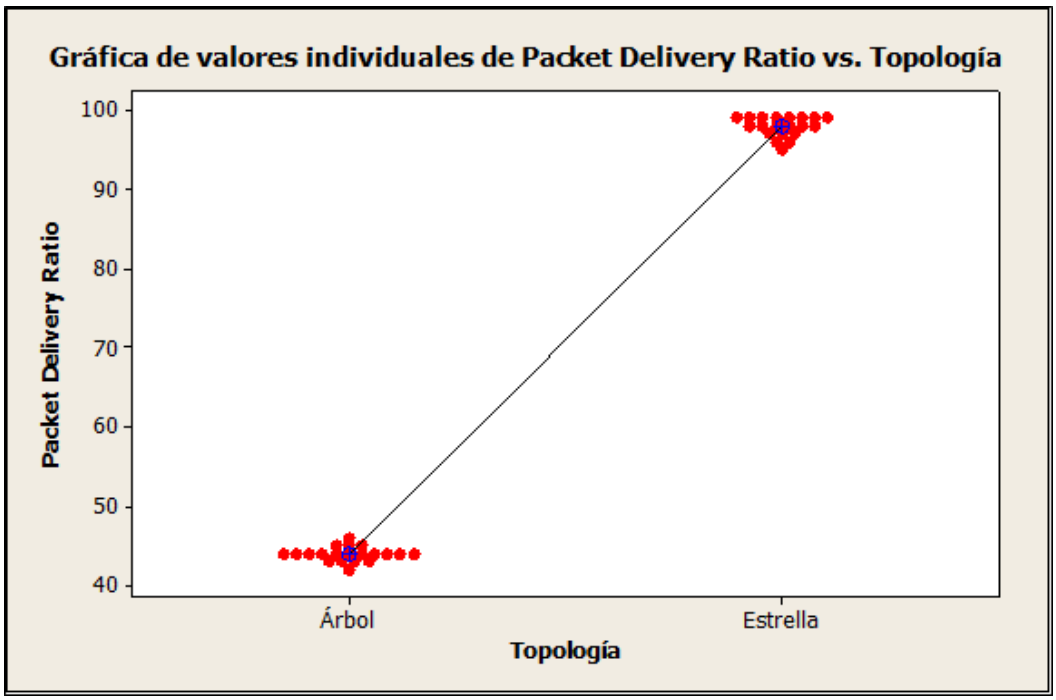

Figura III.59: Packet Delivery Ratio vs Topología

Fuente: Elaboración propia

Interpretación

Mediante el análisis de varianza, al tener un valor de probabilidad p = 1,85992E-55 el cual es menor que el nivel de significancia 0,05 se rechaza la hipótesis nula y se decidio que la topología estrella tiene un mejor efecto de Packet Delivery Ratio.

- **Throughput (Kbps)**

Planteamiento de hipótesis

H0: La media del Throughput en la topología estrella es igual a la media del Throughput en la topología árbol.

$$H_0: \mu_1 = \mu_2$$

H1: La media del Throughput en la topología estrella no es igual a la media del Throughput en la topología árbol.

$$H_1: \mu_1 \neq \mu_2$$

Calculo de varianza

Tabla III.X: Varianza para Throughput

Topologías	*Muestras*	*Suma*	*Promedio*	*Varianza*
ESTRELLA	20	15,0841	0,754205	0,00096434
ARBOL	20	10,2949	0,514745	4,9847E-06

Fuente: Elaboración propia

Análisis de Resultados (ANOVA)

Tabla III.XI: ANOVA de Througput

Origen de las variaciones	*Suma de cuadrados*	*Grados de libertad*	*Promedio de los cuadrados*	*F*	*Probabilidad*	*Valor crítico para F*
Entre grupos	0,57341092	1	0,57341092	1183,10974	3,0427E-30	4,09817166
Dentro de grupos	0,01841724	38	0,00048466			
Total	0,5918281	39				

Fuente: Elaboración propia

Gráfica

Mediante el software Mintab, se puede observar de manera gráfica la variación que existe entre los datos de Throughput recogidos en cada topología.

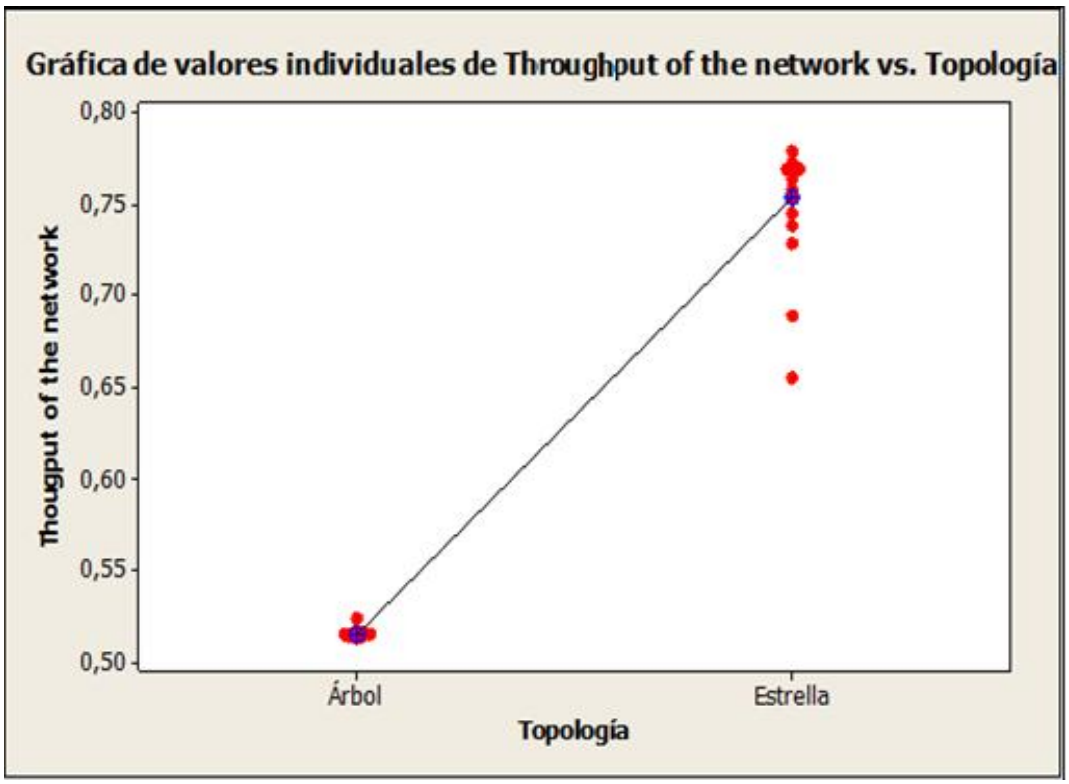

Figura III.60: Throughput vs Topología

Fuente: Elaboración propia

Interpretación

Mediante el análisis de varianza, al tener un valor de probabilidad p = 3,0427E-30 el cual es menor que el nivel de significancia 0,05 se rechaza la hipótesis nula y se decidió que la topología estrella tiene un mejor un throughput.

- **End-to-End Delay (ms)**

Planteamiento de hipótesis

H0: La media del End to End Dalay en la topología estrella es igual a la media del End to End Delay en la topología árbol.

$$H_0: \mu_1 = \mu_2$$

H1: La media del End to End Delay en la topología estrella no es igual a la media del End to End Delay en la topología árbol.

$$H_1: \mu_1 \neq \mu_2$$

Calculo de varianza

Tabla III.XII: Varianza para End to End Delay

Topologías	*Muestras*	*Suma*	*Promedio*	*Varianza*
ESTRELLA	20	0,323833574	0,016191679	3,63777E-06
ARBOL	20	6,831275088	0,341563754	3,49053E-05

Fuente: Elaboración propia

Análisis de Resultados (ANOVA)

Tabla III.XIII: ANOVA de End to End Delay

Origen de las variaciones	*Suma de cuadrados*	*Grados de libertad*	*Promedio de los cuadrados*	*F*	*Probabilidad*	*Valor crítico para F*
Entre grupos	1,058669876	1	1,058669876	54934,42582	1,1547E-61	4,098171731
Dentro de grupos	0,000732318	38	1,92715E-05			
Total	1,059402194	39				

Fuente: Elaboración propia

Gráfica

Mediante el software Mintab, se puede observar de manera gráfica la variación que existe entre los datos de Average End to End Delay recogidos en cada topología.

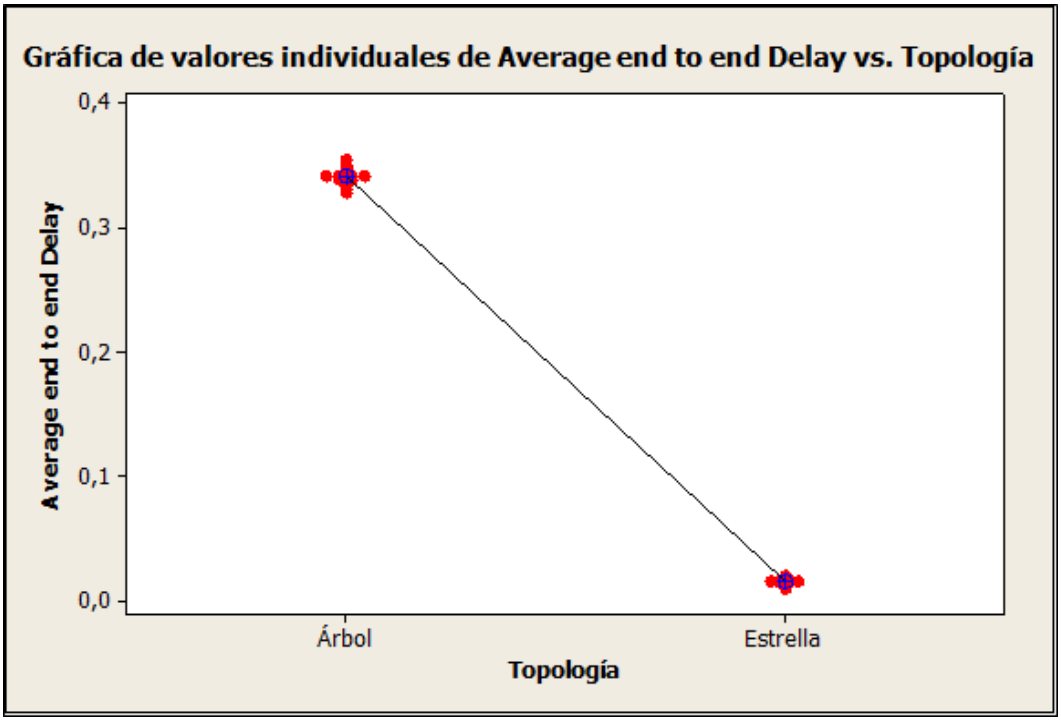

Figura III.61: Throughput vs Topología

Fuente: Elaboración propia

Interpretación

Mediante el análisis de varianza, al tener un valor de probabilidad p = 1,1547E-61el cual es menor que el nivel de significancia 0,05 se rechaza la hipótesis nula y se decidió que la topología estrella tiene un menor End to End Delay.

- **Energía Consumida por toda la Red (Julios)**

Planteamiento de hipótesis

H0: La media del consumo de energía en toda la red en la topología estrella es igual a la media del consumo de energía en toda la red en la topología árbol.

$$H_0: \mu_1 = \mu_2$$

H1: La media del consumo de energía en toda la red en la topología estrella no es igual a la media del consumo de energía en toda la red en la topología árbol.

$$H_1: \mu_1 \neq \mu_2$$

Calculo de varianza

Tabla III.XIV: Varianza para End to End Delay

Topologías	*Muestras*	*Suma*	*Promedio*	*Varianza*
ESTRELLA	20	933243,472	46662,1736	1,98441834
ARBOL	20	932911,668	46645,5834	7,0840367

Fuente: Elaboración propia

Análisis de Resultados (ANOVA)

Tabla III.XV: ANOVA de End to End Delay

Origen de las variaciones	*Suma de cuadrados*	*Grados de libertad*	*Promedio de los cuadrados*	*F*	*Probabilidad*	*Valor crítico para F*
Entre grupos	2752,35024	1	2752,35024	607,016349	5,6983E-25	4,09817173
Dentro de grupos	172,300646	38	4,53422752			
Total	2924,65088	39				

Fuente: Elaboración propia

Gráfica

Mediante el software Mintab, se puede observar de manera gráfica la variación que existe entre los datos de Consumo de energía consumida recogidos en cada topología.

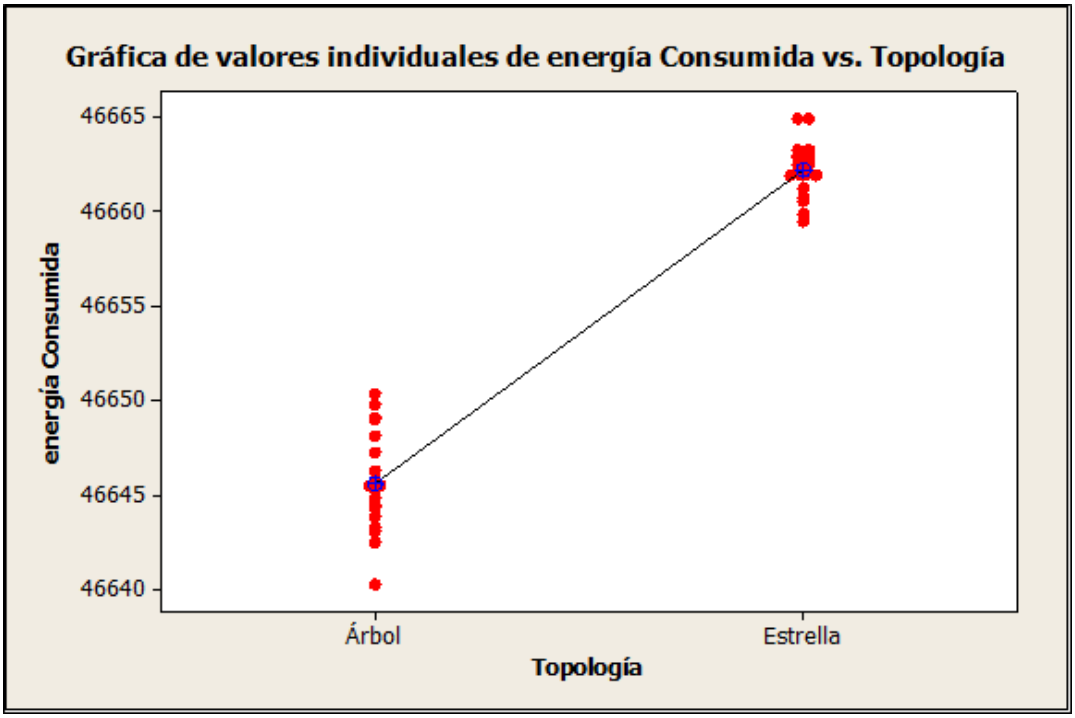

Figura III.62: Throughput vs Topología

Fuente: Elaboración propia

Interpretación

Mediante el análisis de varianza, al tener un valor de probabilidad p = 5,6983E-25 el cual es menor que el nivel de significancia 0,05 se rechaza la hipótesis nula y se decidió que la topología árbol tiene un menor consumo de energía.

- **Área de Cobertura**

Para el análisis del área de cobertura se debe tener algunas consideraciones por lo que no se aplica el mismo método de los otros parámetros, esto se lo realizó en función al área de cobertura de sensado.

El área de cobertura total no se analiza ya que en la topología estrella se ve limitado a un radio de entre 30 a 7000 m dependiendo del radio de comunicación que se utilice mientras que en la topología árbor el área dependerá de la configuración que realice el usuario.

Área de cobertura de sensores

Es el área en que se puede tomar datos de los sensores, lo que quiere decir que únicamente dependerá del tipo de sensores con los que se trabaje ya que en su hoja de datos (datasheet) se indica hasta qué distancia del nodo puede extenderse antes que se atenúe la señal de información.

En los escenario planteados se decidió trabajar con el sensor DHT11 donde el datasheet indica que se puede hacer una conexión cableada de hasta 20 metros de la unidad de procesamiento (ver anexos), donde se obtiene el dato del radio de cobertura de cada nodo, lo que se aplicará para cada caso.

- Topología Estrella

El escenario de la topología estrella consta de 1 nodo coordinador y 10 nodos finales, donde se va a calcular el área de cobertura de sensado como se muestra en la Figura III.63

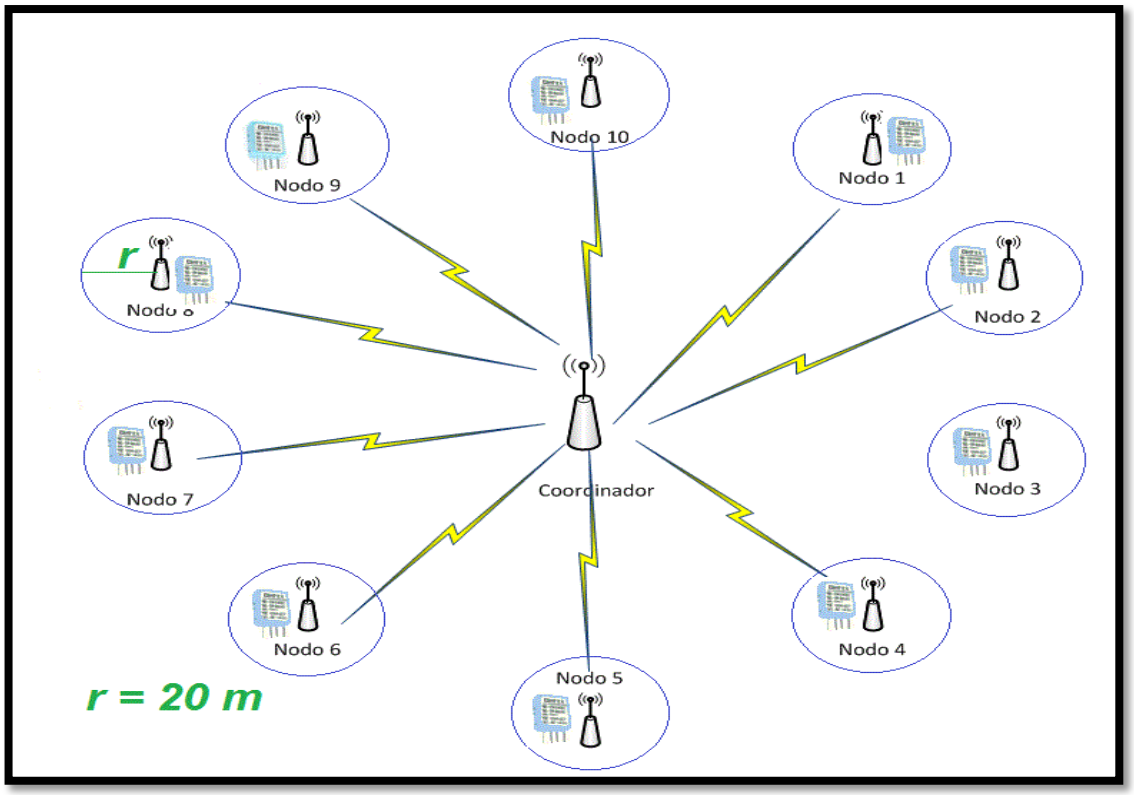

Figura III.63: Área de cobertura de los sensores – Topología Estrella.

Fuente: Elaboración propia

En los nodos se obtienen circunferencias con un radio de veinte metros. Para calcular el área se realiza las siguientes operaciones:

$$Área_{\ sensores/nodo} = \pi r^2$$

$$Área_{\ sensores/nodo} = \pi \times (20)^2\ [m]$$

$$Área_{\ sensores/nodo} = 1\ 256{,}64\ [m^2]$$

$$Área\ cobertura\ _{sensores\ TOTAL} = Área\ \frac{sensores}{nodo} \times 10\ nodos\ finales$$

$$Área\ cobertura\ _{sensores\ TOTAL} = 12566{,}4\ [m^2]$$

- Topología Árbol

El escenario de la topología estrella consta de 1 nodo coordinador, 5 nodos enrutadores y 5 nodos finales, donde se va a calcular el área de cobertura según las circunstancias antes definidas como se muestra en la Figura III.64

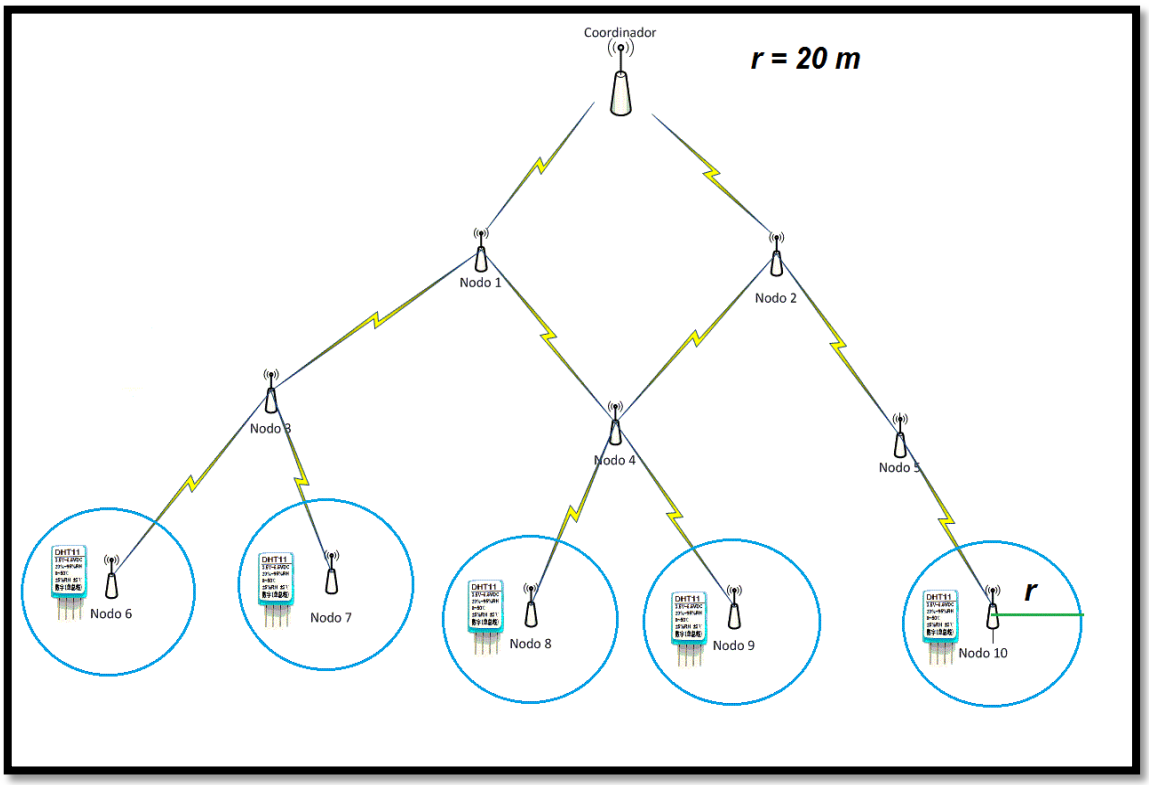

Figura III.64: Área de cobertura de los sensores-Topologías Árbol.

Fuente: Elaboración propia

En los nodos se obtienen circunferencias con un radio aplicación de veinte metros, en este escenario solamente existen 5 nodos finales que van a tomar medidas. Para calcular el área se realiza las siguientes operaciones:

$$Área_{\ sensores/nodo} = \pi r^2$$

$$Área_{\ \ sensores/nodo} = \pi \times (20)^2\ [m]$$

$$Área_{\ sensores/nodo} = 1\ 256{,}64\ [m^2]$$

$$Área\ cobertura_{\ sensores\ TOTAL} = Área_{\ \frac{sensores}{nodo}} \times 5\ nodos\ finales$$

$$Área\ cobertura_{\ sensores\ TOTAL} = 6283{,}19\ [m^2]$$

Interpretación del área de cobertura de las topologías.

En la Tabla III.XVI se observar el área de cobertura de medición para cada topología y con estos resultados se puede interpretar lo siguiente.

Tabla III.XVI: Área de Cobertura

Topología	Área de Sensado $[m^2]$
Estrella	12566,4
Árbol	6283,19

Fuente: Elaboración Propia

- La topología estrella tiene una mayor área de medición de datos mientras que la topología árbol toma datos en la mitad área que la topología estrella.
- Tomando que el estudio está realizado para aplicarse en la toma de datos de variables ambientales se necesita provocar que todos los nodos de la red tomen datos para un mejor análisis lo que se logra con la topología estrella ya que solo necesita un dispositivo para clasificar, almacenar e interpretar la información, consiguiendo una mayor extensión de área para la toma de datos.
- El escenario con topología árbol para este tipo de aplicaciones desperdicia recursos ya que los nodos enrutadores no toman ningún tipo

de medición y únicamente pasan la información de los nodos finales al nodo coordinador.

Por tanto la topología estrella tiene un mayor área de cobertura de sensado.

3.6.3 Comprobación de la Hipótesis

Hipótesis de la Investigación

El estudio del desempeño de una WSN mediante métricas de evaluación (pruebas de envío de paquetes, de consumo de energía y de cobertura) permitirá determinar la topología más adecuada para la implementación de una red de sensores inalámbricos eficiente.

Proceso

Para comprobar la hipótesis se utiliza la Tabla III.XVII para calificar a los parámetros evaluados de acuerdo a los resultados obtenidos en el estudio comparativo.

Tabla III.XVII: Calificación

Valor	Significado
✓	Eficiente

Fuente: Elaboración Propia

Tabla III.XVIII: Resultados

PARÁMETROS	ESTRELLA	ÁRBOL
Packet Delivery Ratio (%)	✓ 97,9	43,95
Throughput (Kbps)	✓ 0,7542	0,5147
Average End to End Delay (ms)	✓ 0,0162	0,3416
Consumo de Energía (J)	46662,1736	✓ 46645,5834
Área de Cobertura de Sensado (m^2)	✓ 12566,4	6283,2

Fuente: Elaboración Propia

Al analizar los resultados obtenidos de cada topología como se muestra en la tabla III.XVII, se pudo comprobar que el estudio del desempeño de una WSN mediante métricas de evaluación (pruebas de envío de paquetes, de consumo de energía y de cobertura) permitió determinar a la topología estrella como la más adecuada para la implementación de una red inalámbrica de sensores eficiente.

CAPÍTULO IV

DISEÑO E IMPLEMENTACIÓN DEL PROTOTIPO WSN

4.1 INTRODUCCIÓN

En este capítulo se describe el diseño e implementación del prototipo WSN utilizando la mejor topología física para el caso de una red inalámbrica de sensores ambientales, basándonos en los resultados del capítulo anterior.

Se describen las características de hardware de los dispositivos que conforman nuestro prototipo, tales como la placa base “Arduino uno”, módulos de comunicación Xbee, la interfaz”Xbee shield” que interconecta la placa base con

los módulos de comunicación, los Xbee Explorer USB y los sensores de adquisición de las variables ambientales.

En lo que respecta al software se explica cómo fueron programados cada uno de los módulos de comunicación Xbee para operar con la topología estrella, además se describe la programación que se realizó en la placa base para que recolecte, interprete y transmita los datos adquiridos en un formato adecuado hacia la estación coordinadora, la cual recibirá los datos de cada uno de los nodos y los almacenará en una base de datos, esto permitirá mantener un registro histórico de los valores de las variables ambientales y mediante el desarrollo de una aplicación presentarlos en forma gráfica, esta aplicación también permitirá mostrar datos y graficas en tiempo real, además nos permitirá monitorear el correcto funcionamiento de cada uno de los nodos que forman la red.

4.2 DISEÑO DEL PROTOTIPO WSN

Luego de haber definido a la Topología Estrella como la más adecuada para nuestro prototipo, se procede a diseñar la red inalámbrica de sensores, la cual estará compuesta por cuatro nodos finales y un coordinador tal como se observa en la Figura IV.65

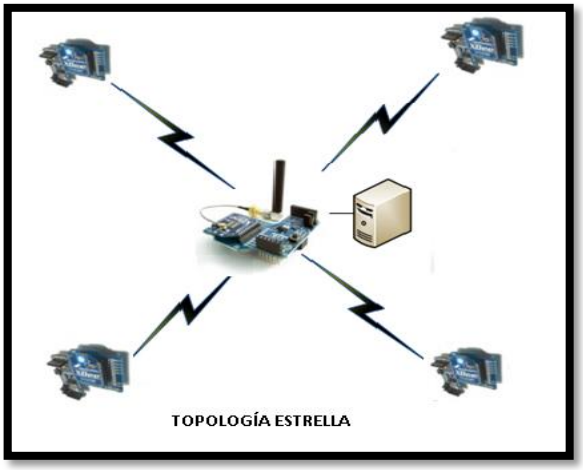

Figura IV.65: Topología Estrella

Fuente: Elaboración Propia

4.2.1 Funcionamiento general

Cada uno de los nodos recogerá información de temperatura y humedad relativa del medioambiente a través de un sensor que estará conectado a la placa base Arduino, la cual procesará la información y le agregara una cabecera a los datos para que cuando el coordinador los reciba a través de los módulos de comunicación Xbee pueda identificar de que nodo provienen, el coordinador está directamente conectado a una computadora, aquí los datos recogidos desde los cuatro nodos serán interpretados y guardados en una base de datos, para tener una interfaz más amigable con el usuario se optó por diseñar una aplicación, aquí se pueden observar las gráficas históricas y en tiempo real de cada uno de los sensores, lo que permite que se monitoree y se tenga control eficaz de toda la red WSN.

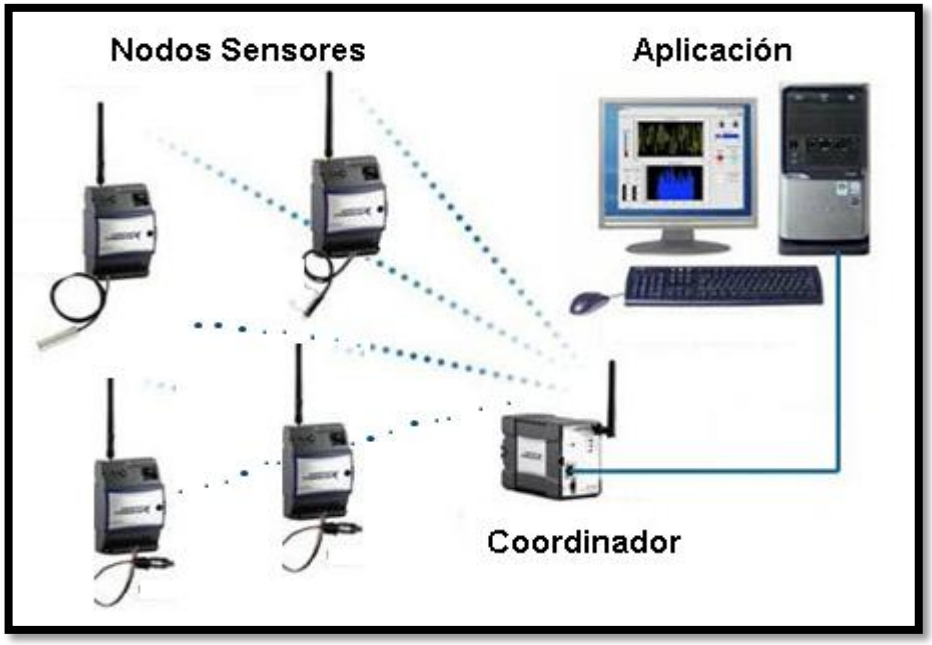

Figura IV.66. Funcionamiento General

Fuenete:http://www.ni.com/cms/images/devzone/tut/figure1_20100521153802.jpg

A continuación se explica cómo están programados cada uno de los dispositivos para que se interconecten y recojan los datos desde el medioambiente y los trasladen hasta un computador.

4.2.2 Dispositivos Finales (Nodos)

Un nodo sensor está compuesto de cuatro componentes básicos, una unidad sensora (sensor DHT11), una unidad de proceso (Arduino Uno), una unidad del transmisor-receptor (Xbee s1), y una unidad de potencia (batería de 9V), tal como se ve en la Figura IV.67.

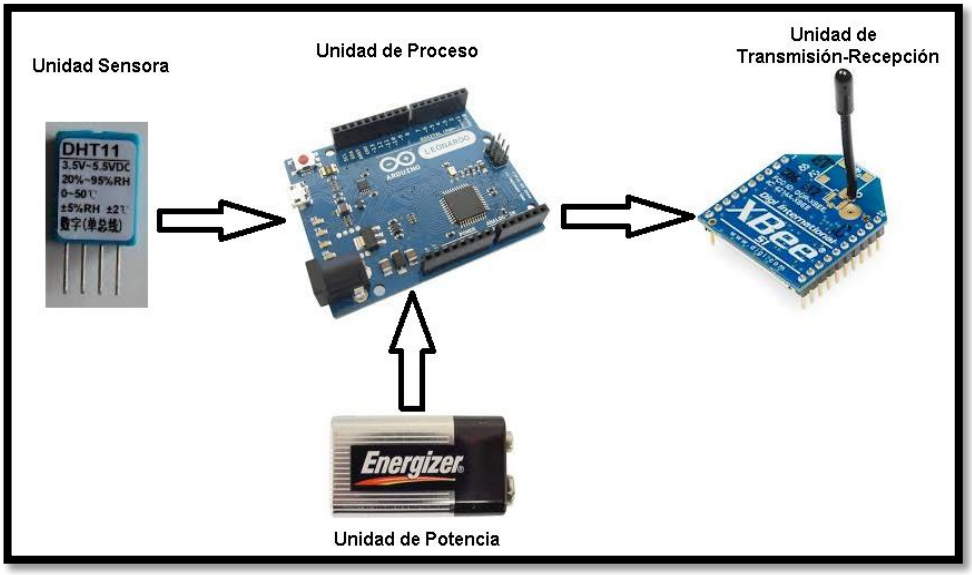

Figura IV.67: Componentes de un Nodo Sensor

Fuente: Elaboración Propia

La unidad sensora tiene como función recolectar información del medio y cada cierto tiempo enviará una señal hacia la unidad de proceso, una vez que los datos han sido procesados la unidad de Transmisión-Recepción (Xbee s1) los enviará hacia el coordinador. La unidad de potencia estará directamente conectada al Arduino y este a la vez alimentará a la unidad sensora y a la unidad de Transmisión-Recepción.

4.2.2.1 Unidad Sensora

Para el prototipo se va a utilizar el DHT11 que es un sensor de temperatura y humedad relativa como el de la Figura IV.68, en el mercado existen muchos sensores de este tipo que dan medidas muy exactas pero ha mayor costo, por este motivo una solución mucho más sencilla es utilizar un componente discreto como el DHT11, la resolución no es tan precisa como la que proporcionan otros sensores teniendo un rango de error ±5%RH y ±2 °C, sin

embargo el precio de estos sensores son relativamente bajos, y se acoplan a las necesidades que se requieren por lo que se decidió utilizarlos para el prototipo.

4.2.2.1.1 DHT11

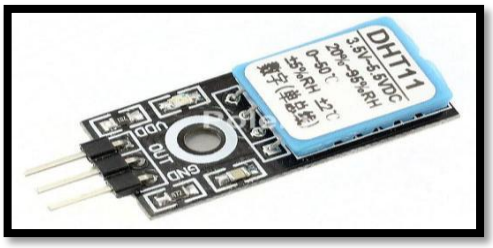

Figura IV.68: Sensor DHT11

Fuente: Elaboración propia

Generalmente el DHT11 consta de 4 pines, pero hoy en día vienen con un adaptado a tres pines y un filtrado de ruido lo que permite tener una medida más precisa tal como se ve en la Figura IV.68.

De las cuatro patas sólo se va a emplear 3 tal, dos son la alimentación entre 3 y 5.5V y la tercera se usa como bus de datos de aplicación hilo tanto para leer como para escribir, es necesario conectar una resistencia de 4.7 K o 5 K en este pin tal como lo indica la Figura IV.69.

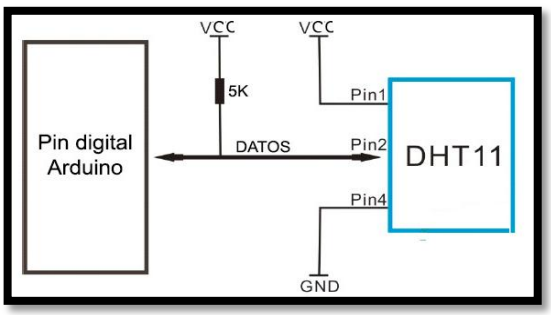

Figura IV.69: Conexión DHT11

Fuente: http://www.micropik.com/PDF/dht11.pdf

4.2.2.1.2 Interconexión con Arduino

La unidad de proceso es decir el Arduino Uno estará conectada directamente a la unidad de poder por lo que para alimentar al sensor se lo hace mediante los pines de VCC y GND del Arduino, y la señal se la conecta a cualquier pin de entrada digital, en este caso lo se hizo al pin digital 9, tal como se observa en las Figura IV.70.

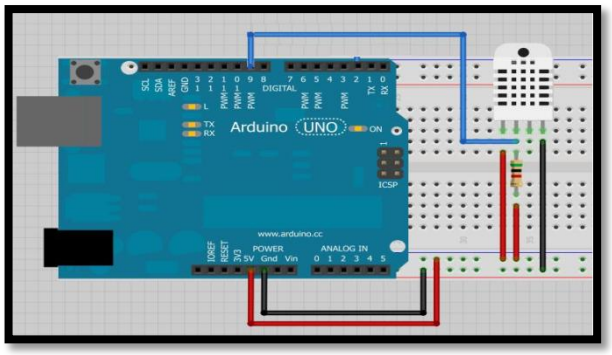

Figura IV.70: Conexión Arduino-DTH11

Fuente: http://tallerArduino.com/2012/12/24/sensor-dht11-humedad-y-temperatura-con-Arduino/

4.2.2.2 Unidad de Proceso

Los Nodos cuentan con un Arduino Uno V1.8.1 para la manipulación e interpretación de los datos que le envía la unidad sensora.

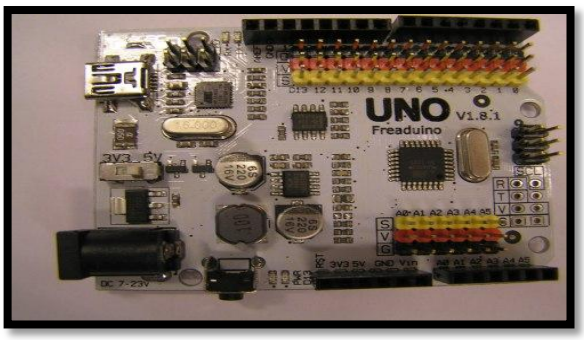

Figura IV.71: Arduino Uno V1.8.1

Fuente: Elaboración propia

La unidad de proceso debe estar alimentada por la unidad de poder ya sea alimentado directamente a una toma de corriente o mediante una batería, el Arduino puede ser alimentado con 7V – 23V de corriente directa (DC), y luego este tendrá que alimentar al sensor y al módulo Xbee. En la Figura IV. 72 se observa el Arduino uno alimentado por una batería de 9V.

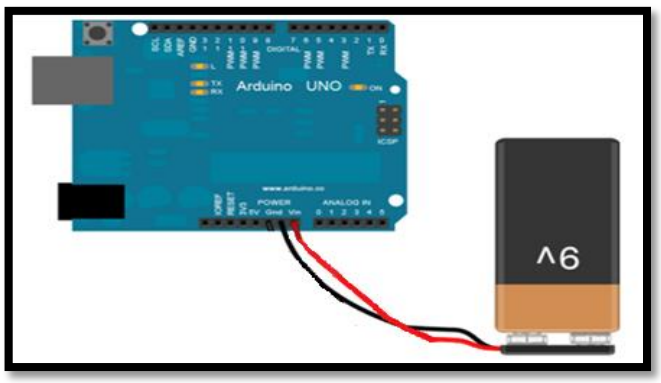

Figura IV.72: Arduino con batería de 9V

Fuente: http://www.robotc.cl/wiki/Tutorials/Arduino_Projects/RC_car_Hacking_Project/Controlling_the_motors.html

4.2.2.3 Unidad de Transmisión y Recepción

Esta unidad es muy importante en las redes WSN ya que permite transmitir la información recolectada de manera inalámbrica hacia el coordinador.

Los módulos Xbee pueden ser configurados desde el PC utilizando el programa X-CTU o bien desde tu microcontrolador. Los Xbee pueden comunicarse en arquitecturas punto a punto, punto a multi punto o en una red mesh. La elección del módulo Xbee correcto pasa por escoger el tipo de antena (chip, alambre o conector SMA) y la potencia de transmisión (2mW para 300 pies o 60mW para hasta 1 milla)

Se decidió trabajar con el Xbee S1 como el de la Figura IV.73, ya que este cumple con los requerimientos mínimos que se necesitan: soporta la configuración de la topología estrella, permite un alcance de 100 m con línea de vista y hasta 30 m con interferencias, además su precio es muy conveniente (35 dólares americanos).

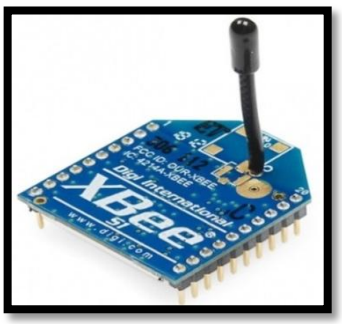

Figura IV.73: Xbee s1

Fuente: http://www.hobbytronics.co.uk/image/cache/data/XBee/XBee_1mw_wire_s1-500x500.jpg

4.2.2.3.1 Conexión con la unidad de proceso.

Es importante conocer para que sirve cada uno de los 20 pines del módulo, para entender mejor se puede ver en la Figura IV.74.

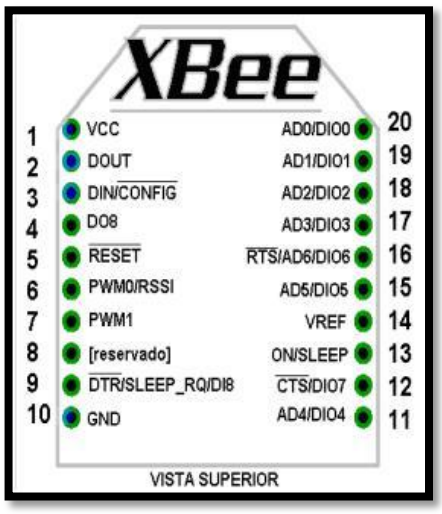

Figura IV.74 : Pines del Xbee S1

Fuente: http://3.bp.blogspot.com/-EsBAEOpE9PE/TofmY_K1huI/AAAAAAAAAE8/G7S1yjA_Ji0/s1600/Pines.jpg

Para que exista conexión entre el Xbee y la placa Arduino Uno se debe conectar cuatro pines como se indica en la Tabla VI.XVI aunque para un mejor funcionamiento se puede conectar el pin de Reset y de Sleep.

Tabla IV.XIX. Pines de Conexión Arduino-Xbee

XBEE S1	ARDUINO UNO
Pin 1	VCC (3.3 V)
Pin 2	Tx
Pin 3	Rx
Pin 5	GND (Opcional para Reset)
Pin 9	GND (Opcional para Sleep)
Pin 10	GND

Fuente: Elaboración Propia

En la Figura IV.75 se observa el diagrama de interconexión.

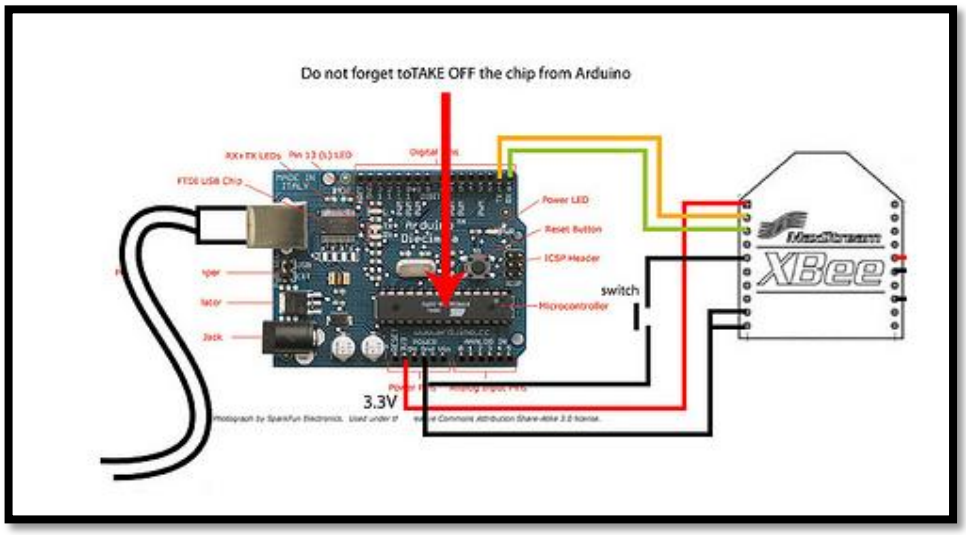

Figura IV.75: Interconexión Arduino-Xbee

Fuente: http://www.kobakant.at/DIY/?p=204

4.2.3 Dispositivo Coordinador

El dispositivo coordinador está formado por un módulo de comunicación Xbee S1 el cual se encarga de recibir los datos de los Nodos Sensores o también conocidos como End Device, y para enviar los información hacia la base de datos se conecta directamente a la computadora mediante un Xbee USB Explorer conectado al puerto serial. Como lo muestra la Figura IV. 76.

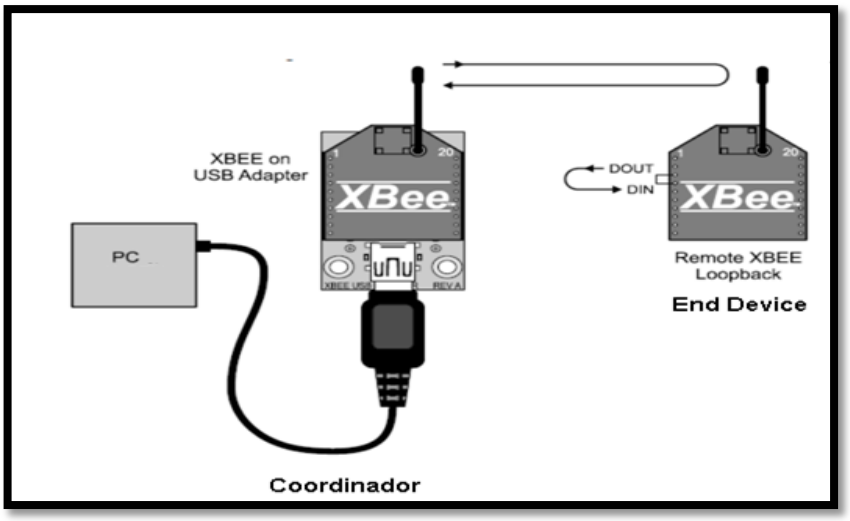

Figura IV.76: Coordinador Xbee

Fuente: http://www.geekytheory.com/wp-content/uploads/2013/04/tumblr_lzay5levCL1qf00w4.png

4.2.4 Aplicación

Al final cuando el coordinador envié los datos hacia el computador se los debe interpretar, identificar de qué nodo proviene, almacenar y por último mostrar en un formato entendible por el usuario así como se muestra en la Figura IV. 77.

Para lograr esto se desarrolló una aplicación en Delphi 7 con una interfaz gráfica amigable que permita extraer los datos desde el puerto serie del módulo Xbee y luego se puedan almacenar en una base de datos, la aplicación también presentará gráficas de cada uno de los sensores en tiempo real.

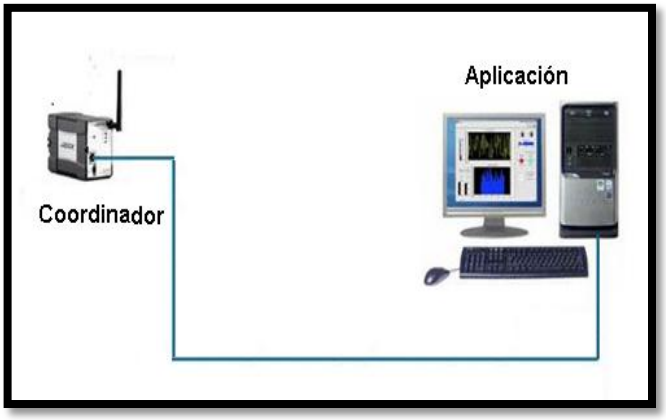

Figura IV.77: Aplicación

Fuente: Elaboración Propia

4.2.4.1 Diseño de la Base de Datos

Lo que se necesita es que la base de datos sea capaz de ir almacenando los datos de acuerdo al nodo, es decir se tendrá cuatro tablas una por cada nodo y cada nodo puede tener varios sensores conectados, en este caso solo se tiene un dispositivo sensor que envía dos señales una de temperatura y otra de humedad relativa, tal como se observa en la Figura IV.78.

Para esto se crea una base de datos en el software MySQL que se llamará SENSORES, y contendrá cuatro tablas una por cada nodo.

Cada nodo contendrá cuatro columnas con su respectivo formato de datos:

Tabla IV.XX. Nombre y Tipo de Datos

Nombre	Tipo
Fecha	Date
Hora	Time
Temperatura	Integer
Humedad	Integer

Fuente: Elaboración propia

Al final se tendrá una base de datos como la de la figura de abajo.

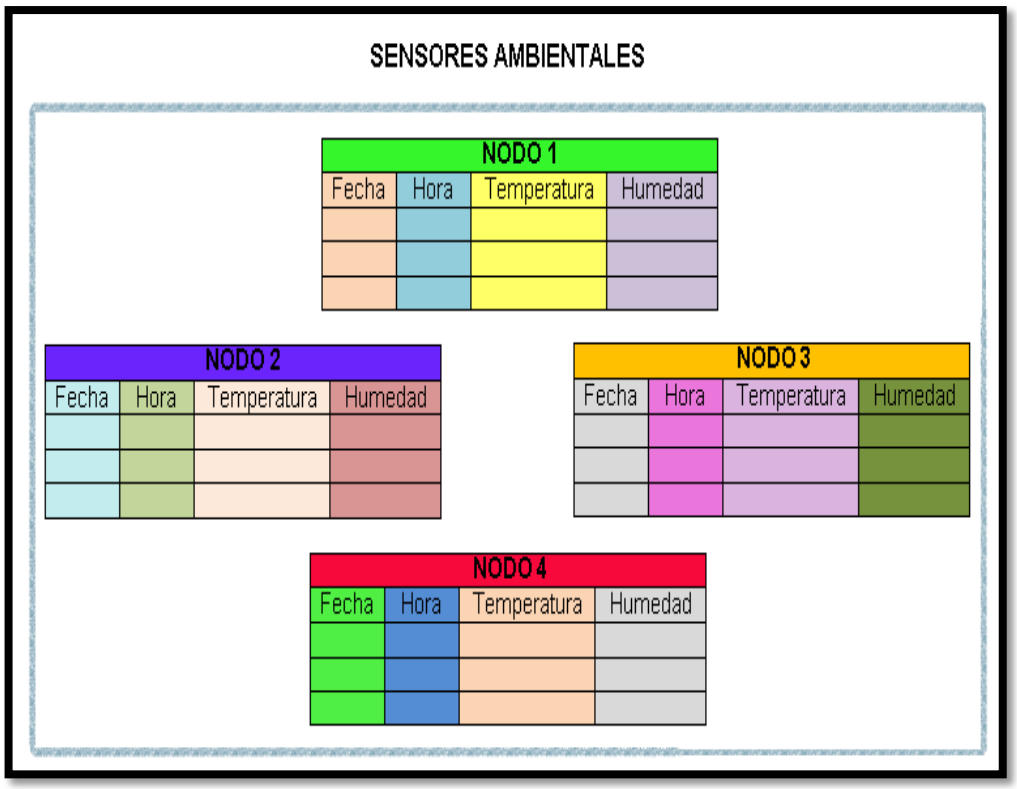

Figura IV.78: Diseño de la Base de Datos

Fuente: Elaboración propia

4.3 IMPLEMENTACIÓN DEL PROTOTIPO WSN

Una vez que se ha diseñado el prototipo, se procede a su implementación, a continuación se describe paso a paso la configuración de cada uno los dispositivos utilizados, así como también se explica cómo se realizó la aplicación.

4.3.1 Configuración de los Módulos Arduino Uno

Una vez que el sensor envía una señal al Arduino este debe ser capaz de interpretar dicha señal, y posteriormente procesarla antes ser enviada al coordinador.

Para esto se necesita los módulos Arduino uno, un cable mini usb y el software Arduino que se puede descargar desde la página oficial: http://arduino.cc/es/Main/Software

4.3.1.1 Instalación de Librerías

Arduino por defecto viene cargado con varias librerías, pero como se va a usar el sensor DHT11 se debe descargar la librería específica para que funcione este sensor.

La librería para DHT11 se la puede descargar desde la página: https://github.com/adafruit/DHT-sensor-library .

Una vez que se tenga el archivo DHT11.XX.zip en nuestra computadora se debe descomprimir y pegar en la carpeta de Arduino>libraries que se creó automáticamente al momento de instalar el software *Arduino.*

4.3.1.2 Programación de la Placa Arduino

Una vez que se tiene el software listo para usar se procede a programar, lo que se necesita que haga es recibir los datos desde el sensor y almacenar 10 valores distintos tanto de humedad y temperatura y luego sacar la media (promedio) antes de enviar al coordinador, con esto obtuvo menos errores a causa de ruido que se puede adherir a la señal debido a factores externos.

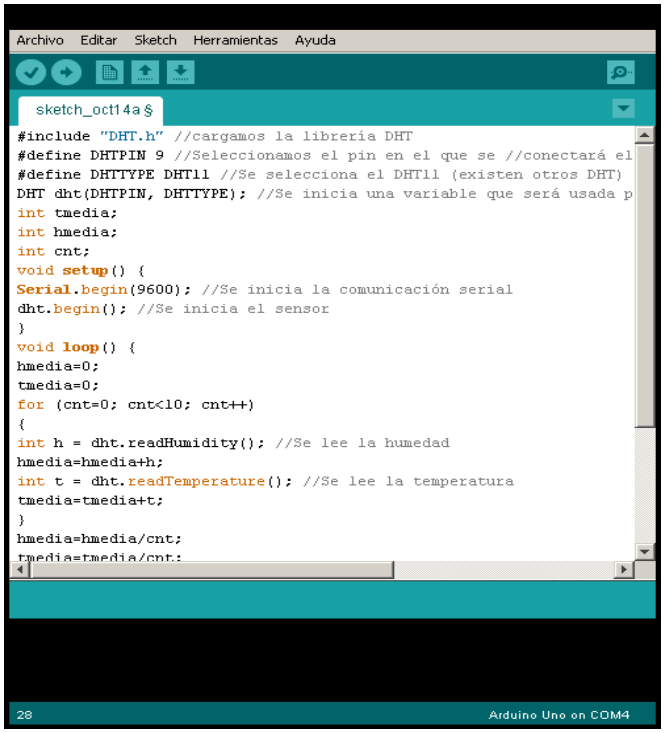

Figura IV.79: Programación Arduino

Fuente: Elaboración propia

Para programar se debe iniciar el software Arduino, programar lo que se desea que realice (Se explica abajo cada línea del código), comprobar que el código esté bien configurado (Dar clic en el ícono de Visto), seleccionar el puerto para este caso es el puerto COM 4 y por ultimo cargarlo a la placa (Dar clic en el ícono de una Flecha).

La programación final es la siguiente:

```
#include “DHT.h”        // Se carga la librería DHT
#define DHTPIN 9       // Seleccionar el pin en el que se conectará el sensor
#define DHTTYPE DHT11          //Se selecciona el DHT11 (existen otros DHT)
DHT dht(DHTPIN, DHTTYPE);   //Se inicia una variable para comunicarse con el sensor
int tmedia;                 // Se crea variables integer para sacar las medias y para el contador.
Int hmedia;
int cnt;
```

```
//Se inicia la configuración
void setup() {
Serial.begin(9600);          //Se inicia la comunicación serial
dht.begin();                  //Se inicia el sensor
}
void loop() {
hmedia=0;                  // Se inician las variables para sacar las medias
tmedia=0;
for (cnt=0; cnt<10; cnt++)          //Se crea un For para que repita el proceso 10 veces
{
int h = dht.readHumidity();    //Se lee la humedad
hmedia=hmedia+h;              //Se suma el valor nuevo de humedad
int t = dht.readTemperature();     //Se lee la temperatura
tmedia=tmedia+t;         //Se suma el valor nuevo de temperatura
}
hmedia=hmedia/cnt;          //Se saca la media de humedad
tmedia=tmedia/cnt;                //Se saca la media de temperatura
Serial.println(hmedia+11000); //Se imprime la media de humedad y se añade una cabecera
Serial.println(tmedia+12000);      // Se imprime la media de temperatura y se añade una cabecera
delay(2000);              //Se espera 2 segundos para seguir leyendo //datos
}
```

4.3.2 Módulos de Comunicación Xbee.

Ya que en el capítulo anterior se determinó que la topología óptima es la Estrella, se debe configurar los Xbee de los nodos finales de tal manera que solo envíen su información hacia el coordinador con esto se logra que optimizar el uso de energía de las baterías porque de otra forma los datos se estarían enviando hacia todos los nodos y habría redundancia de los paquetes, mientras

que al Xbee coordinador se lo debe configurar de tal manera que permita recibir la información de todos los nodos finales, tal como se ve el sentido de comunicación en la Figura IV.80.

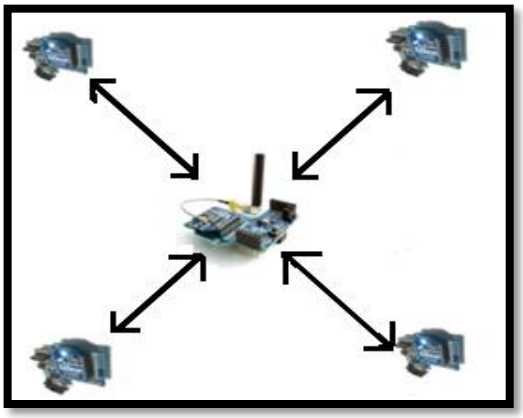

Figura IV.80: Sentidlo de Comunicación entre Xbees

Fuente: Elaboración Propia

Para poder configurar los módulos Xbee se necesita adicionalmente un Xbee explorer USB y el software X-CTU.

4.3.2.1 Software X-CTU

El entorno gráfico que muestra este programa desarrollado por Digi lo hace muy amigable para el usuario, se lo puede descargar libremente desde la página oficial http://www.digi.com/. Una vez que se tenga correctamente instalado se abre dando doble clic en el icono que se nos crea automáticamente en el escritorio.

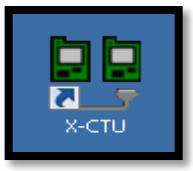

Figura IV.81: Ícono X-CTU

Fuente: Elaboración Propia

Al correr el programa, se puede observar cuatro pestañas que conforman toda ventana del software X-CTU como se ve en la Figura IV.82. Cada una de esas pestañas tiene una función diferente, que se explicará a continuación:

Figura IV.82: Ventana principal X-CTU

Fuente: Elaboración Propia

4.3.1.1.1 Pestaña PC Settings

Cuando el programa se inicia, por defecto, la pestaña seleccionada es la “PC Settings”. La pestaña PC Settings se divide en tres áreas básicas: Com Port Setup (configuración delpuerto COM), Host Setup (configuración del Host), y User Com Ports (usuario de los puertos COM).

<u>COM Port Setup (Configuración del Puerto COM)</u>

La pestaña PC Settings permite al usuario seleccionar un puerto COM como se observa en la Figura IV.83 se observa el COM 7, luego de seleccionarlo se puede configurar. Algunas de las configuraciones son:

- Baud Rate (Velocidad de Transmisión)
- Flow Control (Control de Flujo)
- Data bits (Bit de Datos)
- Parity (Paridad)
- Stop bit (Bit de Parada)

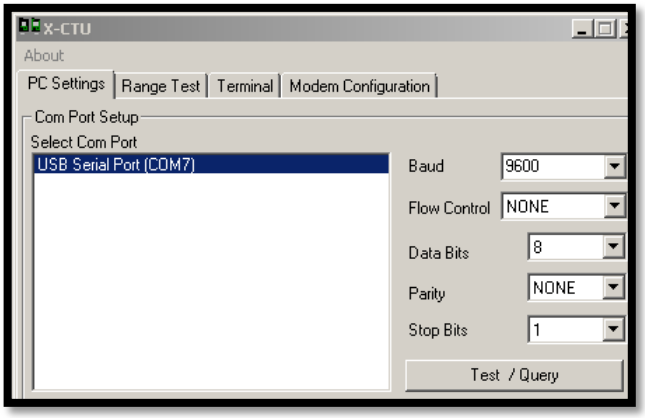

Figura IV.83: Pestaña PC Settings

Fuente: Elaboración Propia

Es importante que la opción de Baud se encuentre puesta en 9600 y Data Bits en 8, ya que se tendrá una comunicación serial entre el módulo y la computadora.

Test/Query (Prueba/Consulta)

Se utiliza para probar el puerto COM seleccionado y las configuraciones de la PC. Si las configuraciones y el puerto COM son correctos, usted recibirá una respuesta similar a la que se muestra en la Figura IV.84

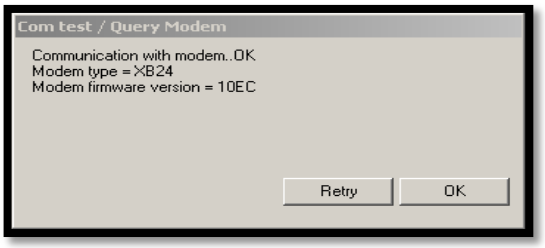

Figura IV.84: Test /Query

Fuente: Elaboración Propia

4.3.1.1.2 Pestaña Range Test

En la pestaña Range Test (Figura IV.85), es posible enviar una cadena de datos de cualquier tipo para probar el rango de alcance de la señal. Esto

genera automáticamente datos y los envía por el módulo, de tal forma que permite verificar cuales datos llegan buenos y cuales no y a partir de esa estadística determinar el rango o alcance de la señal.

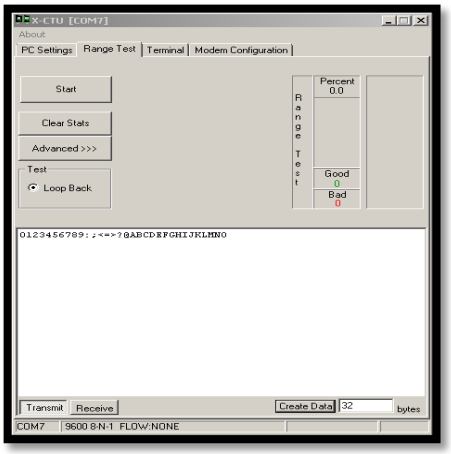

Figura IV.85: Pestaña Range Test

Fuente: Elaboración propia

4.3.1.1.3 Pestaña Terminal

La pestaña Terminal tiene tres funciones básicas:

- Terminal Emulador
- Capacidad para enviar y recibir datos predefinidos (Assemble packet)
- Capacidad para enviar y recibir datos en formatos hexadecimal y ASCII (Show/Hide hex)

<u>Pantalla principal de la ventana Terminal</u>

La porción en blanco de esta pestaña es donde la mayoría de las comunicaciones de información se producirán, mientras el X-CTU sea usado como un terminal emulador. El texto en azul es lo que se ha escrito y será enviado hacia el módulo XBEE mediante el puerto serial, mientras que el texto

en rojo es la entrada de datos (respuesta) desde el módulo XBEE mediante el puerto serial tal como se observa en la Figura IV.86.

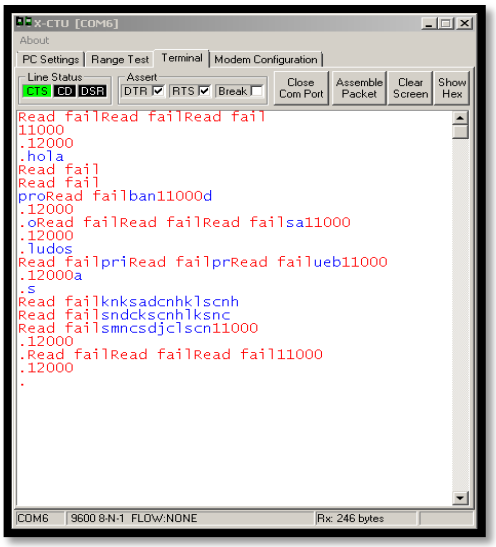

Figura IV.86: Pestaña Terminal

Fuente: Elaboración propia

Assemble Packet

La opción Assemble Packet que se encuentra en la pestaña Terminal está diseñada para permitir al usuario ensamblar un paquete de datos en cualquiera de los dos caracteres ASCII o hexadecimal. Esto se logra mediante la selección de la ventana Assemble Packety eligiendo ya sea en ASCII (por defecto) o hexadecimal. Una vez seleccionado, el paquete de datos, es ensamblado escribiendo los caracteres deseados como se ilustra en la Figura IV.87 y haciendo clic en el botón Send Data.

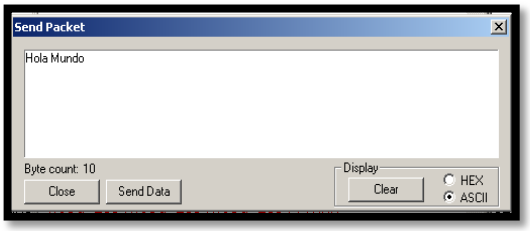

Figura IV.87: Assemble Packet

Fuente: Elaboración propia

4.3.1.1.4 Pestaña Modem Configuration

La pestaña Modem Configuration tiene cuatro funciones básicas:

1. Proveer una interfaz gráfica de usuario con el firmware del módulo XBEE.

2. Leer y escribir firmware al microcontrolador interno del XBEE.

3. Descargar archivos de actualización del firmware, ya sea desde el Internet o desde un archivo comprimido.

4. Guardar o leer el perfil de un módem

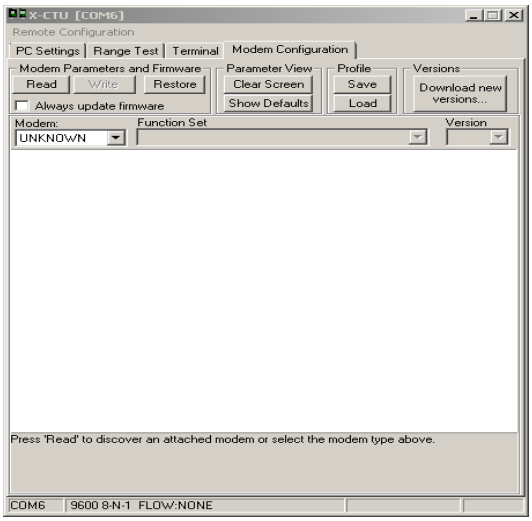

Figura IV.88: Pestaña Modem Configuration

Fuente: Elaboración propia

Leer el firmware del XBEE

Para leer el firmware del módulo XBEE en la pestaña Modem Configuration, seleccione “Read” que se encuentra en la etiqueta Modem Parameters and Firmware (véase la Figura IV.89).

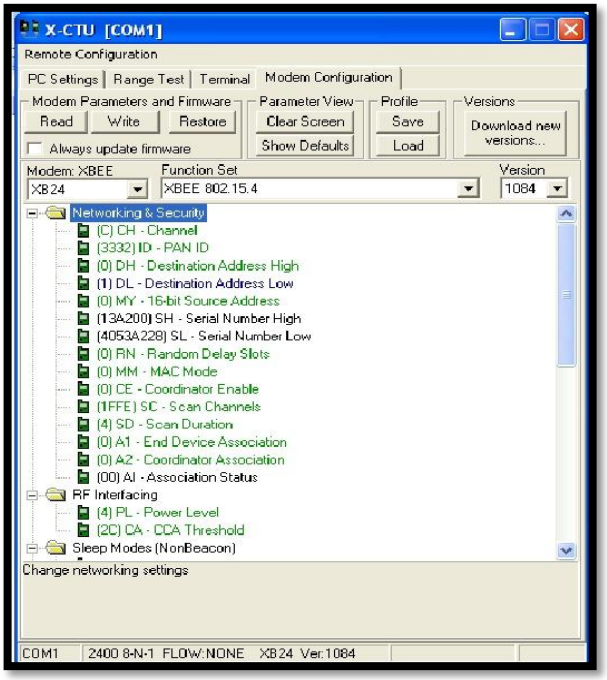

Figura IV.89: Read Firemware

Fuente: Elaboración propia

Cambios en el Firmware del XBEE

Una vez que se ha leído el firmware, los ajustes de configuración se muestran en tres colores tal como se ve en la Figura IV.89.

- Negro: no configurable o sólo de lectura
- Verde: Valor por defecto
- Azul: Parámetros especificados por el usuario

Para modificar cualquiera de los parámetros configurables por el usuario, haga clic en el comando e introduzca el nuevo valor para ese parámetro. Para facilitar la comprensión de un comando específico y una vez que el comando está seleccionado, una breve descripción junto con sus límites es presentada en la parte inferior de la pantalla. Una vez que todos los nuevos valores se han

introducido, estos valores están listos para ser guardados en la memoria no volátil del módulo XBEE.

Guardar cambios del Firmware en el XBEE

Para escribir los cambios de parámetros en la memoria no volátil del módulo XBEE, haga clic en el botón Write situado en sección Modem Parameters and Firmware (Figura IV. 89)

Descargar Archivos de Actualización del Firmware.

Permite al usuario descargarlos archivos de actualización del firmware, ya sea por Internet o instalarlos desde un disco o CD. Esto se logra mediante el siguiente proceso:

1. Clic sobre la opción Download New Versions

2. Haga clic sobre Web para descargar los nuevos archivos de firmware de Internet. Caso contrario dar clic sobre File.

3: Haga clic en OK y Done cuando se le pida.

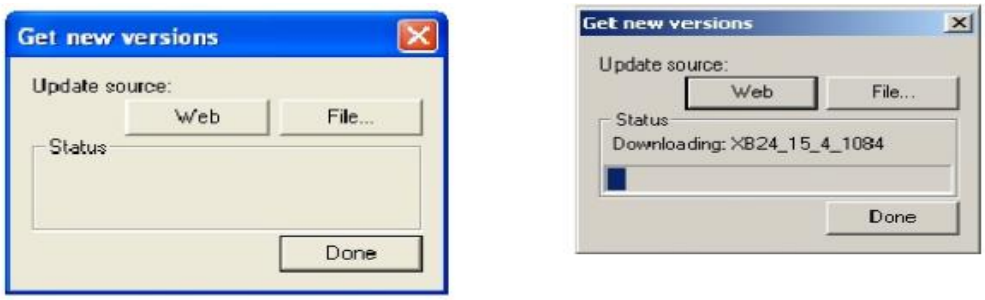

Figura IV.90: Download New Version

Fuente: Elaboración propia

4.3.2.2 Configuración del Coordinador Xbee

En la siguiente imagen se puede ver la pestaña Modem Configuration con la configuración necesaria para el correcto funcionamiento del módulo Xbee que actuará como coordinador.

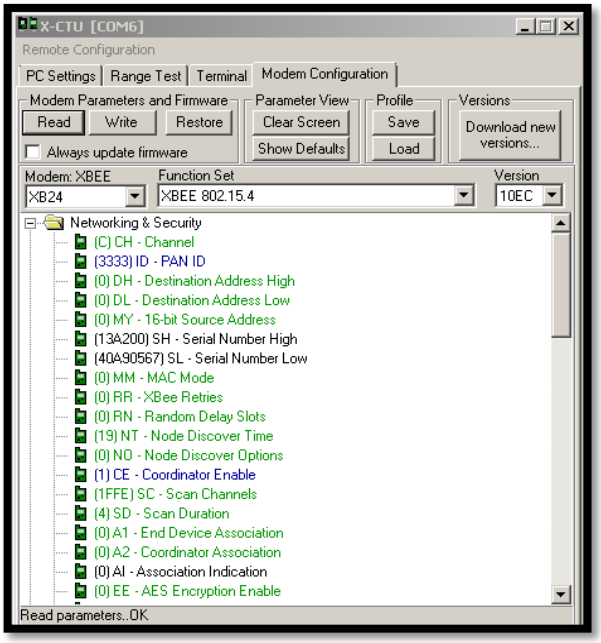

Figura IV.91: Configuración Coordinador Xbee

Fuente: Elaboración propia

En la Figura IV.91 se puede ver la configuración necesaria para el correcto funcionamiento, a modo de resumen se muestra la configuración del coordinador en la siguiente tabla.

Tabla IV.XXI. Parámetros del Coordinador

Parámetro	Valor
Channel	C
PAN ID	3333
Destination Address High	0
Destination Address Low 0	0
Serial Number High	13A200
Serial Number Low	40A90567

Fuente: Elaboración Propia

- *Channel C* es el canal que se ha asignado al módulo Xbee para realizar la comunicación, al indicarle al módulo que trabaje en el Channel C se está trabajando en el rango de frecuencia de: 2,4075 – 2,4125GHz.
- El parámetro *PAN ID* es el identificador de la red, en este caso se ha indicado que use el identificador 3333.
- Los parámetros de *Destination Address High y Destination Address Low* están configurados con el valor cero, esto le indica al módulo Xbee del coordinador que tendrá recepción y transmisión desde todos los nodos finales que estén en el mismo Channel y que contengan la misma PAN ID. Se configura de esta forma ya que desea tener una topología estrella.
- Los parámetros Serial Number High y *Serial Number Low* son los datos identificativos del módulo Xbee del coordinador (No son modificables por el usuario y se los puede encontrar impresos en la parte posterior de cada módulo).

4.3.2.3 Configuración de los Nodos Xbee

En la siguiente imagen se puede ver la pestaña Modem Configuration con la configuración necesaria para el correcto funcionamiento de uno de los módulos Xbee ubicados en el dispositivo final. Se debe realizar el mismo proceso para todos los dispositivos finales.

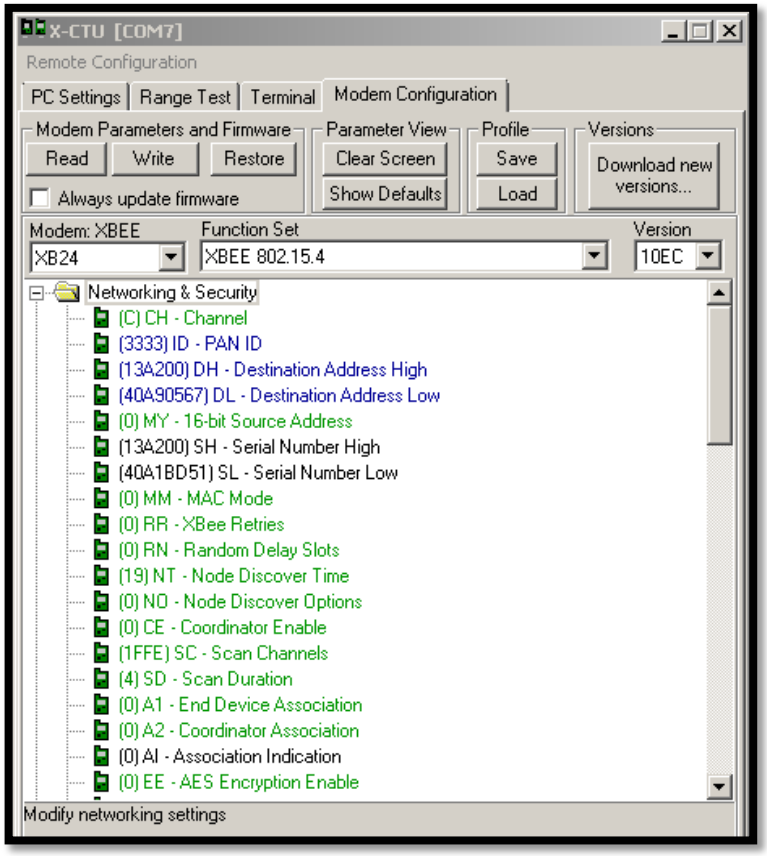

Figura IV.92: Configuración Dispositivo Final Xbee

Fuente: Elaboración propia

En la Figura IV.92 se puede ver la configuración necesaria para el correcto funcionamiento de los dispositivos finales con la topología estrella, a modo de resumen se muestra la configuración básica para del dispositivo final en la siguiente tabla.

Tabla IV.XXII. Parámetros del Nodo Xbee

Parámetro	Valor
Channel	C
PAN ID	3333
Destination Address High	13A200
Destination Address Low 0	40A90567
Serial Number High	13A200
Serial Number Low	40A1BD51

Fuente: Elaboración Propia

- *Channel C* es el canal que se ha asignado al módulo Xbee del dispositivo final para realizar la comunicación, al indicarle al módulo que trabaje en el Channel C se está trabajando en el rango de frecuencia de: 2,4075 – 2,4125GHz.
- El parámetro *PAN ID* es el identificador de la red, en este caso se ha indicado que use el identificador 3333.
- Los parámetros de *Destination Address High y Destination Address Low* están configurados con los valores Serial Number High y Serial Number Low del coordinador, de esta forma el dispositivo final únicamente enviara la información al módulo del coordinador. (Característica de la Topología estrella)
- Los parámetros Serial Number High y *Serial Number Low* son los datos identificativos del módulo Xbee de cada dispositivo final (No son modificables por el usuario y se los puede encontrar impresos en la parte posterior de cada módulo).

4.3.3 Aplicación

Para que los datos sean presentados en un formato claro para el usuario se desarrolló una aplicación en Delphi que permite almacenar la información en una base de datos y también muestra graficas en tiempo real de cada uno de los nodos.

4.3.3.1 Base de Datos

Para la base de datos se utilizó PhpMyAdmin, es una de las aplicaciones más populares para la gestión de bases de datos MySQL. Es una herramienta gratuita escrito en PHP. A través de este software se puede crear, modificar, borrar, eliminar, importar y exportar las tablas de bases de datos MySQL.

XAMPP Control Panel

Esta herramienta que permite administrar la base de datos vía web, para esto se debe correr los procesos de Apache y MySQL tal como se ve en la Figura IV.93.

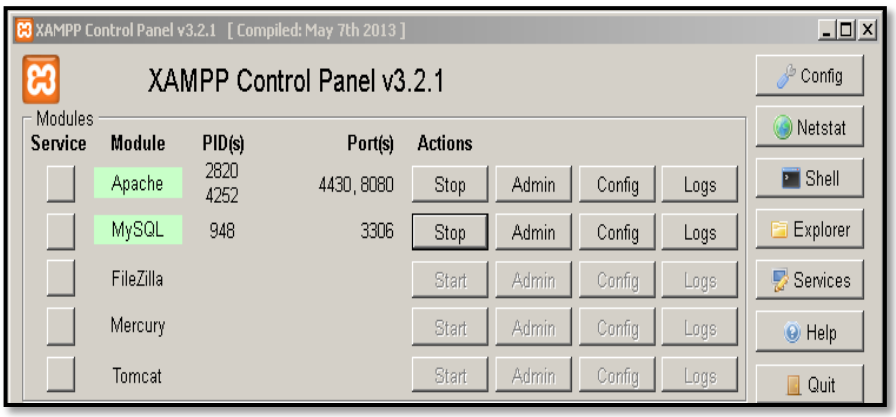

Figura IV.93: XAMPP

Fuente: Elaboración propia

Para crear la base de datos y realizar cualquier modificación hay que dar clic en Admin para MySQL y se abre una pestaña web, y ahí se crea la base de datos con las características deseadas, la interfaz gráfica es muy intuitiva y fácil de usar. Finalmente la base de datos queda tal como se observa en la Figura IV.94.

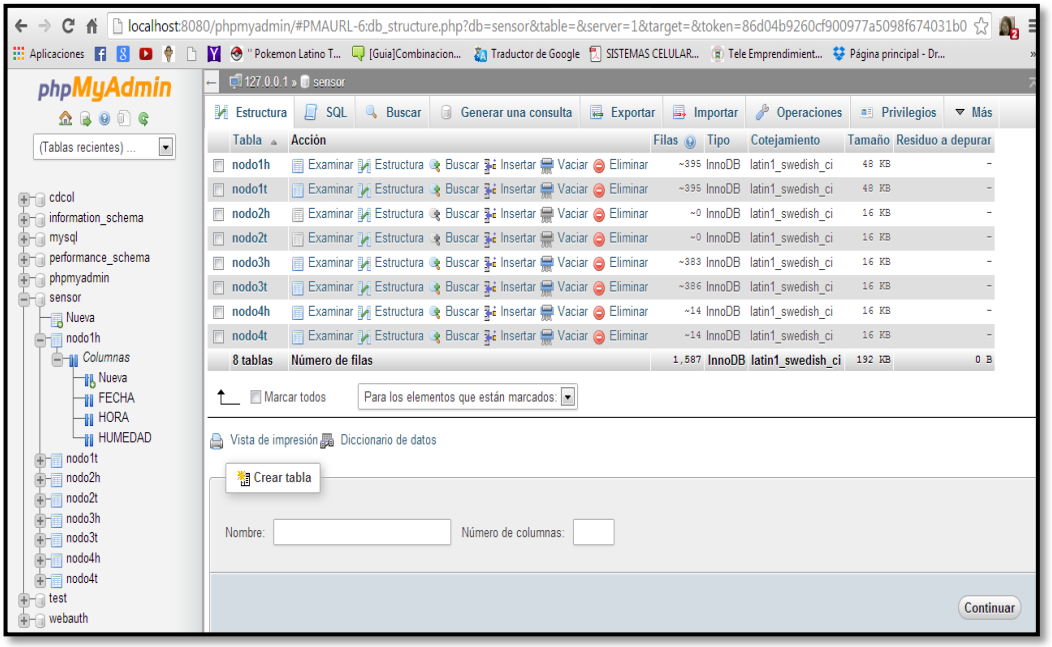

Figura IV.94: Base de Datos

Fuente: Elaboración propia

4.3.3.2 Configuración Delphi

A continuación se va a explicar brevemente los componentes principales para el desarrollo de la aplicación.

Como se observa en la Figura IV. 95 Los datos que le llegan al coordinador se van a extraer de manera serial a través de ComPort hacia Delphi, para almacenar la información en una base de datos se utiliza ADOTable, el componente DBChart permite graficar lo que se almacena en la base de datos.

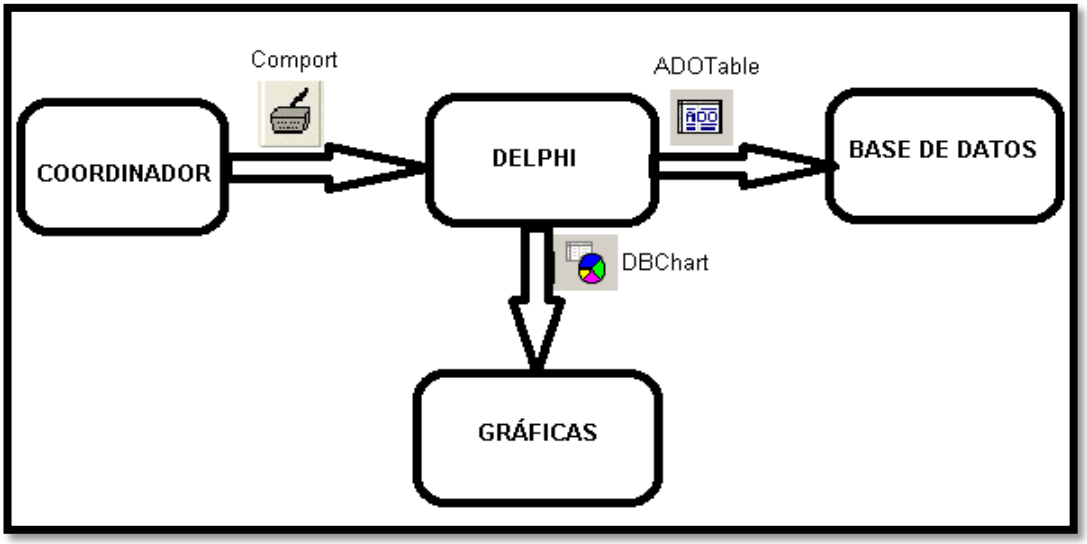

Figura IV.95: Esquema Base de Datos

Fuente: Elaboración propia

ComPort

Para sacar los datos desde el coordinador hacia la computadora se utilizó el componente ComPort (Puerto Com), sirve para establecer comunicaciones serie para Delphi y C++ Builder. Permite el acceso al puerto serie del PC en Windows XP, Vista y 7. TcomPort se basa en la API de Windows por lo que es totalmente compatible con los controladores o drivers pasarela tipo USB – RS232, este se lo encuentra en la paleta de componentes, en la pestaña CportLib, tal como se puede observar en la Figura IV. 96.

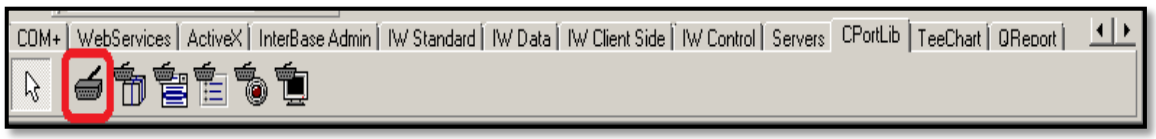

Figura IV.96: ComPort

Fuente: Elaboración propia

Para que tenga interconexión hacia la computadora se configuró los parámetros tal como se ve en la Figura IV.97.

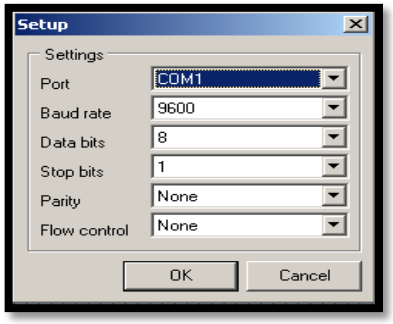

Figura IV.97: Setup ComPort

Fuente: Elaboración propia

ADO Table

Es un componente de acceso a datos de la pestaña ADO (Figura IV.98) que nos permite mantener un conjunto de registros cargados sin necesidad de tener una conexión activa a la base de datos.

Figura IV.98: ADO Table

Fuente: Elaboración propia

Se necesita un componente ADO Table para cada tabla que se vaya a interconectar de la base de datos, solo se explicará una y para las 7 restantes el proceso es similar. Las configuraciones se las puede observar en la Figura IV.99, la opción TableName permite seleccionar desde una pestaña desplegable una de las tablas que contiene la base de datos a la cual se interconecta.

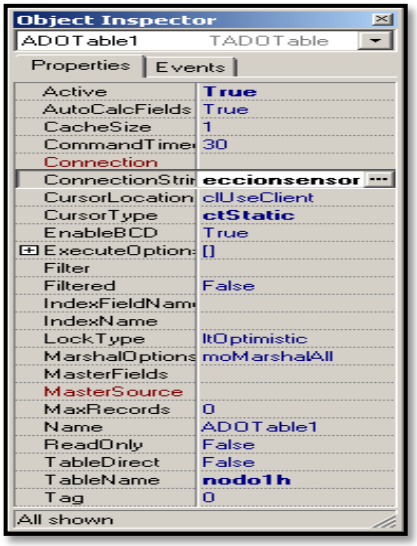

Figura IV.99: Configuración ADO Table

Fuente: Elaboración propia

Para interconectar con la base de datos se sigue en siguiente proceso:

En la opción Conection String de la barra de object inspector se da clic sobre los 3 puntos (…), en la ventana que aparece se elige Use Connection String y clic sobre Build (Figura IV.100).

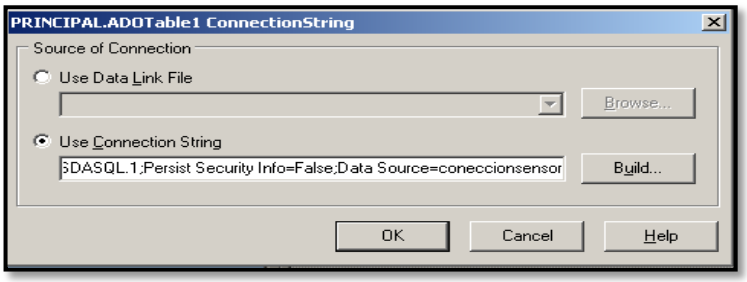

Figura IV.100: Connection String

Fuente: Elaboración propia

Aparece una ventana igual a la de la Figura IV. 101 en la que se debe especificar el origen de datos es decir el ODBC de la base de datos con la que

se interconecta en este caso se llama coneccionsensor. Para verificar que existe conexión dar clic sobre la opción Probar Conexión y después Aceptar.

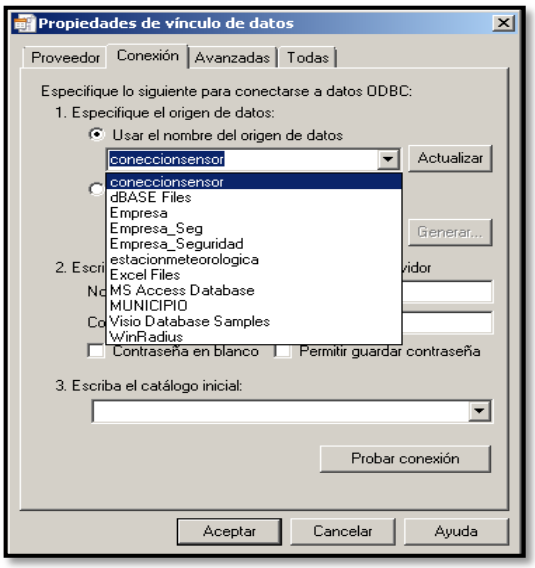

Figura IV.101: Propiedades de Vínculo de Datos

Fuente: Elaboración propia

Una vez que se regresa a la ventana PRINCIPAL.ADOTable (Figura IV.101) dar clic en Ok y se interconecta con la base de datos.

DBChart

El componente DBChart es una poderosa herramienta para crear diagramas de bases de datos y gráficos. Tal como se ve en la Figura IV.102 este componente lo se encuentra en la pestaña DataControls.

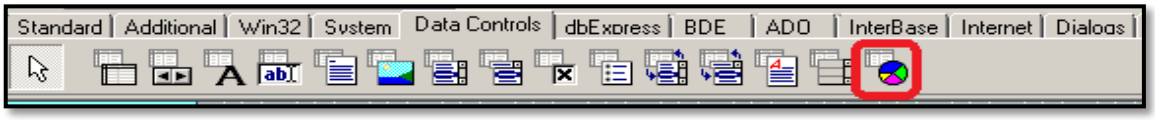

Figura IV.102: DBChart

Fuente: Elaboración propia

Al utilizar el DBChart con el motor de gráficos TeeChart puede hacer rápidamente los gráficos directamente a los datos en bases de datos sin

necesidad de código. TDBChart Delphi se conecta a cualquier fuente de datos. Registros ADO son compatibles de forma nativa. No se requiere código adicional, simplemente hay que especificar como se quiere que se vea. El editor gráfico (Figura IV.103) le guiará por los pasos para conectarse a los datos que ni siquiera se tiene que ir al Inspector de Objetos.

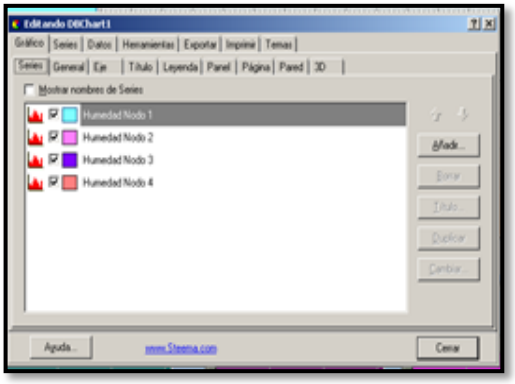

Figura IV.103: TDBChart

Fuente: Elaboración propia

Finalmente se tendrá una ventana como la de la Figura IV.104.

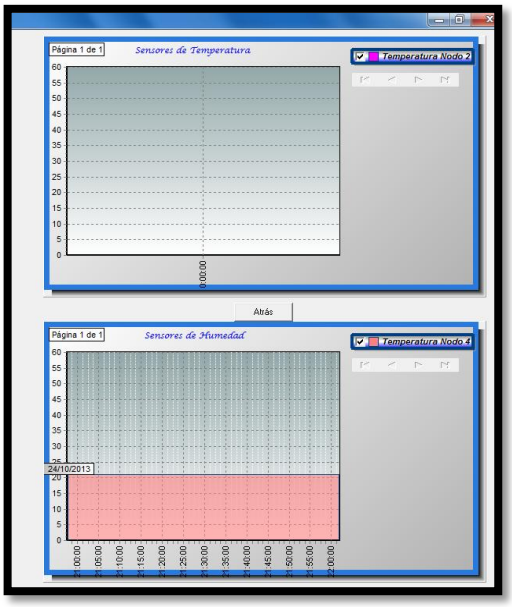

Figura IV.104: Panel Gráfico

Fuente: Elaboración propia

4.3.4 Puesta en marcha de la red WSN

Para evitar que se dañe el circuito de los nodos que serán colocados en los exteriores por factores externos como el sol, lluvia, viento, etc. Se diseñó una caja de acrílico para impermeabilizar. La antena del módulo de comunicación Xbee se dejó afuera de la caja para lograr que la señal no se atenué y alcance mayor distancia, el sensor DHT11 también se lo dejo afuera para que obtenga datos con mayor precisión. Cada nodo o dispositivo final quedará tal como se ve en la Figura IV. 105.

Figura IV.105: Dispositivo Final

Fuente: Elaboración propia

Se ubicó los nodos de la siguiente forma:

- Nodo 1: Aula uno de CISCO
- Nodo2: Aula dos de CISCO
- Nodo 3: Oficina de dirección de CISCO
- Nodo 4: Edificio de la EPEC

Por otra parte para el coordinador que solo cuenta con un módulo Xbee y que estará ubicado en el interior del edificio de DESITEL junto a la máquina que tiene la aplicación, se diseñó una caja de acrílico para evitar que alguien la desconecte o dañe la antena. La Figura IV.106 muestra como se ve el coordinador conectado a la pc.

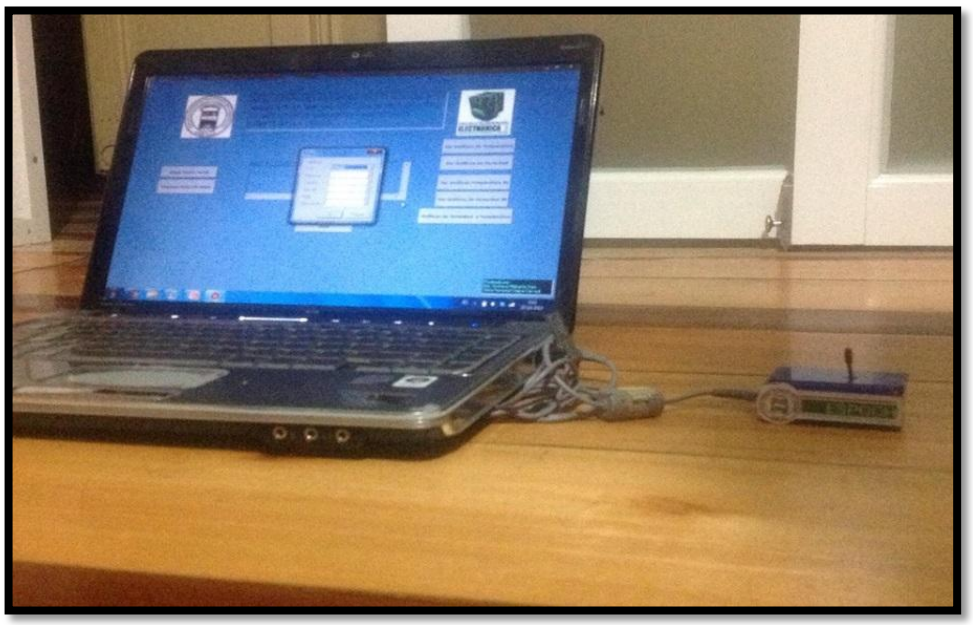

Figura IV.106: Dispositivo Coordinador

Fuente: Elaboración propia

Una vez montada la red ya empieza a tomar datos y se puede observar las variaciones del medio-ambiente a través de la aplicación.

Como se observa en la Figura IV.107 se tiene botones para elegir el puerto de donde se extraerán la información, para empezar a tomar los datos y otros para visualizar la graficas en 2D y 3D.

Figura IV.107. Aplicación Final

Fuente: Elaboración propia

En las Figuras IV. 108 y 109 se observa los datos de la temperatura y humedad de cada nodo respectivamente.

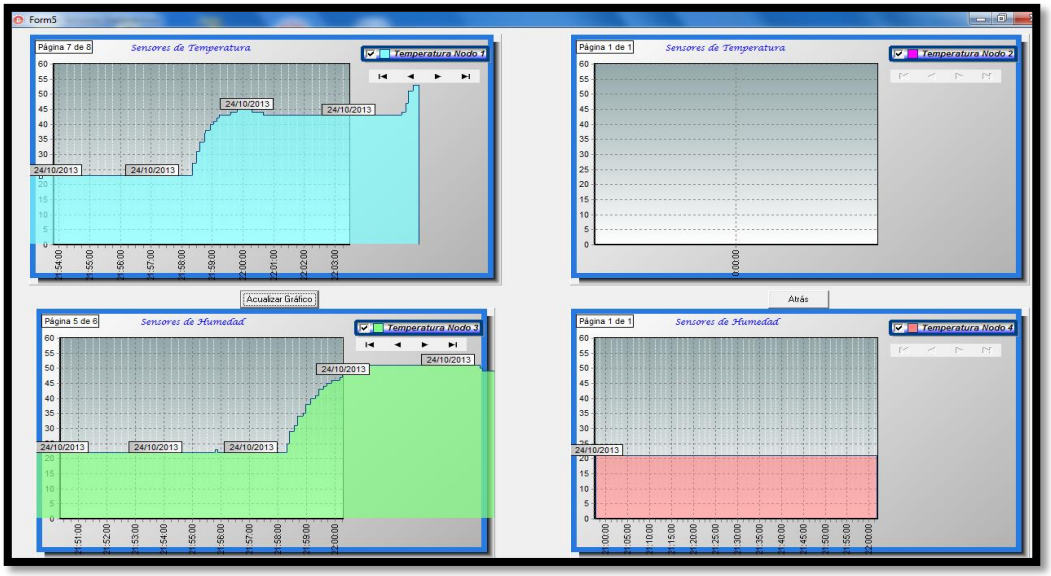

Figura IV.108: Graficas de Temperatura de los 4 Nodos

Fuente: Elaboración propia

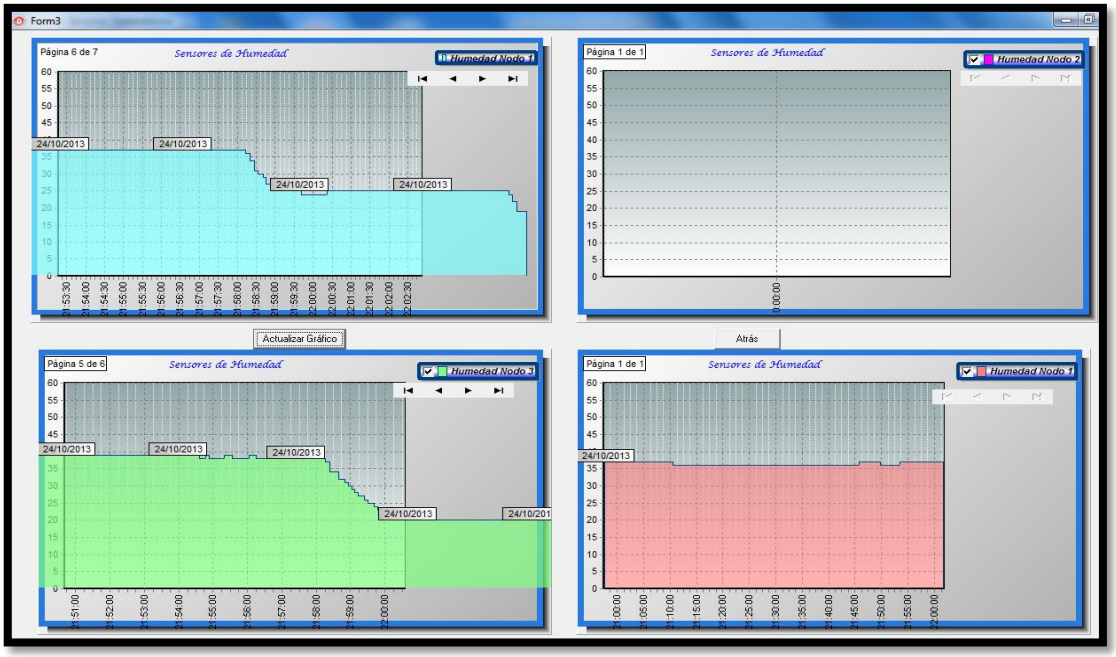

Figura IV.109: Graficas de Temperatura de los 4 Nodos

Fuente: Elaboración propia

CONCLUSIONES

- Al evaluar el desempeño y calidad de la Red WSN mediante métricas de evaluación, utilizando el simulador NS2, se obtuvo como resultado que la topología física adecuada para implementar una red inalámbrica de sensores medio ambientales es la topología estrella, que cuenta con un Packet Delivery Ratio del 99%, Throughput de 0,75 Kbps, un end to end delay de 0,016 ms, mayor área de cobertura y un consumo de energía similar en las dos topologías.

- Los resultados del NS2 en cuestión de energía nos indican que aproximadamente la batería de 9 V dura 18 días, sin embargo ya que para el protitipo se utilizo modulos arduino uno que consumen mucha energía la batería dura aproximadamente 3 o 4 días únicamente, pero en cambio permiten tener una gran gama de sensores debido a que cuentan con muchas entradas analógicas y digitales.

- Las redes de sensores inalámbricos son una solución frente a las tradicionales redes Wireless, debido a sus bajos precios y al bajo consumo energético, además este tipo de redes son muy eficientes ya

que son útiles para cualquier sensor que envie una señal de entre 0 V a 5 V, si algún sensor envía una señal diferente a este valor se puede acoplar la señal antes de enviar al sintema.

- Los módulos de comunicación XBEE S1 son adecuados para implementar una red WSN con topología estrella, y cuando los nodos finales se encuentran hasta 100 metros de distancia, si se requiere mayor distancia con la misma topología se debe optar por módulos de la serie pro 2 que pueden llegar hasta 7 km.

RECOMENDACIONES

- Ya que los archivos de trazas que se generan en el NS-2 son la base de las investigaciones se recomienda saber qué es lo que representa cada campo del archivo y usar AWK para el filtrado de los paquetes ya que en varias ocasiones este resulto mejor que cualquier herramienta gráfica.

- Se recomienda tener en cuenta la aplicación que se le va a dar a una WSN ya que dependiendo de sus características se podrá elegir la topología más adecuada para la implementación.

- Para que no exista interferencia entre Wi-Fi, Bluetooth y Zigbee que operan en la frecuencia libre de 2.4 GHz se recomienda configurar a zigbee en un canal diferente que Wi-Fi.

- Interconectar los proyectos de investigación del G.E.A.A.-ESPOCH, como son los sensores aplicados a la bioelectricidad, tubos al vacio y demás trabajos donde se necesite la medición constante de variables.

- Investigar sobre 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) que es un estándar que posibilita el uso de IPv6 sobre redes basadas en el estándar IEEE 802.15.4. Hace posible que dispositivos como los nodos de una red inalámbrica puedan comunicarse directamente con otros dispositivos IP.

BIBLIOGRAFÍA

1.- AKYILDIZ, I. AND VURAN, M., Wireless Sensor Networks (Advanced Texts In Communications And Networking)., 1a ed., New Jersey-EEUU., Ed. Mankrono Print Media., 2010., Pp. 413-502.

2.- BANZI, M., Introduccion a Arduino., 1a ed., Madrid-España., 2012., Ed. RC libros., Pp. 125-128p

3.- FALUDI, R., Building Wireless Sensor Networks., 1a ed., New York-Estados Unidos., Ed. BRIAN., 2010., 320p

4.- GISLASON, D., Zigbee Wireless Networking., Washington-Estados Unidos., Ed. Elsevier., 2008., Pp.400-413.

5.- AHMAD, K., Revista Automation in Construction., Wireless Sensor Networks as part of a Web-Based Building Environmental Monitoring System., Maryland- Estados Unidos., 2008., Pp 729–736.

BIBLIOGRAFÍA DE INTERNET

[1] **WIRELESS SENSORS NETWORK**

http://arcos.inf.uc3m.es/~sescolar/index_files/presentacion/wsn.pdf.
[2013/03/09]

[2] **ECUADOR WITH A WIRELESS SENSOR NETWORK**
http://www.eecs.harvard.edu/~mdw/proj/volcano/reventador-report.pdf.
[2013/04/09]

[3] **SENSORES INTELIGENTES: UNA HISTORIA CON FUTURO**
http://upcommons.upc.edu/revistes/bitstream/2099/9553/1/Article003.pdf.
[2013/04/10]

[4] **COMENZANDO CON ZIGBEE**
http://webdelcire.com/wordpress/archives/1714.

[2013/04/13]

[5] **ARDUINO**
http://www.arduino.cc/es/.

[2013/04/15]

[6] **INSTITUTO NACIONAL DE METEOROLOGÍA E HIDROLOGÍA**
http://www.inamhi.gob.ec/.

[2013/04/16]

[7] **DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE MONITORIZACIÓN Y DETECCIÓN DE ANOMALÍAS EN AMBIENTES CERRADOS, BASADO EN LA TECNOLOGÍA DE LAS REDES SENSORIALES INALÁMBRICAS (WSN)**
http://biblioteca.espoch.edu.ec/lista.

[2013/04/18]

[8] **ESTUDIO COMPARATIVO DE PROTOCOLOS DE RUTEO EN REDES AD-HOC APLICADO A REDES MÓVILES**
http://dspace.espoch.edu.ec/bitstream/123456789/642/1/38T00259.pdf.

[2013/04/25]

[9] **ESQUEMÁTICO DE CONEXIONADO Y MONTAJE DE MÓDULOS XBEE**
http://qubits.wordpress.com/2009/04/04/esquematico-de-conexionado-y-montaje-de-modulos-xbee/.

[2013/04/28]

[10] **NUEVO ESTÁNDAR DE TECNOLOGÍA INALÁMBRICA**
http://repositorio.espe.edu.ec/bitstream/21000/2995/1/T-ESPE-030912.pdf
[2013/04/31]

[11] **SENSOR NETWORK PROTOCOLS**
http://www.monografias.com/trabajos3/redcom/redcom.shtml

[2013/05/02]

[12] **HANDBOOK OF SENSORS NETWORK**
http://front.cc.nctu.edu.tw/Richfiles/13903-1968_fm.pdf

[2013/05/07]

[13] **REDES INALÁMBRICAS DE SENSORES**
http://riunet.upv.es/bitstream/handle/10251/8592/PFC%20-%20DESARROLLO%20DE%20APLICACIONES%20BASADAS%20EN%2N.pdf

[2013/05/08]

[14] **SENSORES**
http://definicion.de/sensor/.

[2013/05/10]

[15] **THE LAST LAB PROJECT**
http://thelastlabproject.blogspot.com/2010/12/clasificacion-de-los-sensores.html.
[2013/05/12]

[16] **REDES INALÁMBRICAS DE SENSORES TEORÍA Y APLICACIÓN**
http://dialnet.unirioja.es/descarga/libro/377564.pdf

[2013/05/12]

[17] **COMUNICACIONES INALAMBRICAS ZIGBEE,» ECUADOR QUITO 2006.**
http://clusterfie.epn.edu.ec/ibernal/html/CURSOS/Marzo07Agosto07/Actualizacion/ZigBeeNov06.pdf.

[2013/05/13]

[18] **CAMALIE NETWORKS WIRELESS SENSING**
http://camalie.com/WirelessSensing/WirelessSensors.htm.

[2013/05/4]

[19] **TUNGURAHUA SENSOR NETWORK DEPLOYMENT**
http://fiji.eecs.harvard.edu/Volcano.

[2013/05/14]

[20] **WSN(WIRELESS SENSOR NETWORK**
http://ingeniatic.euitt.upm.es/index.php/tecnologias/item/669-wsn-wireless-sensor-networks.

[2013/05/17]

[21] **TRACKSS TECHNOLOGIES**
http://www.trackss.net/technologies.html.

[2013/05/27]

[22] **APLICACIÓN DEL MODELADO ESPECÍFICO DE DOMINIO A LAS REDES DE SENSORES INALÁMBRICOS**
http://repositorio.bib.upct.es:8080/dspace/bitstream/10317/117/1/pfc2535.pdf.

[2013/06/02]

[23] **GUÍA DEL USUARIO XBEE SERIES 1**
http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf

[2013/08/24]

[24] **COMUNICACIÓN INALÁMBRICA USANDO MÓDULOS XBEE**
http://es.scribd.com/doc/147679800/58980339-Teoria-y-Programacion-Modulos-XBEE

[2013/08/25]

[25] **MCI ELECTRONICS**
http://www.olimex.cl/tutorials.php?page=tut_xbee.

[2013/09/03]

[26] **802.15.4 VS ZIGBEE. WIRELESS SENSOR NETWORKS RESEARCH GROUP**
http://www.sensornetworks.org/index.php?page=0823123150.

[2013/09/12]

[27] **Estándar IEEE 802.15.4**
http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/archundia_p_fm/capitulo4.pdf.

[2013/09/23]

[28] **ESTÁNDAR 802.11**
http://multingles.net/docs/Manual%20%20Redes%20WiFi%20inalambricas.pdf.

[2013/09/26]

[29] **ESTÁNDAR IEEE 802.15.1**
http://www.ieee802.org/15/pub/TG1.html.
[2013/09/28]

[30] **REDES DE ÁREA PERSONAL INALAMBRICA (WPAN)**
http://dspace.ups.edu.ec/bitstream/123456789/206/2/Capitulo%201.pdf.
[2013/10/01]

[31] **GUIA BÁSICA DE ARDUINO**
http://tdrobotica.co/tutoriales/81-arduino-2/345-guia-arduino
[2013/10/02]

[32] **MANUAL DE ARDUINO**
http://rua.ua.es/dspace/bitstream/10045/11833/1/arduino.pdf
[2013/10/02]

[33] **NS-2**
http://nsnam.isi.edu/nsnam/index.php/Main_Page
[2013/10/10]

[34] **DISEÑO DE EXPERIMENTOS AL COMPLETO AZAR**

http://reyesestadistica.blogspot.com/2011/07/diseno-de-experimentos-al-completo-azar.html

[35] **NAM (NETWORK ANIMATOR)**
http://www.isi.edu/nsnam/nam/
[2013/10/18]

[36] **PROBABILIDAD Y ESTADÍSTICA PARA INGENIEROS**

http://archuto.files.wordpress.com/2011/02/probabilidad_y_estadistica_basica.pdf
[2013/10/10]

RESUMEN

Se realizó la evaluación de las topologías físicas de Wireless Sensor Network (WSN) mediante la implementación de un prototipo de medición de variables ambientales.

La investigación se efectuó mediante el método inductivo, basado en la medición de las métricas de desempeño como son: envío de paquetes, consumo de energía y cobertura, luego de clasificar y analizar los datos se llegó a la conclusión del estudio.

Mediante el apoyo del software Network Simulator 2 (NS2) se evaluó las métricas antes mencionadas y con los resultados obtenidos se definió la mejor topología, posteriormente se implementó el prototipo de una red WSN para medir variables ambientales. Este contó con cuatro nodos finales y un nodo coordinador WSN basados en Arduino, un computador como gestor de datos y el software Arduino, Delphi 7, phpMyAdmin y NS2.

Como resultado se obtuvo que la topología física adecuada para implementar una red WSN para mediciones ambientales es la topología estrella, que cuenta con un Packet Delivery Ratio del 99%, Throughput de 0,7542, un retardo de 0,01621453 ms, un consumo de energía bajo y mayor área de cobertura.

Se concluye que la topología estrella es el escenario propicio para transmitir datos recogidos de los sensores en tiempo real logrando un buen desempeño de envío de paquetes, de consumo de energía y de cobertura.

Es recomendable saber previamente en qué área se va a implementar la red WSN para elegir la mejor topología física designada para cada caso.

ABSTRACT

Though the implementation of a prototype measurement of environmental variables it was made an evaluation of Wireless Sensor Network (WSN) physic topologies.

An inductive method was applied in the research, based on the performance metrics measurement such as: sending packages, energy consumption and coverage, the conclusion of the study was reached after sorting and analyzing the data.

The aforementioned metrics were evaluated by the support of Network Simulator 2 (NS2) and with the obtained results the best topology was defined. Later, a WSN network prototype was implemented to measure the environmental variables. It was counted with four end nodes and a coordinator node WSN based on Arduino, a computer as a data manager and the Arduino, Delphi 7, phpMyAdmin and NS2 software.

As a result it was obtained that the appropriate physical topology to implement a WSN network for environmental measurements was a start topology that has a Packet Delivery Ratio of 99%, Throughput of 0,754, a delay of 0,01621453 ms, a low energy consumption and a greater coverage area.

It is concluded that the star topology is the appropriate setting to transmit data collect from sensors in real time achieving a good performance of sending packages, energy consumption and coverage.

It is advisable to know in advance in which area the WSN network will be implemented to choose the best physical topology designated for each case.

ANEXOS

Estrella.tcl

```
#####################################################
#   Simulación de la topología Estrella             #
#      Basada en wpam_demo2.                        #
# - - - - - - - - - - - - - - - - - - - - - - - -   #
#       Prepared by Jianliang Zheng                 #
#        (zheng@ee.ccny.cuny.edu)                   #
#####################################################

# ======================================================================
# Define variables
# ======================================================================
set val(chan)               Channel/WirelessChannel         ;# Tipo de canal
set val(prop)               Propagation/TwoRayGround        ;# Modelo de radio Propagación
set val(netif)          Phy/WirelessPhy/802_15_4
set val(mac)            Mac/802_15_4
set val(ifq)            Queue/DropTail/PriQueue         ;# Tipo de interfaz queue
set val(ll)             LL                          ;# Tipoo de capa de enlace
set val(ant)            Antenna/OmniAntenna             ;# Modelo de antena
set val(ifqlen)         150                         ;# Máximo de paquetes en ifq
set val(nn)             11                          ;# Número de nodos
set val(rp)             AODV                        ;# Protocolo de enrutamiento
set val(x)          200                 ;#Área de cobertura
set val(y)          200
set val(MNcoverage)     200.0
set val(nam)        estrella.nam
set val(traffic)        cbr                         ;# Trafico UDP/CBR
proc SetPt { coverage } {
set Gt [Antenna/OmniAntenna set Gt_]
set Gr [Antenna/OmniAntenna set Gr_]
set ht [Antenna/OmniAntenna set Z_]
set hr [Antenna/OmniAntenna set Z_]
set RXThresh [Phy/WirelessPhy set RXThresh_]
set d4 [expr pow($coverage,4)]
set Pt [expr ($RXThresh*$d4)/($Gt*$Gr*$ht*$ht*$hr*$hr)]
return $Pt
}
Phy/WirelessPhy set Pt_ [SetPt $val(MNcoverage)] ;# asigna la potencia calculada


proc getCmdArgu {argc argv} {
    global val
    for {set i 0} {$i < $argc} {incr i} {
        set arg [lindex $argv $i]
        if {[string range $arg 0 0] != "-"} continue
        set name [string range $arg 1 end]
        set val($name) [lindex $argv [expr $i+1]]
    }
}
getCmdArgu $argc $argv
set appTime1        7.0  ;# en segundos
set appTime2        7.1  ;# en segundos
set appTime3        7.2  ;# en segundos
set appTime4        7.3  ;# en segundos
set appTime5        7.4  ;# en segundos
```

```
set appTime6            7.5  ;# en segundos
set appTime7            7.6  ;# en segundos
set appTime8            7.7  ;# en segundos
set appTime9            7.8  ;# en segundos
set appTime10           7.9 ;# en segundos
set stopTime           100  ;# en segundos
# Inicializar Variables Globales
set ns_         [new Simulator]
set tracefd     [open ./estrella.tr w]
$ns_ trace-all $tracefd

if { "$val(nam)" == "estrella.nam" } {
        set namtrace     [open ./$val(nam) w]
        $ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
    $ns_ use-newtrace
}
$ns_ puts-nam-traceall {# nam4wpan #}
Mac/802_15_4 wpanCmd verbose on
Mac/802_15_4 wpanNam namStatus on
# Modelo de propagación 'TwoRayGround'
set dist(5m)  7.69113e-06
set dist(9m)  2.37381e-06
set dist(10m) 1.92278e-06
set dist(11m) 1.58908e-06
set dist(12m) 1.33527e-06
set dist(13m) 1.13774e-06
set dist(14m) 9.81011e-07
set dist(15m) 8.54570e-07
set dist(16m) 7.51087e-07
set dist(20m) 4.80696e-07
set dist(25m) 3.07645e-07
set dist(30m) 2.13643e-07
set dist(35m) 1.56962e-07
set dist(40m) 1.20174e-07
set dist(50m) 7.69113e-08
set dist(75m) 3.41828e-08
set dist(100m) 1.42681e-08
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 0.002
Antenna/OmniAntenna set Gr_ 0.002
Phy/WirelessPhy set CSThresh_ $dist(15m)
Phy/WirelessPhy set RXThresh_ $dist(15m)
Phy/WirelessPhy set CPThresh_ 10
Phy/WirelessPhy set L_ 1.0 ;# Perdida por Trayectoria = 1.0
Phy/WirelessPhy set freq_ 2.4e+6


Phy/WirelessPhy set Pt_ [SetPt $val(MNcoverage)] ;# asigna la potencia trnsmitida a


set pottrans [SetPt $val(MNcoverage)]


puts "potencia de transmicion para cubrir $val(MNcoverage) metros es: $pottrans"


set god_ [create-god $val(nn)]
```

```
# Configuración de objeto  topography
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)

# Creación God
set god_ [create-god $val(nn)]
set chan_1_ [new $val(chan)]

# Configuración de características del nodo
$ns_ node-config -adhocRouting $val(rp) \
         -llType $val(ll) \
         -macType $val(mac) \
         -ifqType $val(ifq) \
         -ifqLen $val(ifqlen) \
         -antType $val(ant) \
         -propType $val(prop) \
         -phyType $val(netif) \
         -topoInstance $topo \
         -agentTrace ON \
         -routerTrace ON \
         -macTrace ON \
         -movementTrace OFF \
                  -energyModel "EnergyModel" \
                        -initialEnergy $pottrans \
                  -rxPower 0.5319 \
                        -txPower 0.2 \
              -sleepPower \
         -channel $chan_1_

$ns_ set WirelessNewTrace_ ON # Configuración nuevo formato de trazas
for {set i 0} {$i < $val(nn) } {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0      ;# disable random motion
}
#Ubicación de los nodos
$node_(0) set X_ 25
$node_(0) set Y_ 25
$node_(0) set Z_ 0
$node_(1) set X_ 25
$node_(1) set Y_ 14
$node_(1) set Z_ 0
$node_(2) set X_ 25
$node_(2) set Y_ 36
$node_(2) set Z_ 0
$node_(3) set X_ 14.5
$node_(3) set Y_ 21.6
$node_(3) set Z_ 0
$node_(4) set X_ 35.5
$node_(4) set Y_ 28.4
$node_(4) set Z_ 0
$node_(5) set X_ 35.5
$node_(5) set Y_ 21.6
$node_(5) set Z_ 0
$node_(6) set X_ 14.5
$node_(6) set Y_ 28.4
$node_(6) set Z_ 0
$node_(7) set X_ 31.5
$node_(7) set Y_ 16.1
```

```
$node_(7) set Z_ 0
$node_(8) set X_ 18.5
$node_(8) set Y_ 33.9
$node_(8) set Z_ 0
$node_(9) set X_ 18.5
$node_(9) set Y_ 16.1
$node_(9) set Z_ 0
$node_(10) set X_ 31.5
$node_(10) set Y_ 33.9
$node_(10) set Z_ 0
$ns_ at 0.0  "$node_(0) NodeLabel Coordinador"
$ns_ at 0.0  "$node_(0) sscs startPANCoord";#startPANCoord <txBeacon=1> <BO=3> <SO=3>
$ns_ at 0.5  "$node_(1) sscs startDevice 1 0" ;# startDevice <isFFD=1> <assoPermit=1>
<txBeacon=0> <BO=3> <SO=3>
$ns_ at 1.5  "$node_(2)  sscs startDevice 1 0"
$ns_ at 2.5  "$node_(3)  sscs startDevice 1 0"
$ns_ at 3.5  "$node_(4)  sscs startDevice 1 0"
$ns_ at 4.5  "$node_(5)  sscs startDevice 1 0"
$ns_ at 5.5  "$node_(6)  sscs startDevice 1 0"
$ns_ at 3.5  "$node_(7)  sscs startDevice 1 0"
$ns_ at 4.5  "$node_(8)  sscs startDevice 1 0"
$ns_ at 5.5  "$node_(9)  sscs startDevice 1 0"
$ns_ at 6.5  "$node_(10) sscs startDevice 1 0"
Mac/802_15_4 wpanNam PlaybackRate 3ms
$ns_ at $appTime1 "puts \"\nTransmisión de datos ...\n\""


# Configuración de flujo de tráfico entre nodods
proc cbrtraffic { src dst interval starttime } {
   global ns_ node_
   set udp_($src) [new Agent/UDP]
   eval $ns_ attach-agent \$node_($src) \$udp_($src)
   set null_($dst) [new Agent/Null]
   eval $ns_ attach-agent \$node_($dst) \$null_($dst)
   set cbr_($src) [new Application/Traffic/CBR]
   eval \$cbr_($src) set packetSize_ 150
   eval \$cbr_($src) set interval_ $interval
   eval \$cbr_($src) set random_ 0
   eval \$cbr_($src) set maxpkts_ 10000
   eval \$cbr_($src) attach-agent \$udp_($src)
   eval $ns_ connect \$udp_($src) \$null_($dst)
   $ns_ at $starttime "$cbr_($src) start"
}
if { ("$val(traffic)" == "cbr") || ("$val(traffic)" == "poisson") } {
   puts "\nTraffic: $val(traffic)"
   #Mac/802_15_4 wpanCmd ack4data on
   puts [format "Acknowledgement for data: %s" [Mac/802_15_4 wpanCmd ack4data]]
   $ns_ at $appTime1 "Mac/802_15_4 wpanNam PlaybackRate 0.5ms"
   $ns_ at [expr $appTime1 + 0.5] "Mac/802_15_4 wpanNam PlaybackRate 1.5ms"
   $val(traffic)traffic 1  0 0.6  $appTime1
   $val(traffic)traffic 2  0 0.6  $appTime2
   $val(traffic)traffic 3  0 0.6  $appTime3
   $val(traffic)traffic 4  0 0.6  $appTime4
   $val(traffic)traffic 5  0 0.6  $appTime5
   $val(traffic)traffic 6  0 0.6  $appTime6
   $val(traffic)traffic 7  0 0.6  $appTime7
   $val(traffic)traffic 8  0 0.6  $appTime8
   $val(traffic)traffic 9  0 0.6  $appTime9
```

```
   $val(traffic)traffic 10 0 0.6  $appTime10
   $ns_ at $appTime1   "$ns_ trace-annotate \"(at $appTime1)   $val(traffic) tráfico desde nodo 1
al nodo 0\""
   $ns_ at $appTime2   "$ns_ trace-annotate \"(at $appTime2)   $val(traffic) tráfico desde nodo 2
al nodo 0\""
   $ns_ at $appTime3   "$ns_ trace-annotate \"(at $appTime3)   $val(traffic) tráfico desde nodo 3
al nodo 0\""
   $ns_ at $appTime4   "$ns_ trace-annotate \"(at $appTime4)   $val(traffic) tráfico desde nodo 4
al nodo 0\""
   $ns_ at $appTime5   "$ns_ trace-annotate \"(at $appTime5)   $val(traffic) tráfico desde nodo 5
al nodo 0\""
   $ns_ at $appTime6   "$ns_ trace-annotate \"(at $appTime6)   $val(traffic) tráfico desde nodo 6
al nodo 0\""
   $ns_ at $appTime7   "$ns_ trace-annotate \"(at $appTime7)   $val(traffic) tráfico desde nodo 7
al nodo 0\""
   $ns_ at $appTime8   "$ns_ trace-annotate \"(at $appTime8)   $val(traffic) tráfico desde nodo 8
al nodo 0\""
   $ns_ at $appTime9   "$ns_ trace-annotate \"(at $appTime9)   $val(traffic) tráfico desde nodo 9
al nodo 0\""
   $ns_ at $appTime10  "$ns_ trace-annotate \"(at $appTime10)  $val(traffic) tráfico desde nodo
10 al nodo 0\""
   Mac/802_15_4 wpanNam FlowClr -p AODV -c tomato
   Mac/802_15_4 wpanNam FlowClr -p ARP -c green
   Mac/802_15_4 wpanNam FlowClr -p MAC -s 0 -d -1 -c navy
   if { "$val(traffic)" == "cbr" } {
     set pktType cbr
   } else {
     set pktType exp
   }
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 1  -d 0 -c blue
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 2  -d 0 -c green4
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 3  -d 0 -c cyan4
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 4  -d 0 -c red
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 5  -d 0 -c green
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 6  -d 0 -c blue
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 7  -d 0 -c green4
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 8  -d 0 -c cyan4
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 9  -d 0 -c red
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 10 -d 0 -c green
}

# Definición del tamaño de los nodos en el nam
for {set i 0} {$i < $val(nn)} {incr i} {
    $ns_ initial_node_pos $node_($i) 2
}
# Configurar a los nodos para el fin del proceso
for {set i 0} {$i < $val(nn) } {incr i} {
   $ns_ at $stopTime "$node_($i) reset";
}
$ns_ at $stopTime "stop"
$ns_ at $stopTime "puts \"NS Saliendo...\n\""
$ns_ at $stopTime "$ns_ halt"
proc stop {} {
   global ns_ tracefd appTime1 val env
   $ns_ flush-trace
   close $tracefd
   set hasDISPLAY 0
   foreach index [array names env] {
```

```
        #puts "$index: $env($index)"
        if { ("$index" == "DISPLAY") && ("$env($index)" != "") } {
                set hasDISPLAY 1
        }
    }
    if { ("$val(nam)" == "estrella.nam") && ("$hasDISPLAY" == "1") } {
     exec nam estrella.nam &
    }
}

puts "\nComenzando Simulación..."
$ns_ run
```

Tree.tcl

```
#################################################
#Simulación de Topolog[ia Árbol basada en       #
# el ejemplo wpan_demos tres propio de NS2      #
#      Copyright (c) 2003 Samsung/CUNY          #
#        Prepared by Jianliang Zheng            #
#         (zheng@ee.ccny.cuny.edu)              #
#################################################
#=================================================================
# Define opciones
#=================================================================
set val(chan)           Channel/WirelessChannel    ;# Tipo de canal
set val(prop)           Propagation/TwoRayGround   ;# Modelo de radio-propagación
set val(netif)          Phy/WirelessPhy/802_15_4
set val(mac)            Mac/802_15_4
set val(ifq)            Queue/DropTail/PriQueue    ;# Tipo de interfaz queue
set val(ll)             LL                         ;# Tipoo de capa de enlace
set val(ant)            Antenna/OmniAntenna        ;# Modelo de antena
set val(ifqlen)         150                        ;# Máximo de paquetes en ifq
set val(nn)             11                         ;# Número de nodos
set val(rp)             AODV                           ;# routing protocol
set val(x)      200                 ;#Área de cobertura
set val(y)      200
set val(nam)      tree.nam
set val(traffic)        cbr                        ;# Tráfico UDP/CBR
proc getCmdArgu {argc argv} {
        global val
        for {set i 0} {$i < $argc} {incr i} {
                set arg [lindex $argv $i]
                if {[string range $arg 0 0] != "-"} continue
                set name [string range $arg 1 end]
                set val($name) [lindex $argv [expr $i+1]]
        }
}
getCmdArgu $argc $argv
set appTime1            8.1;# in seconds
set appTime2            8.3;# in seconds
set appTime3            8.5;# in seconds
set appTime4            8.7;# in seconds
set appTime5            8.9;# in seconds
set stopTime           100  ;# in seconds
# Inicio de variables Globales
set ns_         [new Simulator]
set tracefd     [open tree.tr w]
```

```
$ns_ trace-all $tracefd
if { "$val(nam)" == "tree.nam" } {
        set namtrace     [open ./$val(nam) w]
        $ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
     $ns_ use-newtrace
}
$ns_ puts-nam-traceall {# nam4wpan #}        ;# inform nam that this is a trace file for wpan (special handling needed)
Mac/802_15_4 wpanCmd verbose on
Mac/802_15_4 wpanNam namStatus on        ;# default = off (should be turned on before other 'wpanNam' commands can work)
# Modelo de Radio/propagación 'TwoRayGround'
set dist(5m)  7.69113e-06
set dist(9m)  2.37381e-06
set dist(10m) 1.92278e-06
set dist(11m) 1.58908e-06
set dist(12m) 1.33527e-06
set dist(13m) 1.13774e-06
set dist(14m) 9.81011e-07
set dist(15m) 8.54570e-07
set dist(16m) 7.51087e-07
set dist(20m) 4.80696e-07
set dist(25m) 3.07645e-07
set dist(30m) 2.13643e-07
set dist(35m) 1.56962e-07
set dist(40m) 1.20174e-07
set dist(50m) 7.69113e-08
set dist(75m) 3.41828e-08
set dist(100m) 1.42681e-08
Phy/WirelessPhy set CSThresh_ $dist(15m)
Phy/WirelessPhy set RXThresh_ $dist(15m)
Phy/WirelessPhy set CPThresh_ 10
Phy/WirelessPhy set L_ 1.0 ;# Perdida por Trayectoria = 1.0
# Configuraciónde objeto topography
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)

# Creación del God
set god_ [create-god $val(nn)]
set chan_1_ [new $val(chan)]

# Configuración de características del nodo
$ns_ node-config -adhocRouting $val(rp) \
         -llType $val(ll) \
         -macType $val(mac) \
         -ifqType $val(ifq) \
         -ifqLen $val(ifqlen) \
         -antType $val(ant) \
         -propType $val(prop) \
         -phyType $val(netif) \
         -topoInstance $topo \
         -agentTrace ON \
         -routerTrace ON \
         -macTrace ON \
         -movementTrace OFF \
                  -energyModel "EnergyModel"\
             -initialEnergy 5760 \
         -txPower 0.2 \
```

```
        -rxPower 0.05319 \
        -idlePower 0.00001377 \
        -sleepPower 0.1 \
        -channel $chan_1_
#Configuración de nuevo formato de trzas
$ns_ set WirelessNewTrace_ ON
for {set i 0} {$i < $val(nn) } {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0      ;#
}
#Ubicación de los nodos
$node_(0) set X_ 30
$node_(0) set Y_ 40
$node_(0) set Z_ 0
$node_(1) set X_ 22
$node_(1) set Y_ 32
$node_(1) set Z_ 0
$node_(2) set X_ 35
$node_(2) set Y_ 32
$node_(2) set Z_ 0
$node_(3) set X_ 13.5
$node_(3) set Y_ 24
$node_(3) set Z_ 0
$node_(4) set X_ 29
$node_(4) set Y_ 24
$node_(4) set Z_ 0
$node_(5) set X_ 40
$node_(5) set Y_ 23
$node_(5) set Z_ 0
$node_(6) set X_ 9
$node_(6) set Y_ 14
$node_(6) set Z_ 0
$node_(7) set X_ 16.5
$node_(7) set Y_ 13
$node_(7) set Z_ 0
$node_(8) set X_ 24
$node_(8) set Y_ 13
$node_(8) set Z_ 0
$node_(9) set X_ 34
$node_(9) set Y_ 13.5
$node_(9) set Z_ 0
$node_(10) set X_ 44
$node_(10) set Y_ 13
$node_(10) set Z_ 0
$ns_ at 0.0  "$node_(0) NodeLabel Coordinador"
$ns_ at 0.0  "$node_(0) sscs startPANCoord 1"      ;#  startPANCoord  <txBeacon=1>  <BO=3>
<SO=3>
$ns_ at 0.5  "$node_(1) sscs startDevice 1 1 1"      ;#  startDevice  <isFFD=1>  <assoPermit=1>
<txBeacon=0> <BO=3> <SO=3>
$ns_ at 1.5  "$node_(2) sscs startDevice 1 1 1"
$ns_ at 2.5  "$node_(3) sscs startDevice 1 1 1"
$ns_ at 3.5  "$node_(4) sscs startDevice 1 1 1"
$ns_ at 4.5  "$node_(5) sscs startDevice 1 1 1"
$ns_ at 5.5  "$node_(6) sscs startDevice 0"
$ns_ at 5.8  "$node_(7) sscs startDevice 0"
$ns_ at 6.5  "$node_(8) sscs startDevice 0"
$ns_ at 6.8  "$node_(9) sscs startDevice 0"
$ns_ at 7.0  "$node_(10) sscs startDevice 0"
```

```
$ns_ at 6.0 "$node_(3) sscs stopBeacon"
$ns_ at 8.0 "$node_(3) sscs startBeacon"
$ns_ at 9.0 "$node_(5) sscs startBeacon 4 4"        ;# change beacon order and superframe order
$ns_ at 10.0 "$node_(4) sscs stopBeacon"
Mac/802_15_4 wpanNam PlaybackRate 3ms
$ns_ at $appTime1 "puts \"\nTransmitting data ...\n\""
# Configuración de flujo de tráfico entre nodods
proc cbrtraffic { src dst interval starttime } {
   global ns_ node_

   set udp_($src) [new Agent/UDP]
   eval $ns_ attach-agent \$node_($src) \$udp_($src)

   set null_($dst) [new Agent/Null]
   eval $ns_ attach-agent \$node_($dst) \$null_($dst)

   set cbr_($src) [new Application/Traffic/CBR]
   eval \$cbr_($src) set packetSize_ 150
   eval \$cbr_($src) set interval_ $interval
   eval \$cbr_($src) set random_ 0
   eval \$cbr_($src) set maxpkts_ 10000
   eval \$cbr_($src) attach-agent \$udp_($src)
   eval $ns_ connect \$udp_($src) \$null_($dst)
   $ns_ at $starttime "$cbr_($src) start"
}
if { ("$val(traffic)" == "cbr") } {
   puts "\nTraffic: $val(traffic)"
   #Mac/802_15_4 wpanCmd ack4data on
   puts [format "Acknowledgement for data: %s" [Mac/802_15_4 wpanCmd ack4data]]
   $ns_ at $appTime1 "Mac/802_15_4 wpanNam PlaybackRate 0.50ms"
   $ns_ at [expr $appTime1 + 0.5] "Mac/802_15_4 wpanNam PlaybackRate 1.5ms"
   $val(traffic)traffic 6  0 0.2 $appTime1
   $val(traffic)traffic 7  0 0.2 $appTime2
   $val(traffic)traffic 8  0 0.2 $appTime3
   $val(traffic)traffic 9  0 0.2 $appTime4
   $val(traffic)traffic 10 0 0.2 $appTime5
   $ns_ at $appTime1 "$node_(6) add-mark m1 blue circle"
   $ns_ at $appTime1 "$node_(0) add-mark m2 blue circle"
   $ns_ at $appTime1 "$ns_ trace-annotate \"(at $appTime1) $val(traffic) traffic from node 6 to node 0\""
   $ns_ at $appTime2 "$node_(7) add-mark m3 yellow circle"
   $ns_ at $appTime2 "$node_(0) add-mark m4 yellow circle"
   $ns_ at $appTime2 "$ns_ trace-annotate \"(at $appTime2) $val(traffic) traffic from node 7 to node 0\""
   $ns_ at $appTime3 "$node_(8) add-mark m5 brown circle"
   $ns_ at $appTime3 "$node_(0) add-mark m6 brown circle"
   $ns_ at $appTime3 "$ns_ trace-annotate \"(at $appTime3) $val(traffic) traffic from node 8 to node 0\""
   $ns_ at $appTime4 "$node_(9) add-mark m7 red circle"
   $ns_ at $appTime4 "$node_(0) add-mark m8 red circle"
   $ns_ at $appTime4 "$ns_ trace-annotate \"(at $appTime4) $val(traffic) traffic from node 1 to node 6\""
   $ns_ at $appTime5 "$node_(10) add-mark m9 green4 circle"
   $ns_ at $appTime5 "$node_(0) add-mark m10 green4 circle"
   $ns_ at $appTime5 "$ns_ trace-annotate \"(at $appTime5) $val(traffic) traffic from node 10 to node 0\""
```

```
   Mac/802_15_4 wpanNam FlowClr -p AODV -c tomato
   Mac/802_15_4 wpanNam FlowClr -p ARP -c green
   Mac/802_15_4 wpanNam FlowClr -p MAC -c navy
   if { "$val(traffic)" == "cbr" } {
     set pktType cbr
   } else {
     set pktType exp
   }
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 6  -d 0 -c blue
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 7  -d 0 -c green4
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 8  -d 0 -c red
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 9  -d 0 -c navy
   Mac/802_15_4 wpanNam FlowClr -p $pktType -s 10 -d 0 -c tomato
}
# Definición del tama;o de los nodos en el nam
for {set i 0} {$i < $val(nn)} {incr i} {
     $ns_ initial_node_pos $node_($i) 2
}
# Configurar a los nodos para el fin del proceso
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $stopTime "$node_($i) reset";
}
$ns_ at $stopTime "stop"
$ns_ at $stopTime "puts \"\nNS Finalizando...\n\""
$ns_ at $stopTime "$ns_ halt"
proc stop {} {
    global ns_ tracefd appTime val env
    $ns_ flush-trace
    close $tracefd
    set hasDISPLAY 0
    foreach index [array names env] {
        #puts "$index: $env($index)"
        if { ("$index" == "DISPLAY") && ("$env($index)" != "") } {
                set hasDISPLAY 1
        }
    }
    if { ("$val(nam)" == "tree.nam") && ("$hasDISPLAY" == "1") } {
     exec nam tree.nam &
    }
}
puts "\nComenzando Simulation..."
$ns_ run
```

Estadisticas.awk

```
BEGIN {
print("\n\n******** Estadísticas de la Topologia Árbol********\n");

#Se debe poner el número de nodos dentro del corchetes
packet_sent[11] = 0;
packet_drop[11] = 0;
packet_recvd[11] = 0;
packet_forwarded[11] = 0;
# Se debe poner la energía inicial de cada nodo en Julios
energy_left[11] = 5760;
total_pkt_sent=0;
total_pkt_recvd=0;
total_pkt_drop=0;
total_pkt_forwarded=0;
pkt_delivery_ratio = 0;
total_hop_count = 0;
avg_hop_count = 0;
overhead = 0;
start = 0.000000000;
end = 0.000000000;
packet_duration = 0.0000000000;
recvnum = 0;
delay = 0.000000000;
sum = 0.000000000;
i=0;
total_energy_consumed = 0.000000;
}
#El signo “$” hace referencia a el número de columna del archivo trace
{
state       =   $1;
time        =   $3;
node_num    =   $5;
energy_level     =   $7;
node_id     =   $9;
level       =   $19;
pkt_type    =   $35;
packet_id     =   $41;
no_of_forwards      =   $49;
# Conteo de los paquetes en la red
if((pkt_type == "cbr") && (state == "s") && (level=="AGT")) {
    for(i=0;i<11;i++) {
        if(i == node_id) {
        packet_sent[i] = packet_sent[i] + 1; }
}
}else if((pkt_type == "cbr") && (state == "r") && (level=="AGT")) {
    for(i=0;i<11;i++) {
        if(i == node_id) {
        packet_recvd[i] = packet_recvd[i] + 1; }
}
}else if((pkt_type == "cbr") && (state == "d")) {
    for(i=0;i<11;i++) {
        if(i == node_id) {
        packet_drop[i] = packet_drop[i] + 1; }
}
}else if((pkt_type == "cbr") && (state == "f")) {
    for(i=0;i<11;i++) {
```

```
        if(i == node_id) {
        packet_forwarded[i] = packet_forwarded[i] + 1; }
}
}
# Cálculo del promedio End to End Delay

if (( state == "s") &&  ( pkt_type == "cbr" ) && ( level == "AGT" ))  { start_time[packet_id] = time; }
 if (( state == "r") &&  ( pkt_type == "cbr" ) && ( level == "AGT" )) {  end_time[packet_id] = time;
}
 else {  end_time[packet_id] = -1;  }
if(state == "N") {
    for(i=0;i<11;i++) {
        if(i == node_num) {
                    energy_left[i] = energy_left[i] - (energy_left[i] - energy_level);
                }
            }
}
}
# In this for loop also change
END {
for(i=0;i<11;i++) {
#Se guarda la información de los paquetes de cada nodo en archicos .txt
printf("%d %d \n",i, packet_sent[i]) > "pktsent.txt";
printf("%d %d \n",i, packet_recvd[i]) > "pktrecvd.txt";
printf("%d %d \n",i, packet_drop[i]) > "pktdrop.txt";
printf("%d %d \n",i, packet_forwarded[i]) > "pktfwd.txt";
printf("%d %.6f \n",i, energy_left[i]) > "energyleft.txt";
total_pkt_sent = total_pkt_sent + packet_sent[i];
total_pkt_recvd = total_pkt_recvd + packet_recvd[i];
total_pkt_drop = total_pkt_drop + packet_drop[i];
total_pkt_forwarded = total_pkt_forwarded + packet_forwarded[i];
total_energy_consumed = total_energy_consumed + energy_left[i];
}
printf("Total Pquetes Enviados       :   %d\n",total_pkt_sent);
printf("Total Packets Recividos      :   %d\n",total_pkt_recvd);
printf("Total Packets Desechados     :   %d\n",total_pkt_drop);
printf("Total Packets Reenviados     :   %d\n", total_pkt_forwarded);
pkt_delivery_ratio = (total_pkt_recvd/total_pkt_sent)*100;
printf("Packet Delivery Ratio        :   %d\n",pkt_delivery_ratio);
printf("Througphut de la red (KBps):   %.4f\n", ((total_pkt_recvd/1000)*512)/1024);
# For End to End Delay
for ( i in end_time ) {
 start = start_time[i];
 end = end_time[i];
 packet_duration = end - start;
 if ( packet_duration > 0 )  { sum += packet_duration; recvnum++; }
}
 delay=sum/recvnum;
printf("Promedio End-to-End Delay     :%.9f ms\n", delay);
printf("Total Energy Consumida        :%.6f\n", (11*10000)-total_energy_consumed);

#printf("Protocol Energy Consumption :%.6f\n",                                    100.000000-
((total_energy_consumed/(11*10000.000000))*100.000000));
if(((total_pkt_recvd + total_pkt_drop)/total_pkt_sent)==1) {
printf("Cálculo de estadística Completa !!!");
}
}
```

SUNROM TECHNOLOGIES

For more products visit our website http://www.sunrom.com

Document: Datasheet Date: 20-Jun-12 Model #: 3732 Product's Page: www.sunrom.com/p-1141.html

DHT11 - Humidity and Temperature Sensor

The DHT11 is a basic, low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed).

Its fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is you can only get new data from it once every 2 seconds.

Features

- Full range temperature compensated
- Relative humidity and temperature measurement
- Calibrated digital signal
- Outstanding long-term stability
- Extra components not needed
- Long transmission distance
- Low power consumption
- 4 pins packaged and fully interchangeable

Details

This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high-performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness. Each DHT11 element is strictly calibrated in the laboratory that is extremely accurate on humidity calibration. The calibration coefficients are stored as programmes in the OTP memory, which are used by the sensor's internal signal detecting process.
The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and up-to-20 meter signal transmission making it the best choice for various applications, including those most demanding ones. The component is 4-pin single row pin package.

Pin Signals

Figure 1-03. XBee®/XBee-PRO® RF Module Pin Numbers

(top sides shown - shields on bottom)

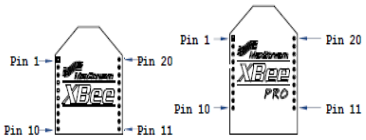

Table 1-02. Pin Assignments for the XBee and XBee-PRO Modules
(Low-asserted signals are distinguished with a horizontal line above signal name.)

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / **CONFIG**	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	**RESET**	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	DTR / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	**CTS** / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / **SLEEP**	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	**RTS** / AD6 / DIO6	Either	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

* Function is not supported at the time of this release

Design Notes:

- Minimum connections: VCC, GND, DOUT & DIN
- Minimum connections for updating firmware: VCC, GND, DIN, DOUT, RTS & DTR
- Signal Direction is specified with respect to the module
- Module includes a 50k Ω pull-up resistor attached to **RESET**
- Several of the input pull-ups can be configured using the PR command
- Unused pins should be left disconnected