CAPITULO 1

INTRODUCCION A WIRELESS SENSOR NETWORKS

1.1 INTRODUCCION

Los grandes avances de la tecnología en conjunto con las necesidades de un mundo que busca permanentemente el progreso han impulsado una forma de percibir los distintos "sabores" de la naturaleza, tan importantes dentro del área laboral, industrial y tecnológica de la humanidad, de manera menos dependiente de un medio físico, como lo es el cableado, además, por su puesto de la necesidad de mantener todo en comunicación mutua, a través de una red.

En definitiva una red sensorial inalámbrica (WSN) consiste en un gran número de nodos desplegados en un área determinada e integrados para colaborar en la misma red. La necesidad de una distribución inteligente y adecuada de una red sensorial lo suficientemente capaz de trabajar con cada uno de los parámetros necesarios dentro de un medio determinado, tiene grandes metas alcanzadas gracias a los avances tecnológicos de los medios y principios inalámbricos, metas tales como la reducción de los costos en equipos de procesamiento de señales esto a causa de que las plataformas de sensores ya no necesitan la misma capacidad de procesamiento individualizada distribuida en tuberías llenas de cableado. Además, la susceptibilidad a fallas se observa considerablemente debido a que los sistemas de procesamiento y de red se encuentran entre sensores.

También es muy importante reconocer la tremenda ventaja de la flexibilidad e independencia de los sistemas inalámbricos al momento de instalarlos en un espacio físico determinado.

Sobre todo, las distintas aplicaciones del entorno humano, en ambientes adversos como el área de la milicia, investigación o manufactura, son destinatarios esenciales

para el consumo de esta tecnología inalámbrica para el sensamiento de distintas variables necesarias para la producción.

1.2 ANTECEDENTES Y ACTUALIDAD

Teniendo como primordial causa la necesidad de una reducción de costos acompañada de una optimización en la tecnología de recepción, procesamiento y manejo de señales o variables en el entorno productivo, se puede ser partícipe de cómo ha ido evolucionando este tipo de tecnología hasta nuestros días. Desde las pruebas realizadas en un laboratorio, con un radar virtual con frecuencias de ultra-anda ancha, alcanzando velocidades de transmisión de por encima de 1 Gbits/seg.

Velocidades que ahora están siendo superadas en redes y sistemas inalámbricos y de fibra óptica, todo por medio de una distribución de monitoreo y análisis computarizado. Así, en conjunto con el aumento de las capacidades del Protocolo IP dentro del sector comercial, la milicia está comenzando a empatar su formato IP para hacerlas compatibles con el sistema comercial, adaptando los sistemas a las muy altas velocidades alcanzadas y permitiendo una armonía aplicativa no solo para la industria sino para la satisfacción de necesidades básicas y nada despreciables de la vida diaria de los seres humanos.

La capacidad de poder distribuir el procesamiento en una arquitectura sensorial, ofrece un alto grado de robustez en la calidad y necesidad de la red misma. Permitiendo aplicaciones muy importantes y necesarias en el presente, tales como:

- Geo-localización
- Identificación de Objetivos Móviles

A diferencia de las redes sensoriales cableadas, una inalámbrica deberá contener en su estructura potencialmente largos retardos de propagación punto-a-punto, esto en cuanto al flujo de control y enrutamiento.

Específicamente, los sistemas de red sensorial móvil o inalámbrica usan dos de los tipos de enrutamiento, tanto pro-activo como reactivo. Es decir, en el primer caso, el *pro-activo* permite la distribución de rutas en todas direcciones, mientras que en el caso *reactivo*, se ofrece solamente frente a una demanda.

Por supuesto, no todo brilla en cuanto a las redes sensoriales inalámbricas, de acuerdo al protocolo que usan, la tasa de BIT de error (BER) es cuatro o cinco veces mayor al de las redes cableadas, dato que puede afectar en su interpretación original como la de confundir dicha tasa con congestión o desconexión de la red misma.

1.3 VENTAJAS DE APLICACIÓN

Como hemos podido ser participes ya de la primera introducción acerca de lo importantes y aplicativas que pueden ser las redes de este tipo, que son el resultado de la convergencia de tres tecnologías esencialmente:

1.Circuitos Digitales;
2.Comunicaciones Inalámbricas; y,
3.Nodos de Micro Sistemas Electromecánicos (MEMS).

En la mayoría de las aplicaciones de este tipo de tecnología requerirán de gran cantidad de dispositivos, tantos como en el orden de decenas de cientos de nodos, justamente por la simplicidad que representa su instalación ,mantenimiento y manejo al momento de trabajar con ellos, debido a que las redes tradicionales representan una extensa complejidad, no solo en su instalación sino también en su mantenimiento, además de los altos costos por la cantidad de material necesarios dentro de la infraestructura implementada, teniendo en cuenta cableados, equipos de conmutación, puentes rieles y otros presentes en su estructura.

Definitivamente, la introducción de las redes Inalámbricas de Sensores, traen consigo una amplia gama de ventajas al mercado y a la producción, tales como:

- Mejor y Mayor Cobertura,

- Mayor Resolución,
- Alta Tolerancia a Fallas,
- Alta Robustez,
- Entre otros.

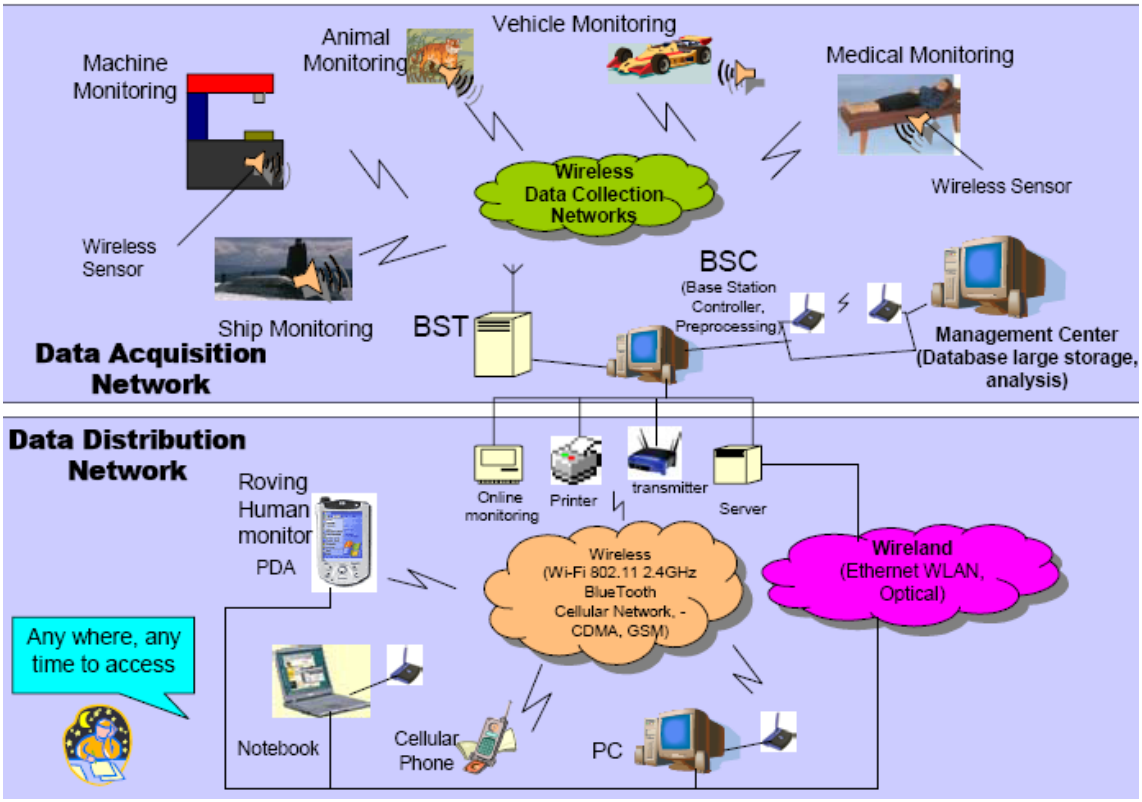


Figura 1.1: Esquema de una Red Sensorial Inalámbrica

Capacidades y virtudes que son actualmente aprovechadas por sectores como la milicia para la identificación y/o prevención de fenómenos de carácter ambiental, catástrofes, desastres, zonas de riesgo, contaminación, y más. Drásticamente, el uso y aprovechamiento de la carga computacional en aplicaciones y administración de recursos dentro de la tasa de producción y reducción de riesgos.

Justamente, la administración, monitoreo y control de las WSN's deben ser de tipo autónoma, es decir, que para el efecto de corrección e independencia deben ser auto-organizado, auto-desinfectantes, auto-optimizadoras, auto-sustentables, auto-diagnosticables; con el mínimo de interferencia humana.

Es muy importante destacar que los sistemas de este tipo mantienen su robustez y calidad de servicio frente a los posibles cambios del sistema sensorial.

La meta más altas a ser alcanzada, en el campo de la administración de las Redes Sensoriales Inalámbricas, es la formación de un sistema de integración global promoviendo la productividad, al margen de mantener la calidad del servicio prestado en la Red.

Una WSN es una herramienta distribuida para sensar uno o varios fenómenos que produce un reporte de los datos a un observador o monitor específico. Por lo tanto, la importancia de la implementación de este tipo de tecnología ofrece la posibilidad de sensar, monitorear y en algunos de los casos, hasta controlar ambientes lejanos de la ubicación de observación, es decir, remotos.

La energía es un recurso de carácter crítico en las Redes de este tipo, por lo tanto, su abastecimiento debe ser primordial durante la operación, para su completo y correcto funcionamiento.

Otra ventaja es la afirmación de que la topología es totalmente dinámica, esto debido a que un nodo puede ser perfectamente ser puesto fuera de servicio, temporal o permanentemente (esto puede ser a causa de que los nodos se pierdan, sean destruidos o por ser desabastecidos de energía).

Los sistemas de este tipo, es decir, las redes sensoriales inalámbricas tienen la posibilidad y la característica de ser auto-administrables de manera computacional con el mínimo de interferencia humana en su operación. El sistema en sí, utiliza en su estructura, específicamente en su procesador, una serie de algoritmos que ayudan a determinar la más eficiente manera de distribución de los varios nodos de la estructura del sistema, además de la eficiencia en costos a la par con dicho análisis tanto para la distribución como para el almacenamiento de datos de reporte.

Una red en malla es realmente un nombre genérico para determinar a través de un nombre a una clase de los Sistemas informáticos incrustados en una red que comparten muchas características tales como:

- **Multi-Hop (Salto múltiple);** la capacidad de enviar mensajes par – a – par a una estación base, y habilitando de paso la extensión del rango de forma escalable.

- **Self-Configuring (Auto configuración);** es decir la capacidad de este tipo de Sistemas de la formación de la red sin la intervención humana para su desarrollo de por medio.

- **Self-Healing (Auto Limpieza);** es decir, la capacidad de los Sistemas de añadir o deshacerse automáticamente de nodos con fallas o fuera de función, sin la necesidad de resetear la red entera; y

- **Dynamic Routing (Enrutamiento Dinámico);** Sistemas capaces de determinar adaptativamente la ruta basándose en condiciones dinámicas de red, como por ejemplo calidad de conexión, cuenta de saltos, gradiente, u otros parámetros.

Cuando estas características son combinadas con un adecuado manejo de energía a través de baterías, se permite que las redes Sensoriales sean de 'larga vida', fácilmente desarrolladas, y altamente resistentes frente a canales impredecibles inalámbricos. En definitiva, con este tipo de orientación tecnológica, la visión de un sensamiento altamente confiable y fortalecido, se vuelve una realidad.

La implicación y desarrollo, así como su utilización, de las redes Sensoriales, está compuesta básicamente de tres niveles de software básicamente:

- El Nivel del Cliente, que provee de la visión desde el punto de vista del usuario, así como la muestra de la necesidad de una interfaz gráfica de visualización para la administración de la red.

- El nivel de Servidor, la parte, en cierta forma, administrativa de la información, siempre a la vanguardia de la necesidad de manejo de datos así como la traducción de la información proveniente desde los sensores (*nodos*) y además, provee de un puente de enlace entre los módulos inalámbricos y los clientes de Internet. *XServe* y *XOtap* son ejemplos de software de este nivel, los cuales pueden ser ejecutados a través de una PC.

- El nivel de Sensor, donde *XMesh* reside, es el software que se ejecuta dentro de la nube donde se encierran a todos los sensores que conforman la malla de la red sensorial inalámbrica. *XMesh* es un software que provee los algoritmos de interconexión necesarios para una confiable conformación del esqueleto de comunicaciones que conecta a todos los dentro de la misma red al servidor.

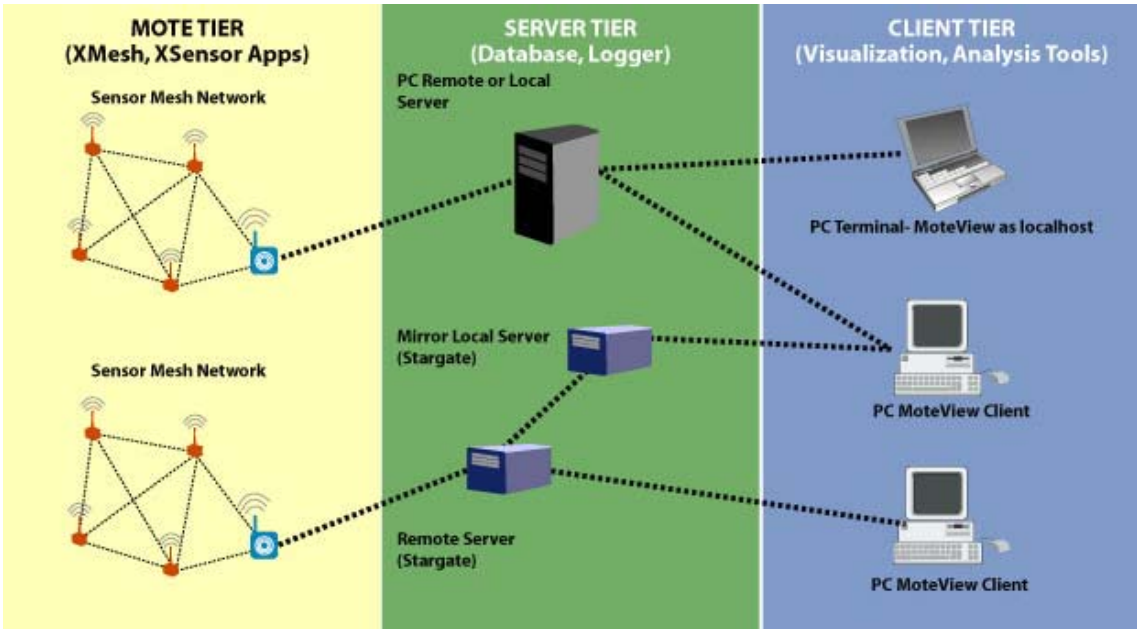


Figura 1.2: Niveles de Software en una Red Sensorial[1]

1.4 USOS Y APLICACIONES

Una red WSN consiste en dispositivos autónomos distribuidos, que se usan para supervisar condiciones físicas o ambientales, tales como temperatura, sonido, vibración, presión, movimiento o agentes contaminadores, en diversas localizaciones. El desarrollo de las redes sensoriales inalámbricas fue motivado originalmente por usos militares

[1] MoteView Users Manual, página 2

tales como vigilancia del campo de batalla. Sin embargo, ahora se utilizan en muchas áreas de aplicación civiles, incluyendo la supervisión del ambiente y del hábitat, usos de monitoreo médico, automatización casera, y control de tráfico.

Además uno o más sensores en cada nodo de la red se equipa típicamente con un transmisor-receptor de radio, un microcontrolador pequeño, y una fuente de energía, generalmente una batería. El tamaño de un solo nodo del sensor puede variar desde tamaños relativos a una caja de zapatos, hasta el de un grano del polvo. El costo de nodos del sensor es igualmente variable, extendiéndose desde centenares de dólares hasta algunos centavos, dependiendo del tamaño de la red y de la complejidad requerida en nodos individuales del sensor. Los apremios del tamaño y de coste en nodos del sensor dan lugar a algunas ventajas correspondientes a recursos tales como energía, capacidad de memoria, velocidad de cómputo y ancho de banda.

En cuanto a la informática, las redes sensoriales inalámbricas son un área muy activa de la investigación con numerosos talleres, prácticas y conferencias expuestas dispuestas cada año, debido a la gran contribución que representan.

Los usos para WSN’s son muchos y variados. Se utilizan en aplicaciones comerciales e industriales para supervisión de los datos que serían, comúnmente, difíciles o altamente costosos de supervisar con los sensores manejados o controlados a través de alambre.

Los más recientes avances de las comunicaciones reflejan los verdaderos requerimientos de la tecnología inalámbrica en la industria del presente. Las distintas interfaces inalámbricas para sistemas de sensores inteligentes en redes, permiten la simple medición de la capacidad de transmisión de datos y sus implicaciones cuando se deben realizar desde robots móviles y plataformas del mismo tipo, así como partes o maquinaria que no son fácilmente accesibles de un proceso, como por ejemplo piezas rotatorias.

La industria automotriz es otro de los potenciales clientes en un futuro muy cercano en el área de las redes de sensores y equipamiento inalámbrico, por su gran aporte al desarrollo de la mencionada tecnología.

Muchos sensores en fábricas o coches, por ejemplo, tienen su propia red conectada con un ordenador o una caja de controles a través de un cable y, al detectar una anomalía, envían un aviso a la caja de controles. La diferencia entre los sensores que todos conocemos y la nueva generación de redes de sensores sin cable es que estos últimos son ***inteligentes*** (es decir, capaces de poner en marcha una acción según la información que vayan acumulando) y no son limitados por un cable fijo.

Pero nuevos avances en la fabricación de microchips de radio, nuevas formas de routers y nuevos programas informáticos relacionados con redes están logrando eliminar los cables de las redes de sensores, multiplicando así su potencial.

Las redes de sensores pueden utilizar distintas tecnologías de sin cable, incluyendo *IEEE 802.11*, *LANS* sin cable, *Bluetooth* e identificación de la frecuencia de radio. Actualmente se trabaja con radios de baja frecuencia con un alcance de hasta **80 metros** y velocidades de hasta **300 Kb/segundo**.

Las últimas investigaciones apuntan hacia una eventual proliferación de redes de sensores inteligentes, redes que recogerán enormes cantidades de información hasta ahora no registrada que contribuirá de forma favorable al buen funcionamiento de fábricas, al cuidado de cultivos, a tareas domésticas, a la organización del trabajo y a la predicción de desastres naturales como los terremotos. En este sentido, la computación que penetra en todas las facetas de la vida diaria de los seres humanos está a punto de convertirse en realidad.

Aunque la tecnología relacionada con las redes de sensores sin cable está todavía en su primera fase, equipos de investigación en la Universidad de California Berkeley ya han fabricado una caja que se puede adaptar a muchos tipos de sensores. Los científicos utilizan los sensores sin cable para encontrar y controlar microclimas y

plagas en plantaciones de uva, para estudiar los hábitos de aves y para controlar sistemas de ventilación y calefacción. En la Universidad de California, Los Angeles, investigadores utilizan las redes de sensores sin cable para recibir información detallada sobre el efecto de los movimientos sísmicos en los edificios.

Si los avances tecnológicos en este campo siguen a la misma velocidad que han hecho en los últimos 2 años, las redes de sensores sin cable revolucionará la capacidad de interacción de los seres humanos con el mundo.

En la actualidad, uno de los inmediatos medios de aplicación de las WSN's es la creación de una versión inalámbrica de una "*Nariz Electrónica*". A cambio de sonidos y alarmas audibles, los sensores inalámbricos pueden ser fácilmente instalados en el techo de un auto, tan fácil como se instalaría un domo en el techo de una casa. De tal forma, que en un automóvil, sin conexiones de cables y nodos visibles, un sensor inalámbrico podrá transmitir la posición centralizada del automotor constantemente a un receptor remoto.

Otra muy interesante forma de aplicar los sensores de este tipo, es a través de sensores sin conexión física implantados en el cuerpo humano, esto para monitoreo de salud, arritmia u otro tipo de señales de nuestra humanidad.

Podrían ser desplegados en áreas ambientalmente importantes para la humanidad y el ecosistema, donde permanecerían por muchos años (que supervisan una cierta variable ambiental) sin la necesidad de recargar o sustituir sus fuentes de alimentación. En el mismo campo, en el ambiental, estos nodos podrías formar un perímetro sobre cerrado, centrando su operación y característica en la supervisión del ingreso de intrusos (pasando la información a partir de un nodo al siguiente a manera de cadena); por lo que son de gran ayuda en este tipo de aplicaciones.

Las aplicaciones típicas de las Redes Sensoriales Inalámbricas incluyen la supervisión, seguimiento, y control. Algunos de los usos específicos son:

- El monitoreo en un hábitat,
- Seguimiento de objetivos móviles,
- Control nuclear de un reactor,
- Detección de fuego,
- Tráfico, etc.

En un campo de acción típico de las WSN's, los nodos se dispersa en una región donde se disponen para recoger datos con sus nodos del sensor. Esto en aplicaciones como:

Tabla 1.1: Aplicaciones de Redes Sensoriales

PRINCIPALES APLICACIONES DE REDES SENSORIALES
Monitorización de un hábitat (para determinar la población y comportamiento de animales y plantas)
Monitorización del medio ambiente, observación del suelo o agua
El mantenimiento de ciertas condiciones físicas (temperatura, luz)
Control de parámetros en la agricultura
Detección de incendios, terremotos o inundaciones
Monitoreo de edificios "inteligentes"
Control de tráfico
Asistencia militar o civil
Control de inventario
Control médico
Detección acústica
Cadenas de montaje

Entre las más vistosas y posibles aplicaciones de las WSN's, podemos enumerar algunas como la supervisión de áreas es una aplicación típica de esta tecnología. Dentro del área que se supervisa, los distintos nodos se despliegan sobre una región donde va a ser monitoreado un cierto fenómeno. Como por ejemplo, una cantidad grande de nodos del sensor se podía desplegar sobre un campo de batalla para detectar la intrusión enemiga en vez de usar minas de tierra. Cuando los sensores detectan el acontecimiento que es supervisado (calor, presión, sonido, luz, campo electromagnético, vibración, etc.), el evento necesita ser divulgado a una de las estaciones bajas, que pueden tomar la acción apropiada (p.e., enviar un mensaje en el Internet o a un satélite). Dependiendo del uso exacto, diversas funciones objetivas requerirán diversas estrategias de la propagación de datos, dependiendo de cosas tales como necesidad de la respuesta en

tiempo real, redundancia de los datos (que se pueden abordar vía técnicas de la agregación de los datos), necesidad de la seguridad, etc.

Dentro de las características de una Red Sensorial Inalámbrica podemos enumerar algunas como:

- Nodos en reducida escala del sensor
- Energía limitada que pueden cosechar o almacenar
- Condiciones ambientales ásperas
- Faltas del nodo
- Movilidad de nodos
- Topología dinámica de la red
- Faltas de comunicación
- Heterogeneidad de nodos
- Escala grande del despliegue
- Funcionamiento sin atención

Los nodos se pueden imaginar como computadoras pequeñas, extremadamente básicas en términos de sus interfaces y sus componentes. Consisten en generalmente una unidad de proceso con energía de cómputo y memoria limitada, sensores (trazado de circuito de condicionamiento específico), un dispositivo de comunicación (los transmisores-receptores o alternativamente ópticos, generalmente de radio), y una fuente de energía generalmente bajo la forma de batería. Otras inclusiones posibles son energía que cosechan los módulos, ASIC's secundarios, y posiblemente dispositivos secundarios de la comunicación (p.e. RS232 o USB).

Las estaciones bajas son unos o más componentes distinguidos de la Red Sensorial con recursos mucho más de cómputo, energía y de la comunicación. Actúan como entrada entre los nodos sensoriales y el usuario final.

- **Contemplaciones**

Dentro de la estructura de las redes sensoriales inalámbricas, se deberán tener en cuenta algunos parámetros de acción como cambios de topología, características del grupo de sensores, tráfico de la red, errores de transmisión, pérdida de nodos, entre otros. Para lo cual, en la infraestructura de un sistema de este tipo, al momento de la implementación o planeación de la misma deberemos tener en cuenta aspectos como:

- **Topología,** formada en la etapa de planeación del sistema, deberemos tener en cuenta la forma en la cual serán distribuidos los distintos nodos, dependiendo claro está de ciertos parámetros de acción como las condiciones del ambiente donde se los va a instalar, distribuciones de mejor y/o mayor capacidad de información y aprovechamiento de recursos. De tal forma que, al momento de analizar como es que se va a repartir el determinado número de nodos en el espacio físico asignado hay que determinar la distancia, altura, posición, entre otros.

- **Tráfico de la Red,** un aspecto que puede afectar de manera muy significativa a la estructura de la red, dependiendo de la naturaleza en la que se desenvuelva la red implementada. Es decir, dependiendo del fenómeno especifico que se observe a través de la red, podemos determinar la capacidad necesaria para la transmisión de datos a través de red.

- **Mantenimiento de las Comunicaciones,** es decir el aseguramiento del correcto estado y mantenimiento de los equipos y por lo tanto de la operación de las comunicaciones dentro de la red inalámbrica, que dentro de lo acordado (el protocolo de comunicación) deberá contemplar aspectos como la formación inicial de la Red, enrutamiento, mantenimiento y optimización de la estructura de la red, entre otros.

Actualmente, las Redes Sensoriales son la llave de reducir no solo en costos sino en aprovechamiento de recursos las necesidades de transmisión y supervisión de fenómenos necesarios en ambientes de cualquier índole. Simplemente es una forma inteligente de utilizar recursos, ya sea en edificaciones, utilidades, industria, el hogar,

transportación, sistemas de automatización, o en cualquier aplicación en la que sean dimensionados.

- **IEEE 1451 y Sensores Inteligentes**

Las Redes WSN satisfacen completamente dichos requerimientos de funcionalidad y mantenimiento, las funciones deseadas para los nodos sensoriales incluyen: facilidad de instalación, auto-identificación, auto-evaluación, precisión, alerta en el tiempo y sincronización para poder coordinar operaciones con otros nodos, algunas funciones de software y DSP, y protocolos o interfaces en estándares de control del tipo IEEE 1451.

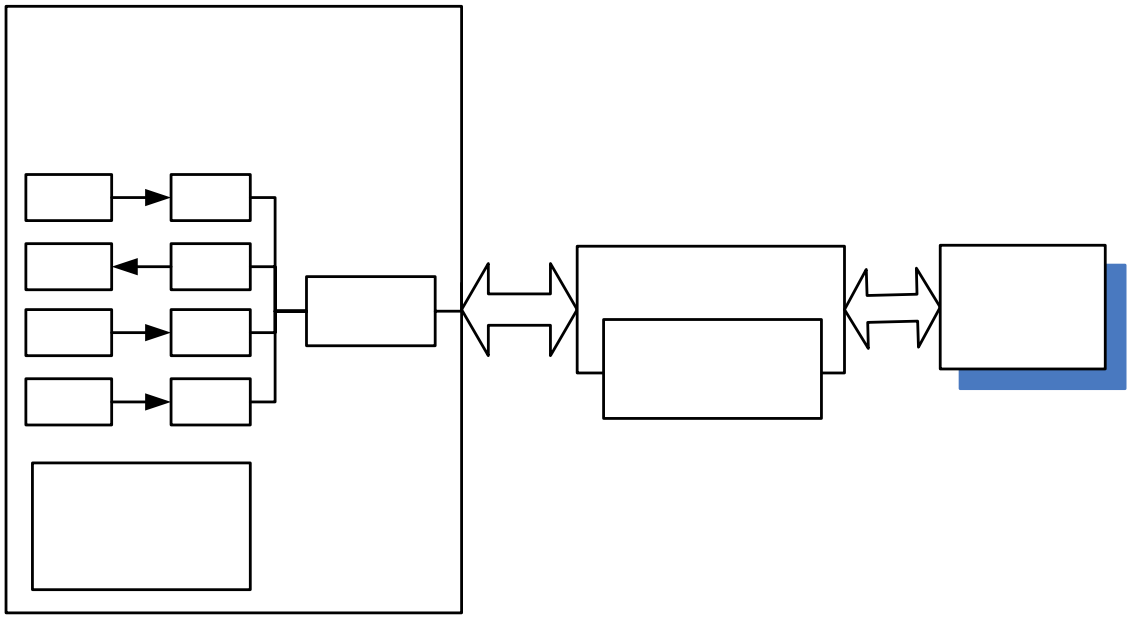


Figura 1.3: Estándar IEEE 1451 para Redes de Sensores Inteligentes

- **Plataformas**

Hardware

El desafío principal es producir bajo costo y nodos minúsculos de sensores. Con respecto a estos objetivos, los nodos actuales del sensor son principalmente prototipos. La miniaturización y el bajo costo se entienden para seguir de progreso reciente y futuro

en los campos de MEMS y de NEMS. Algunos de los nodos existentes del sensor se dan a continuación. Algunos de los nodos todavía están en etapa de la investigación.

- **Estándares**

Los estándares utilizados por esta tendencia tecnológica, son principalmente, debido a la característica de ser inalámbricos, los siguientes:

- ZigBee
- 6lowpan

Software

La energía es el recurso más escaso de los nodos de WSN, y determina el curso de la vida de los mismos. Una Red Sensorial Inalámbrica se crea para ser desplegada en numerosos ambientes, incluyendo regiones alejadas y hostiles, con comunicaciones ad hoc como clave de comunicación. Por esta razón, los algoritmos y los protocolos necesitan tratar las ediciones siguientes:

- Maximización del curso de la vida
- Tolerancia de la robustez y de avería
- Auto configuración

Entre los asuntos más importantes de lo que se refiere a software de las Redes de este tipo, debemos estar concientes de ciertas características como:

- Seguridad
- Movilidad (cuando los nodos del sensor o las estaciones bajas se están moviendo)
- Middleware: el diseño de los primitivos del medio-nivel entre el software y el hardware

- **Sistemas Operativos**

1. Bertha

2. BTnut Nut/OS
3. Contiki
4. CORMOS: Una comunicación orientada al sistema Runtime para las redes del sensor.
5. eCos
6. EYESOS
7. Magnetos
8. PREDICADOR (sensores "in-situ" de las redes de MultimodAl).
9. SenOS
10. SOS
11. TinyOS
12. T-Núcleo
13. LiteOS

- **Middleware**

Hay una necesidad y esfuerzos considerables de la investigación invertidos actualmente en el diseño del middleware para Redes Sensoriales Inalámbricas. Hay varios esfuerzos de la investigación en el middleware que se convierte para las redes sensoriales. En general los acercamientos pueden ser clasificados en la base de datos distribuida, agentes móviles, y acontecimiento.

- **Lenguajes de programación**

La programación de los nodos de sensores es difícil cuando está comparada a los sistemas informáticos normales. La naturaleza obligada recurso de estos nodos da lugar a nuevos modelos de programación.

- c@t (cómputo en un punto en el espacio (@) Tiempo).
- DCL (lengua compositiva distribuida).
- galsC
- NESC
- Protothreads
- SQTL

CAPÍTULO 2

ESTUDIO DEL SOFTWARE DE PROGRAMACION DE SENSORES INALAMBRICOS

2.1 INTRODUCCIÓN

Dentro del presente capítulo se realiza ya, una introducción de lleno con los equipos físicos seleccionados para la implementación del presente proyecto, es por eso, que durante el desarrollo del mismo, se necesita entender sobre el proceso completo que comprende el software específico de programación, así como de visualización y recopilación de la información necesaria para el completo funcionamiento de la Red Sensorial mencionada para el proyecto.

Una vez aclarado el objetivo del presente capítulo, el software en mención, para el caso, será el *MoteView*, una versión del sistema de programación para los módulos de sensamiento del sistema, adquiridos y provistos por la empresa **Cross Bow Technology Inc.** a través de los cuales nos ayudamos para el desarrollo del sistema de sensores inalámbricos, de tal forma que mediante su estudio, vamos a determinar la importancia de su uso, así como las ventajas y desventajas que ofrece.

Como se explicó, ya durante el desarrollo del primer capítulo, de los tres escalones o niveles de programación, *MoteView*, pertenece al nivel de cliente (donde se denota la necesidad de una interfaz gráfica de visualización), entre un usuario y una red

sensorial inalámbrica ya desarrollada. En general, este software, lo que hace es proveer a los usuarios de herramientas para simplificar las tareas de desarrollo, implementación, desarrollo, monitoreo y análisis de información, provista por la Red; y justamente, dentro de la parte relacionada con el análisis de los datos, MoteView hace fácil la tarea de conectarse a una Base de Datos para dicho análisis y generación de gráficas de las lecturas de cada uno de los nodos implícitos en la red sensorial.

Dentro de lo que implica el desarrollo de las Redes Sensoriales Inalámbricas, la ***Figura 1.2*** muestra la manera en que interactúan los niveles de software, es decir, podemos observar, como la primera parte corresponde a la conformación o disposición de los nodos en la red misma, los Sensores, en este nivel son programados a través de aplicaciones específicas, que en esta caso son *XMesh*/TinyOS para realizar una tarea específica, como por ejemplo: monitoreo de microclimas, rastreo, detección de intrusos, entre otras.

La segunda etapa del gráfico corresponde al nivel del Servidor, donde se tiene acceso a la información directamente, así como la facilidad de la formación de Bases de Datos recopilados dentro del sistema en sí; es decir, en este nivel de operación es donde se efectúa la recepción de los datos generados o más bien, sensados, por los módulos de la primera etapa, hasta la Estación Base, que puede ser uno de los módulos disponibles para comercialización, como por ejemplo MIB510, MIB520, MIB600 o *Stargates* disponibles en la configuración del sistema en si; datos que son almacenados en un Servidor.

Y, finalmente, la tercera parte de la Figura, muestra la parte del cliente, justamente, la parte en la que se aloja nuestro software (*MoteView*), es decir el nivel de visualización, monitoreo y análisis con herramientas que se muestran en la interfaz para interpretación de los datos de los sensores; y es este nivel el que será detallado en esta sección del proyecto, para poder determinar cada una de las características que ofrece el software con el cual se tiene acceso al funcionamiento de los módulos sensoriales, plataformas y tableros.

Tal y como se ha comentado, los motes MICAz funcionan bajo el sistema operativo TinyOS (Tiny microthreading Operative System), desarrollado en la Universidad de Berkeley. En este apartado se describen las características de este sistema operativo y de su lenguaje de programación. También se ha realizado una breve descripción del programa Serial Forwarder y CSharp (C#) para recoger datos de los motes y del software de encaminamiento NSTAODV.

2.2 ANÁLISIS DEL SOFTWARE

Dentro de lo específicamente es el software *MoteView,* el cual trabaja basado en el sistema *TinyOs (Tiny Microthreading Operative System)*, que tiene como lenguaje de programación un esquema similar al C, llamado NesC, conociendo esto se debe analizar los principales requerimientos básicos necesarios para la operación del software, debiendo pasar de por medio, por las necesidades de capacidad, plataformas de operación, puertos de interfaz, entre otros, así, a continuación se muestra un resumen acerca de estos temas.

En cuanto a **plataformas y sistemas operativos** soportados, MoteView, tiene capacidad de trabajar directamente con plataformas comerciales basadas en Windows principalmente:

- Windows XP Home
- Windows XP Profesional
- Windows 2000 con Service Pack 4

Además, se deberá tener en cuanta que la resolución de la pantalla de visualización deberá ser de por lo menos de 800 x 600, caso contrario, la interfaz de visualización requerirá de barras de desplazamiento.

Luego, una vez conocidas las necesidades de plataformas de operación, es importante que también tengamos en cuanta los puertos de interfase, es decir, la forma en que vamos a lograr la conexión física entre la PC y nuestro módulo de comunicación con los sensores; de tal forma, tenemos de acuerdo al módulo de sensores adquirido, opciones de comunicación, de tal forma que:

- Para módulos del tipo **MIB510,** requerimos de una conexión de tipo Serial, es decir, necesitamos un puerto RS-232.

- Para el modelo **MIB520,** con salida de recepción de datos tipo USB; un puerto USB.

- Para el modelo **MIB600,** por otro lado, con salida de datos vía Ethernet, requerimos conectada a la PC una tarjeta Ethernet o 802.11 inalámbrica, solamente si es necesaria la implementación de una LAN con acceso inalámbrico.

- Finalmente, para un **Stargate** o para otro servidor, será necesaria la existencia de una tarjeta Ethernet, una tarjeta 802.11 wireless simplemente, si el *Stargate* tiene un módem inalámbrico o si se encuentra en una LAN con acceso inalámbrico, o un módem celular para acceso a Internet de forma inalámbrica.

Además, después de verificar la necesidad de estas características para el correcto funcionamiento de nuestro software, y garantizar su operación con los módulos, también tenemos la posibilidad de ayudarnos mediante algunas herramientas adicionales, como por ejemplo PostgreSQL 8.0 (servicio de *base de datos*), o PostgreSQL (servicio *ODBC*); o también *Microsoft .NET*. De cualquier forma, son algunas características, que a través de la guía de instalación del software se puede ir accediendo y añadiendo para tener acceso a dichas características ofrecidas por la firma desarrolladora.

2.2.1 TinyOs y NesC

El sistema operativo TinyOS, como su nombre indica es un reducido núcleo multitarea, útil para pequeños dispositivos, tales como las motas. Es un sistema operativo “event-driven”, quiere decir que funciona a partir de eventos producidos que llamarán a funciones. Ha sido desarrollado para redes de sensores con recursos limitados. El entorno de desarrollo de TinyOS soporta directamente la programación de diferentes microprocesadores y permite programar cada tipo con un único identificador

para diferenciarlo, o lo que es lo mismo se puede compilar en diferentes plataformas cambiando el atributo.

TinyOS es un sistema operativo de código abierto diseñado para redes sin hilos de sistemas embebidos. Su arquitectura está basada en componentes. En este tipo de arquitecturas las aplicaciones se crean enlazando componentes, permitiendo de esta manera la reutilización de código a través de sus interfaces y la abstracción de las características del hardware con el que se trabaja. La versión de TinyOS que se empleara en el proyecto es la 1.1.10 que es una versión ya testeada, actualización de la 1.1.0.

TinyOS crea un nivel intermedio entre las capas inferiores PHY y MAC proporcionadas por el 802.15.4 y las capas superiores de encaminamiento tal y como muestra la siguiente figura:

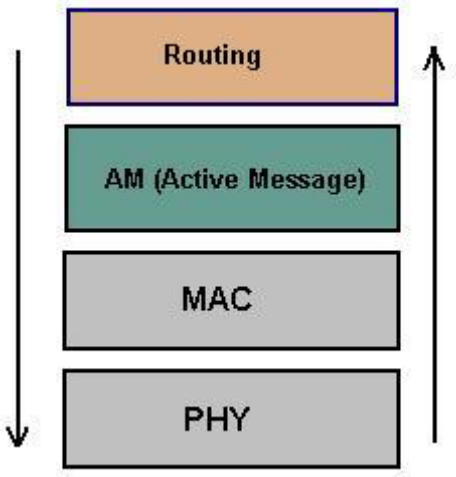


Figura 2.1: Interacción de capas zigbee

La mencionada capa intermedia es la AM (Active Message). Gracias a ella se abstraen las capas superiores de las inferiores, siendo esta capa la que actúa de puente entre ellas. La siguiente figura muestra el formato de los mensajes AM así como su encapsulado dentro de los paquetes 802.15.4:

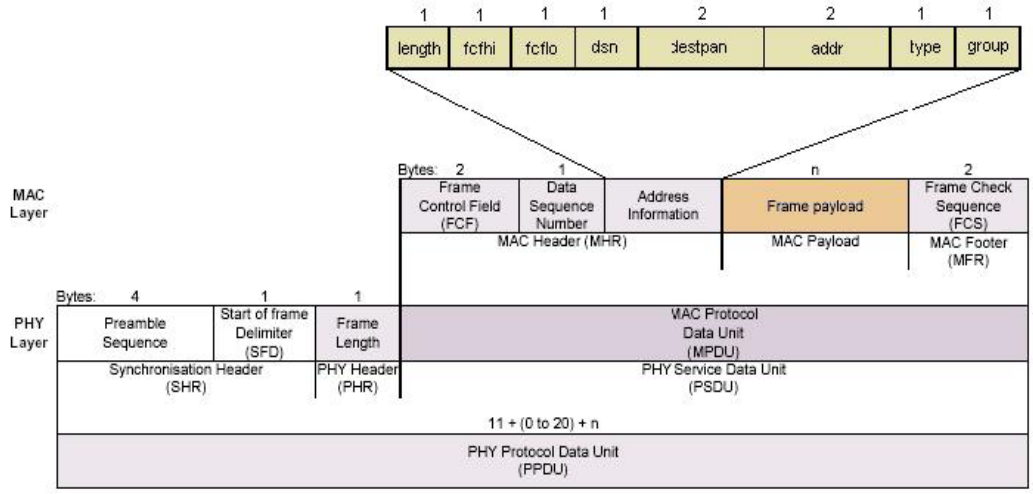


Figura 2.2: Vista esquemática del formato de la trama de datos IEE 802.15.4

El lenguaje de programación de TinyOS es NesC, una extensión del lenguaje C utilizada para la programación de aplicaciones basadas en componentes. Este lenguaje permite el desarrollo de aplicaciones que necesitan poco espacio de memoria, uno de los requerimientos más importantes en los sistemas empotrados. El diseño está basado en un núcleo multihilo simple y eficiente, con tres tipos de abstracciones: eventos, tareas y comandos.

Los eventos son llamados hacia arriba, es decir, de componentes de bajo nivel a componentes de un nivel más alto, para avisar de algo que ha ocurrido (por ej., la recepción de un mensaje). Al contrario que los eventos, las tareas son llamadas hacia abajo. Finalmente, las tareas son trozos de código que se ejecutan asíncronamente

cuando no hay ningún evento a ejecutar, dado que los eventos tienen más prioridad que las tareas. Éstas se ejecutan según el orden FIFO.

En NesC existen dos tipos de componentes: los módulos y las configuraciones. Los componentes proporcionan y utilizan interfaces, que son los puntos de acceso hacia y desde los componentes. En las interfaces se declaran comandos que ha de implementar el que proporciona la interfaz y eventos que han de ser implementados por el usuario de dicha interfaz. Los módulos contienen el código del programa y es donde se implementan las interfaces, mientras que las configuraciones son las que se usan para enlazar las interfaces de los componentes entre sí, proceso que se conoce también como wiring. Los tipos de datos implementados en ella son los que ya dispone C, además de los tipos uint8_t (unsigned integer de 8 bits), uint16_t (lo mismo pero de 16 bits), result_t y bool.

Los componentes se enlazan a través de sus interfaces. Estas interfaces son bidireccionales y especifican un conjunto de funciones que están implementadas bien por los proveedores o bien por los que la utilizan. NesC esperará que el código que va a ser generado cree un programa con un ejecutable que contenga todos los elementos del mismo, así como los manejadores de las interrupciones de programas de más alto nivel.

TinyOS tiene las siguientes características:

- Pequeño núcleo de footprint (huella del ejecutable del SO) de 400bytes entre código y datos.
- Arquitectura basada en componentes.

- Capas de abstracción bien establecidas, limitadas claramente a nivel de interfaces, a la vez que se pueden representar los componentes automáticamente a través de diagramas.
- Amplios recursos para elaborar aplicaciones.
- Adaptado a los recursos limitados de las motas: energía, procesamiento, almacenamiento y ancho de banda.
- Operaciones divididas en fases (Split-phase).
- Dirigido por eventos (Event Driven): reacciona ante sensores y mensajes.
- Concurrencia de tareas y basada en eventos.
- Implementación en nesC.
- Las interfaces realizan servicios.
- Interfaces bidireccionales, con comandos y eventos.
- Los comandos los implementa el proveedor.
- Los eventos son implementados por el usuario.
- Un módulo implementa una interfaz.
- Los componentes proveen y usan interfaces (representado en el código por las etiquetas “provide and use”).
- Una configuración enlaza las interfaces internas y externas (wire).
- Una aplicación consiste en una configuración de alto nivel y todos los módulos asociados.

Filosofía de programación TinyOs

El sistema operativo TinyOs, las librerías, y las aplicaciones están escritos en NesC, un nuevo lenguaje de basado en componentes estructurados. El lenguaje NesC esta destinado primariamente hacia sistemas embebidos (embedded) tales como redes de

sensores. NesC tiene una sintaxis tipo, pero soporta el modelo de concurrencia, tales como mecanismos para estructurar, nombrar, y enlazar en si los componentes del software en sistemas de redes embedded robustas. El objetivo principal es permitir a los diseñadotes de la aplicación construir componentes que pueden ser fácilmente en sistemas concurrentes completos, y así realizar chequeo extensivo al momento de compilar.

TinyOs también define un número de conceptos importantes que pueden ser expresados en NesC. Aquí se provee una breve explicación:

Tabla 2.1: Descripción de los conceptos principales de TinyOs/NesC

Concepto	Descripción
TinyOs/NesC Aplicación	Una aplicación TinyOs/NesC consiste en uno o mas componentes, enlazados (“alambrados”) entre si para formar un ejecutable en tiempo de corrida
Componente	Componentes son la estructura de bloques básica para una aplicación en Nesc. Existen 2 tipos de componentes: módulos y configuraciones. Un componente TinyOs puede proveer y usar interfaces
Modulo	Un componente que implementa una o mas interfaces
Configuración	Un componente que une alámbricamente otros componentes entre si, conectando interfaces usadas por componentes hacia interfaces provistas por otros. (Esto es llamado cableado). La idea es que un desarrollador pueda construir una aplicación como un set de módulos, cableando entre si los módulos provistos de una configuración. Además cada aplicación NesC es descrita como una configuración de alto nivel que describe los componentes en la aplicación y como estos se llaman entre si
Interfase	Una interfase es usada para proveer una definición abstracta de la interacción de dos componentes. Este concepto es similar a Java en que una interfaz no debe contener código ni cableado. Esto es simplemente la declaración de un set de funciones que el proveedor de la interfase debe implementar, con comandos y otro set de función que se debe implementar también, que son los eventos. En este caso es diferente a las interfaces Java donde se especifica una dirección de llamada. Las interfaces Nesc son bidireccionales. Para un componente realizar las llamadas de los comandos en una interfase, debe implementar los eventos de dicha interfase. Un componente simple debe requerir o proveer múltiples interfaces e instancias múltiples para la misma interfase. Estas interfases son el único punto de acceso al componente.

2.2.2 Serial Forwarder y CSharp

Para recoger los datos de los sensores se usa el software de soporte Serial Forwarder, que actúa de interfaz entre el PC y el nodo "base" acoplado a la placa programadora. Los datos que recoge el nodo base los reenvía al puerto UART de la placa, y es el Serial Forwarder el que se puede comunicar con otros programas mediante sockets. En el caso del proyecto, se han utilizado diferentes programas en lenguaje Csharp (C#) para recoger los datos recibidos por el puerto UART.

2.3 MANEJO DEL SOFTWARE

Una vez que se tiene el software de operación de los módulos adquiridos para el desarrollo de la Red Sensorial, podemos acceder al manejo del programa, para lo cual se podrá empezar a maniobrar su interfaz, de tal manera, que esta sección del capítulo estará dedicada a la operación y reconocimiento de las funcionalidades del software en cuestión.

De tal forma, que se realizará una descripción específica de cada una de las funciones ofrecidas por MoteView, entre las cuales básicamente tenemos cuatro secciones principales, una subdivisión realizada para analizar por bloques las funcionalidades del sistema, y son:

1. Barras de Herramientas/Menús; que permiten al usuario tener acceso a acciones específicas o a comandos.
2. Listado de Nodos; que muestra todos los nodos de la red en cuestión junto con su diagnóstico de funcionamiento
3. Comandos de Visualización; opciones de visualización en varias formas para el usuario, de los datos captados por los sensores.

4. Mensajes del Servidor, que muestras acciones de acceso, de eventos del servidor o mensajes entrantes para el usuario.

Realizaremos entonces un análisis de cada una de las funcionalidades disponibles para el proyecto, en el software a ser usado, a través de la subdivisión antes mencionada, se puede acceder a una descripción generalizada acerca de lo importante que es su ayuda como asistente de programación.

Así pues, entre los varios ítems del listado de funciones del sistema, se puede observar, a lo cual se refiere como *listado de nodos*, importantísimo, aspecto al momento de analizar cada uno de los componentes del sistema, de tal forma, que a través de este sistema de visualización, se puede tener una clara y real idea del número y del estado de cada uno de los componentes del sistema de sensores, es decir, que se tendrá acceso total a las características y estado de los módulos sensoriales implícitos, ya sea que estén o no en funcionamiento correcto dentro del mismo sistema, esto debido a que como se había mencionado en otra sección del documento, una de las ventajas de este tipo de redes sensoriales, es justamente la habilidad de reconocer el estado de un nodo, si es posible, se recuperan de forma autónoma, o, también pueden reconocer cuando un nodo ha entrado o salido del sistema o red a la cual esta verificando.

Dentro de la información disponible, sobre cada uno de los nodos del sistema, se tienen datos importantes para el análisis como se puede observar en la *Tabla 2*, se tiene una clara y vistosa forma de conocer el estado de un nodo dentro de la Red en cuestión, es decir que podemos tanto acceder a la configuración de la Red en sí, como al

monitoreo del sistema total, pudiendo controlar cada uno de los aspectos y propiedades de los sensores que conformen nuestra Red Sensorial.

Entre otra de las partes con conforman la interfaz del software en cuestión, se tiene, la parte que tiene que ver con las opciones de visualización de los datos, del estado o de la operación de monitoreo del sistema total.

Tabla 2.2: Características Visualizadas en de los Nodos

Lista de características de los nodos	Descripción
Selección de Nodo por ID	Selecciona los nodos a ser graficados en esa opción de Visualización
Cambio de propiedades de nodos	Doble clic en un nodo de la lista o clic derecho para ver sus propiedades.
Ordenar datos	Clic en la columna cabecera, para ordenar los daos mostrados, pero esto fuera del modo en tiempo real.
Añadir Nodos	Clic derecho sobre cualquiera de los nodos, o se puede usar “auto-detectar”.
 Ícono Gris “Mote”	Sin resultados recibidos
 Ícono Verde “Mote”	Resultados frescos en lapso de 20 minutos
 Ícono Verde Claro “Mote”	Resultado no actual (>20 minutes)
 Ícono Amarillo “Mote”	Resultado no actual (>40 minutes)
 Ícono Naranja “Mote”	Resultado no actual (>60 minutes)
 Ícono Rojo “Mote”	Resultado no actual (más de un día)

Una vez que se ha determinado la forma, es decir, las características de la interfaz de usuario, entonces podemos proceder a la explicación del proceso de manejo del mismo, y en este aspecto *MoteView* nos ofrece la gamma de posibilidades del acople con el hardware, de tal forma, que una vez que se han programado los Motes, entonces la comunicación con nuestro software se realiza a través de la opción "*Connect*" del menú principal de la interfaz, en seguida de esto se visualizará una pantalla de opciones como la q se muestra en la figura de a continuación.

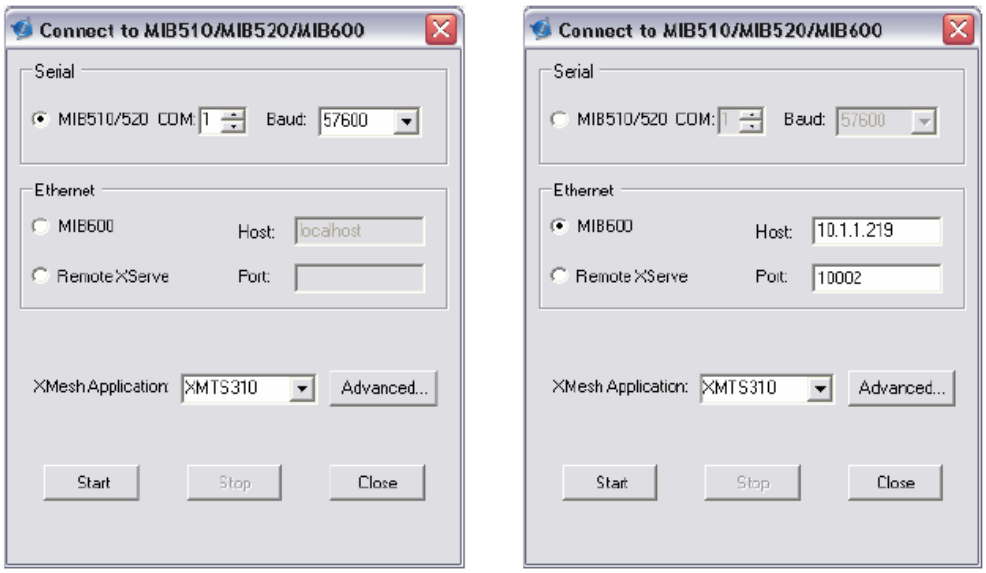


Figura 2. 3 Ventana de Conexión Software-Hardware

Ventana en la que destacan cada una de las siguientes características o propiedades de cambio:

1. Selección del Hardware, es decir, la parte dedicada a escoger la plataforma e interfaz de conexión directa con la parte física de nuestra PC, para el caso, puede ser cualquiera de las opciones MIB510 (Puerto Serial) ó MIB520 (Puerto USB) ó mediante Ethernet con el MIB600.
2. En el caso de la conexión directa con el hardware de la aplicación se debe aclarar la necesidad de la cooperación del programa con la solución de software de DOS

denominada XMesh, una de las aplicaciones básicas de este tipo de dispositivos diseñado para la presentación de los datos en formatos necesarios para la adquisición y manejo de los mismo, con opciones en el botón de *Advanced…* de poder tener plegados los datos en unidades Crudas, de ingeniería y transformadas en formatos necesarios dependiendo de la aplicación específica.

Una vez aplicado el tipo de hardware a ser utilizad, lo siguiente es la recepción de la información en la pantalla de la interfaz del *MoteView*, teniendo en cuenta que para eso habrá que estar seguros de la activación de la casilla *LIVE* en la parte superior de la pantalla principal del programa.

Para mayor facilidad, el usuario tiene la posibilidad de verificar la recepción de los datos de un sensor remoto, a través de los mensajes de estado presentes en la parte inferior de la pantalla del *MoteView*.

Ahora, una vez conectada la PC de manera activa con la red de sensores a través de la Estación Base (MIB's) es que se puede tener acceso a la visualización y manejo de la información proporcionada por la interfaz. Así, dentro de la constitución de la parte visual del programa, tenemos básicamente cuatro opciones desplegadas en pestañas como se muestra en la Figura:

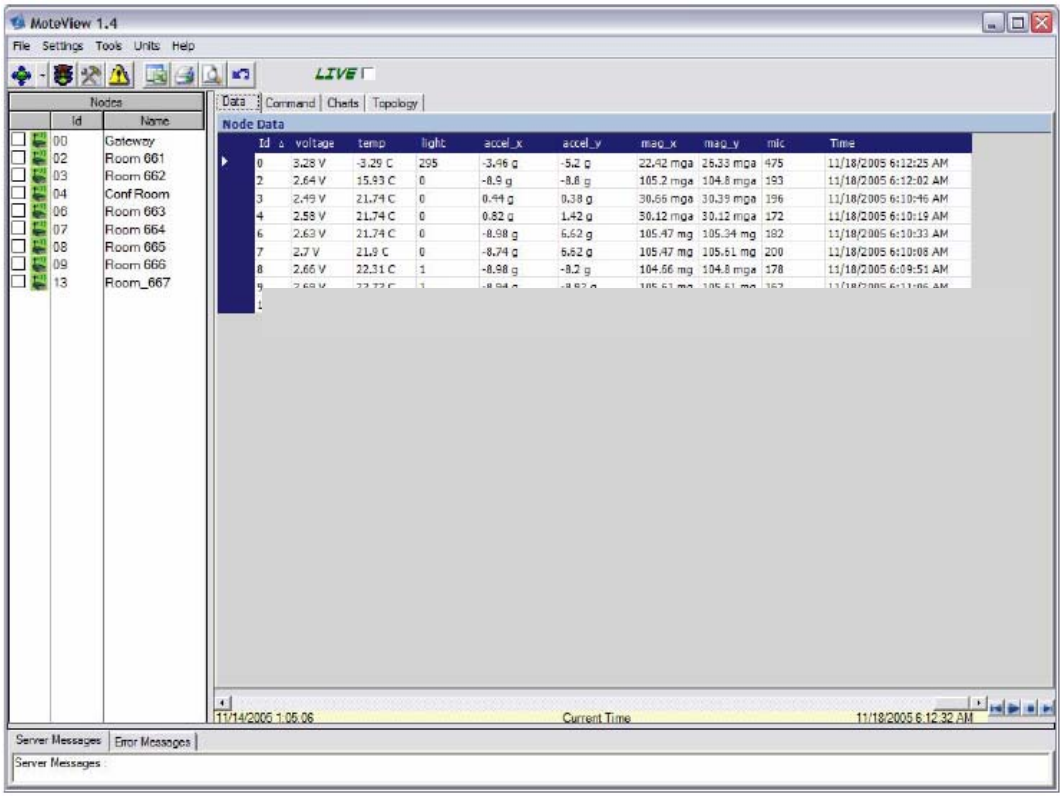


Figura 2.4: Opciones de Visualización de MoteView

Como se puede apreciar en la Figura 2.2, tenemos cuatro opciones de visualización posibles, las mismas que permiten acceder a ciertas características que para el usuario resultan de ventaja asombrosa, dependiendo de la aplicación.

En primer lugar; tenemos, la opción ***DATA***, a través de la cual, como usuarios tenemos la opción de visualizar cada parámetro posible a ser medido por lo equipos adquiridos, en el muy específico caso de la presente aplicación, por ejemplo, la temperatura, claro está, lo que permite esta opción es observar todos los parámetros y sus valores actuales sensados por el sistema.

Id	voltage	temp	light	accel_x	accel_y	mag_x	mag_y	mic	Time
0	3.28 V	-3.29 C	295	-3.46 g	-5.2 g	22.42 mga	26.33 mga	475	11/18/2005 6:12:25 AM
2	2.64 V	15.93 C	0	-8.9 g	-8.8 g	105.2 mga	104.8 mga	193	11/18/2005 6:12:02 AM
3	2.49 V	21.74 C	0	0.44 g	0.38 g	30.66 mga	30.39 mga	196	11/18/2005 6:10:46 AM
4	2.58 V	21.74 C	0	0.82 g	1.42 g	30.12 mga	30.12 mga	172	11/18/2005 6:10:19 AM
6	2.63 V	21.74 C	0	-8.98 g	6.62 g	105.47 mg	105.34 mg	182	11/18/2005 6:10:33 AM
7	2.7 V	21.9 C	0	-8.74 g	6.62 g	105.47 mg	105.61 mg	200	11/18/2005 6:10:08 AM
8	2.66 V	22.31 C	1	-8.98 g	-8.2 g	104.66 mg	104.8 mga	178	11/18/2005 6:09:51 AM
9	2.69 V	22.72 C	1	-8.94 g	-8.92 g	105.61 mg	105.61 mg	162	11/18/2005 6:11:06 AM
13	2.68 V	21.98 C	0	-8.98 g	-8.98 g	106.55 mg	106.28 mg	448	11/18/2005 6:12:32 AM

Figura 2.5: Visualización de Datos Sensados por el Sistema

En segundo lugar, dentro de las cuatro opciones de visualización, tenemos la parte correspondiente a ***COMMAND***, la misma que lo que nos permite es la posibilidad de cambiar ciertos parámetros de los sensores de manera inalámbrica. Dentro de los parámetros que son posibles cambiar con esta selección, se tiene:

- Tasa de Transmisión de Datos, en la pantalla claramente mostrada, de tal forma que, podemos escoger los valores adecuados, que para el caso de sensores de alta potencia puede variar entre 300 milisegundos y 2 segundos, y 3 minutos por defecto en los de bajo poder.
- Se puede obtener una única Dirección de 64 bits a través de la opción **Get** dentro de esta ventana.
- Además, podemos tener acceso al cambio del estado de los LED's de cada uno o en conjunto de los sensores. Teniendo acceso a cada uno de los 3 LED's existentes en los tableros del sensor (Rojo, Amarillo y Verde) y con tres posibles estados: *Encendido, Apagado, o Alternado*, estados que se pueden observar en la *Figura2.4*, en la sección correspondiente a los LED's.

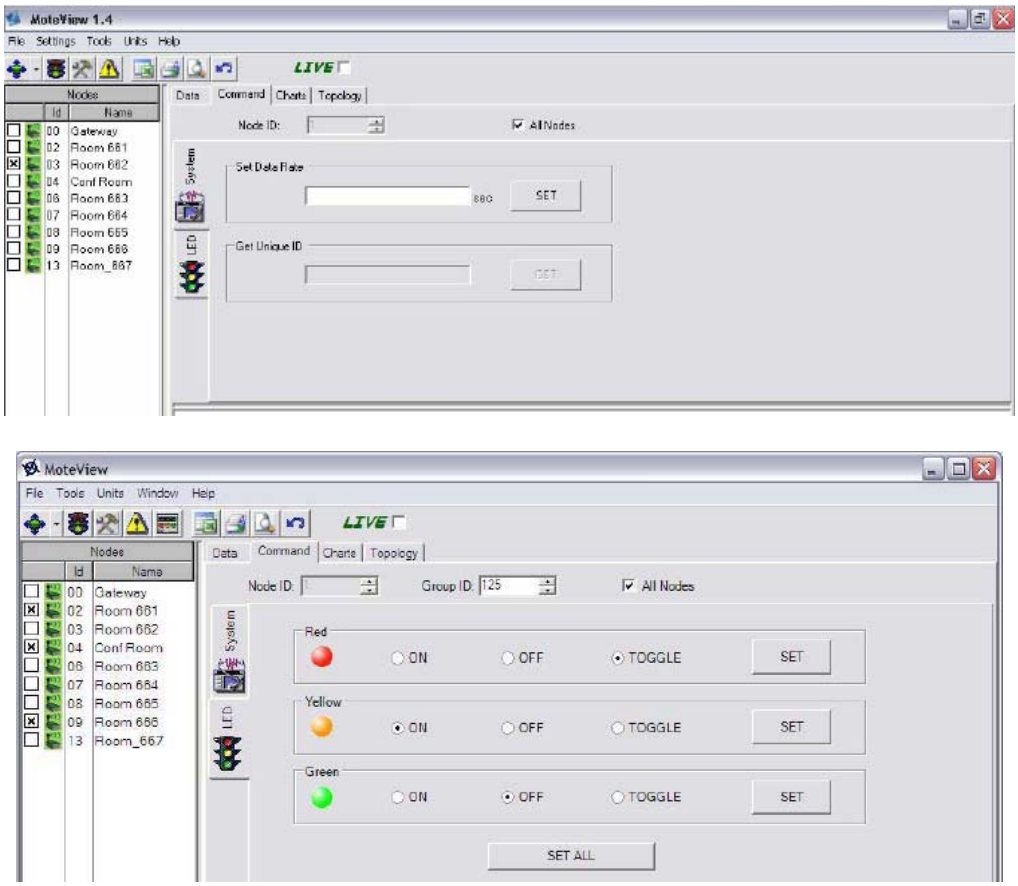


Figura 2.6: Opción Command (Cambio de Tasa de Transmisión y Estado de LED's)

CHART, es la tercera posibilidad de visualizar la información, puesto a que en esta sección se tiene la posibilidad de la generación de tres distintas gráficas, las mismas que muestran uno de los parámetros posibles a ser sensados por los equipos versus el tiempo. Las características de esta sección de la visualización son que se pueden obtener las gráficas de hasta 24 nodos de la red con distinta coloración y con la descripción de los mismos en la parte derecha de la pantalla. De la información, hay que tener en cuenta que la información sensada está en el eje de las ordenadas, mientras que el tiempo (con hora y fecha), en el de las absisas.

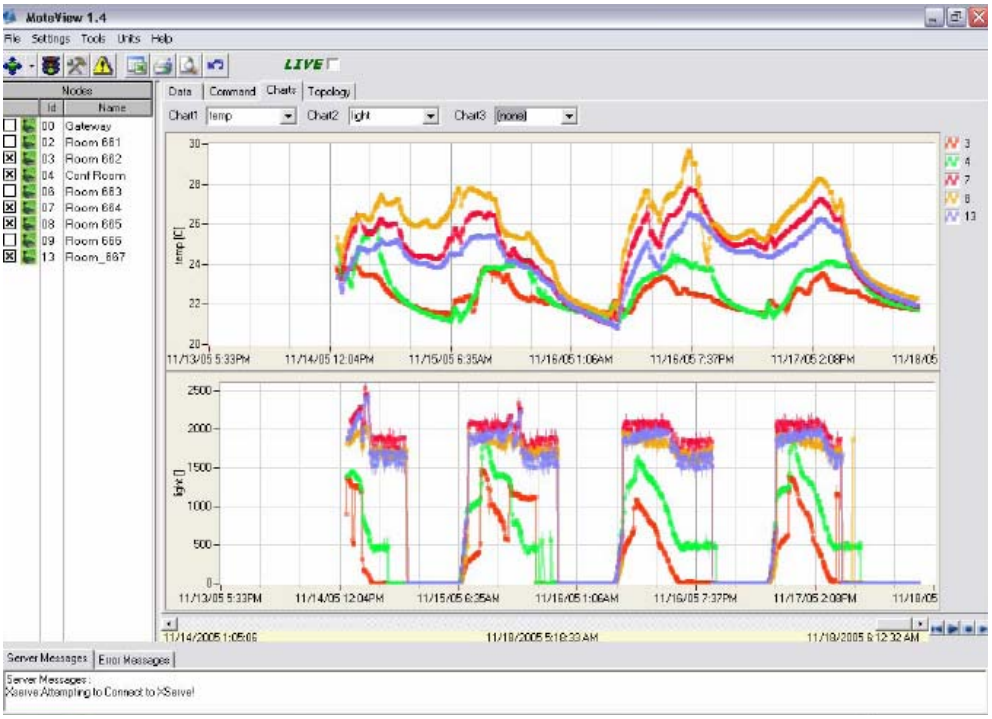


Figura 2.7: Generación de Gráficas en el tiempo

Además de las ventajas de poder obtener una visualización completa para cada uno de los sensores de nuestra red, y con opciones de mostrar hasta tres parámetros de acción, podemos también tener información detallada a través de la realización de ZOOM y PAN, es decir de acercar o alejar por secciones de cada una de las gráficas, así:

Para realizar un Zoom específico sobre una región, simplemente hay que tener presionada la tecla *Shift* y seleccionar el área que deseamos aumentar con el botón izquierdo del Mouse. Para alejar la imagen, es decir, para hacer PAN, se debe presionar la tecla *ctrl.* Y con el botón derecho del Mouse presionar hasta que se desee o se obtenga la resolución por defecto.

Finalmente, pero no por eso menos importante, se debe tener en cuenta la cuarta opción de visualización, que para el caso es ***TOPOLOGY***, esta ventana nos permite

visualizar la distribución de los nodos a través de un área determinada, con posibilidad de obtener una imagen gradiente dependiente de coloración escogida por el usuario y por supuesto, también que depende del parámetro de acción de los sensores escogidos.

La presentación de la pantalla es como se muestra en la Figura.

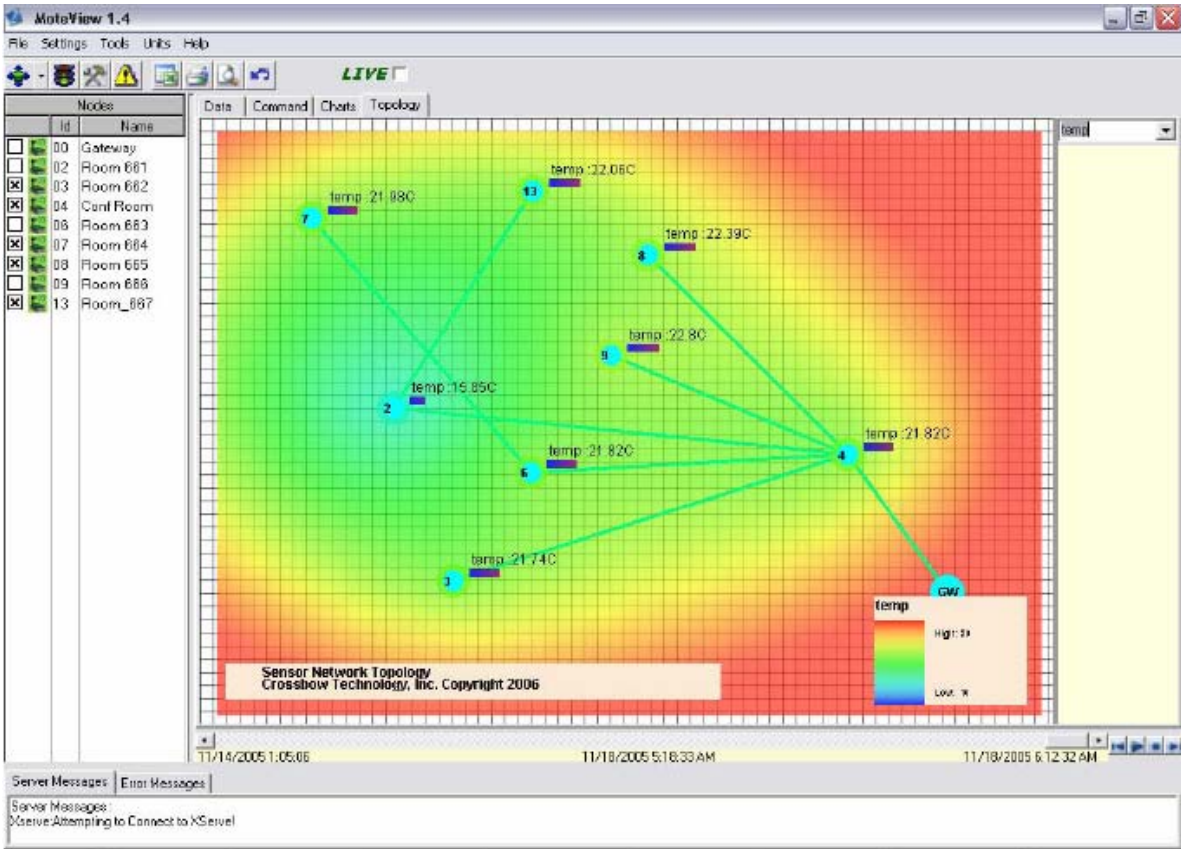


Figura 2.8: Pantalla de Opción TOPOLOGÍA

Ahora, si se observa la Figura mencionada, podemos ver como es que esta opción nos permite tener una idea física de la distribución de los distintos nodos en la Red de Sensores que se disponga, por ejemplo, que para nuestro caso es para el sensamiento de temperatura en un cuarto de Telecomunicaciones, o un cuarto frío.

2.4.1 Formas De Programación

Para la programación se necesita la versión del TinyOs 1.1.0 la cual llega a la actualización 1.1.10 que son las versiones totalmente testeadas por la empresa. A través de esta programación inicial se cubrirá tópicos de programación importantes como instrucciones básicas para compilar y descargar aplicaciones tipo Firmware para los Mote (sensores), etc.

Una vez instalado el TinyOs es necesario reparar un bug que tiene el comando ***make reinstall,*** función muy importante para compilar los programas, la cual forza al build de la aplicación incluso si la aplicación ya ha sido construida ("compilada") para reparar esto es necesario ingresar en el archivo *reinstall.extra* en el directorio *C:\tinyos\cygwin\opt\tinyos-1.x\tools\make\avr*. Esto se realiza removiendo el exe0 de la línea 5, cambiando de la siguiente manera:

```
BUILD_DEPS = exe0 setid program delsetid
```

A esto

```
BUILD_DEPS = setid program delsetid
```

Además, existe una carpeta que no existe durante la instalación del programa TinyOs y es la carpeta xbow que contiene gran cantidad de aplicaciones *Firmware* importantes para los motes. Esto se encuentra en el cd de soporte de TinyOs, o bien se lo puede descargar del Internet de la página de crossbow en aplicaciones, www.xbow.com es necesario que la carpeta se ubique bajo el siguiente directorio *tinyos\cygwin\opt\tinyos-1.x\contrib.*,

La importancia de esta carpeta a mas de tener las aplicaciones para los motes es que permite instalar una utilidad necesaria para los mismos llamada XInstal ya que setea un enlace comúnmente usado para los programas realizados por Crossbow permitiendo programar desde cualquier directorio en Cygwin, además instala un match para programar mediante la interfase USB MIB520 mediante los siguientes comandos.

```
cd /opt/tinyos-1.x/contrib/xbow/bin
./xinstall
```

El ambiente de desarrollo TinyOs requiere del uso de un compilador dentro del sistema operativo *Cygwin.* Primero se necesita saber si la versión del software ha sido instalada correctamente con sus herramientas y si las variables del ambiente están ajustadas. Esto se realiza mediante el script *Toscheck* ubicada en el directorio *opt/tinyos-1.x/tools/scripts/ esta* función se la corre de la siguiente manera:

```
cd c:/<install dir>/tinyos/cygwin/opt/tinyos-
1.x/tools/scripts
toscheck
```

En donde para verificar debe salir la siguiente línea al final del comando una vez que este haya corrido:

```
toscheck completed without error
```

Otra aplicación muy importante es el Serial Forwarder como se hablo anteriormente. Para configurar el Serial Forwarder usando la placa MIB510, los parámetros a introducir son los mostrados a continuación:

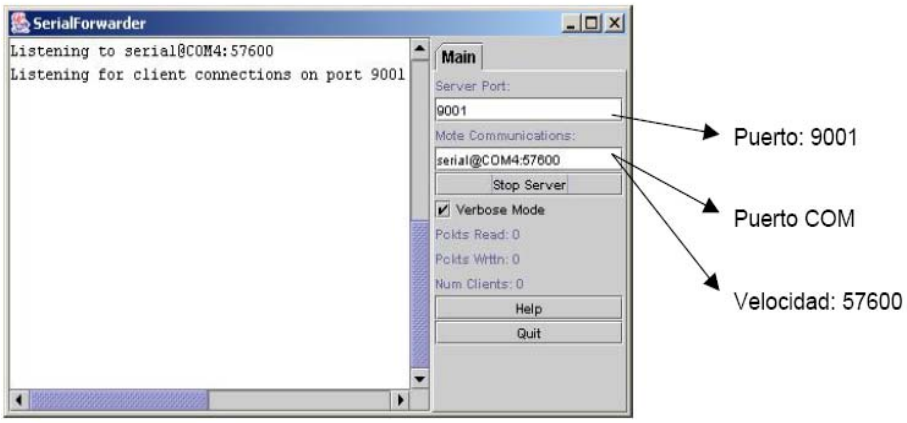


Figura 2.9: Descripción del Serial Forwarder

Cabe resaltar que la capa MAC incorpora un modo promiscuo para la escucha de mensajes incluso cuando el destino final no es el nodo local. Gracias a esta característica el nodo "base" puede monitorizar todos los paquetes que le lleguen dentro de su rango de recepción.

2.5 SIMULACIONES

Una vez revisado el software de gestión y programación de los sensores inalámbricos es importante tomar en cuenta cuales son los algoritmos de programación principales para programar las opciones básicas de los WSN y poder realizar así mismo la gestión de la red. A continuación se verán configuraciones básicas y algoritmos para la programación de sensores empleando el sistema operativo TinyOs.

2.5.1 Los Algoritmos

Las WSN's se componen de una gran cantidad de nodos, por lo tanto, un algoritmo para un WSN es implícito un algoritmo distribuido. En WSNs el recurso más escaso es energía, y uno de la operación energía-más costosa es transmisión de datos. Por esta razón, la investigación algorítmica en WSN se centra sobre todo en el estudio y el diseño de los algoritmos enterados de la energía para la transmisión de datos de los nodos del sensor a las estaciones de bases. La transmisión de datos es generalmente multi-hop (de nodo al nodo, hacia las estaciones bajas), debido al crecimiento polinómico en la energía-coste de transmisión de radio con respecto a la distancia de la transmisión.

El acercamiento algorítmico a WSN se distingue del acercamiento del protocolo por el hecho de que los modelos matemáticos usados son más abstractos, más general, pero a veces menos realista que los modelos usados para el protocolo diseñan.

- **Seteando alias**

Una vez instalado exitosamente el TinyOs, es recomendado setear los alias para comandos usados comúnmente y así tener acceso a los directorios. Los alias necesitan ser editados al final del archivo llamado profile el cual esta ubicado en *<install dir>/tinyos/cygwin/etc/*.

Estos alias son necesarios para cambios rápidos hacia los directorios más importantes del Shell dentro de Cygwin. Estos alias pueden ser seteados para el cambio fácil de directorios y no son los únicos que pueden existir, el formato para realizar los alias es representado con el código siguiente:

```
    alias cdtinyos="cd c:/<install
dir>/tinyos/cygwin/opt/tinyos-1.x"
    alias cdjava="cd c:/<install
dir>/tinyos/cygwin/opt/tinyos-1.x/tools/java"
    alias cdxbow="cd c:/<install
dir>/tinyos/cygwin/opt/tinyos-1.x/contrib/xbow"
```

- **Compilando aplicaciones Tiny Os**

TinyOs soporta las plataformas hardware Mote de procesamiento de radio (MPR) MICAz, MICA2 y MICA2DOT. La sintaxis para compilar (building) de un código para una aplicación es de la siguiente manera:

```
make <platform>
```

El nombre usado para <platform> puede ser encontrado en la siguiente tabla:

Tabla 2.3: Listado de las plataformas Hardware

Processor/Radio Platform	For <platform> use
MICAz (MPR2400 series)	micaz
MICA (MPR3x0 series)	mica
MICA2 (MPR4x0 series)	mica2
MICA2DOT (MPR5x0 series)	mica2dot

- **Programando las Interfaces**

El ambiente de desarrollo soporta una gran variedad de herramientas para programación. Los nombres de estos tableros interfaces son:

- MIB500 plataforma de programación por puerto paralelo.
- MIB510CA plataforma de programación por puerto serial.
- MIB520CA plataforma de programación por puerto USB.
- MIB600CA plataforma de programación por Ethernet.

El software de programación estándar usado en TinyOs es el Micro In-System Programmer o UISP. Este programa, que viene como parte del TinyOs, toma varios argumentos de acuerdo al hardware programador y las acciones particulares deseadas (borrar, verificar, programas, etc). Para simplificar usando esta herramienta, el ambiente TinyOs llama al UISP con los argumentos adecuados cuando se llama un comando **`install`** o **`reinstall.`** También es necesario especificar el dispositivo con el cual se va a comunicar. Esto se realiza mediante el uso de variables de entorno.

Tabla2.4: Listado de las plataformas tipo interfaz Mote ("MIB") plataformas de programación (<programmer>)

MIB Board	For *<programmer>* use
MIB600	`eprb`
MIB510	`mib510`
MIB520	`mib520`
MIB500	Default, no additional information needed

Todas las aplicaciones bajo TinyOs usan la sintaxis make. un ejemplo de esta sintaxis se encuentra bajo el siguiente directorio tinyos-1.x/tools/make.

Ejemplo:

El makefile usado por las aplicaciones ubicado en el directorio tinyos/cygwin/opt/tinyos-1.x/contrib/xbow/

```
    # $Id: Makefile,v 1.6 2004/08/06 09:30:24 husq Exp $
    XBOWROOT=%T/../contrib/xbow/tos
    COMPONENT=TestSensor
    SENSORBOARD=mts310
    PFLAGS=          -I$(XBOWROOT)/interfaces          -
I$(XBOWROOT)/system  -I$(XBOWROOT)/platform/$(PLATFORM)  -
I$(XBOWROOT)/lib                                        -
I$(XBOWROOT)/sensorboards/$(SENSORBOARD)
    include ../MakeXbowlocal
    include ${TOSROOT}/tools/make/Makerules
```

- ***MIB500/ Programador de Puerto paralelo***

Este es el dispositivo de programación por defecto. No son necesarios parámetros adicionales para los comandos de línea cuando se utiliza este programador. El comando por defecto es **`make <platform> install.`**

- ***MIB510/ Programador de Puerto Serial***

Tomando en cuenta la línea comando con *mib510,com<x>* donde x es el número de puerto serial en el cual el MIB510 esta conecto. Antes de correr este comando chequea el sistema propio verificando que puertos están disponibles.

Ejemplo:

Programación en MICAz para un MIB510 el cual es conectado al puerto serial por el puerto com1

```
make micaz install mib510,com1
```

- ***MIB520/ Programador de Puerto USB***

El MIB520 emplea FTDI FT2232C para usar USB como puerto COM virtual. Así de esta manera es necesario instalar los drivers FT2232C VCP ubicados en el cd de soporte de TinyOs. De esta manera el comando para el MIB520 será el mismo empleado para el MIB510, una vez que se sabe cual es el nuevo puerto virtual a ser empleado.

Ejemplo:

Programación en MICAz para un MIB520 el cual es conectado al puerto serial por el puerto com7

```
make micaz install mib520,com7
```

- ***MIB600/ Programador de Ethernet***

Para emplear el programador MIB600 plataforma de programación Ethernet es necesario asignar una dirección IP al dispositivo. Cada dispositivo es conectado a una red IP que debe poseer una dirección IP única. Esta dirección es usada como referencia para la unidad especifica. Cada conexión TCP y cada datagrama UDP es definido mediante una dirección IP destino y un número de puerto.

1. Instalación el dispositivo Lantronix (deviceinstaller36.zip) del CD ROM dentro de la carpeta Miscellaneous, o descargar de la pagina www.lantronix.com.
2. Conectar el MIB600 a la red mediante el uso del cable de Ethernet RJ-45, de esta manera se busca la dirección del hardware q coincida al MIB600 ingresando al programa Device Installer de Lantronix (p.e. 00-20-4ª-63-47-31) se la selecciona y se asigna una dirección IP.
3. Una vez asignada la dirección IP en el MIB600, la línea de comando dentro de la ventana Cygwin para el Mote es:

```
make <platform> install eprb,<IP_Address_of_MIB600>
```

Ejemplo:

Programación en MICAz para un MIB600 el cual esta conectado mediante ethernet y es asignado con una dirección IP 192.168.100.123

```
make micaz install eprb,192.168.100.123
```

2.5.2 Instalación de aplicaciones en un Mote

Las herramientas de programación también incluyen un método de programar direcciones de los nodos únicas sin tener que editar El ambiente de desarrollo directamente. Para setear las direcciones ID de los nodos durante la carga del programa, la sintaxis general de instalación es:

```
make <platform> re|install,<n> <programmer>,<port>
```

Donde <programmer>,<port> es el nombre del programador, la dirección de puerto ID o numero de host PC en la cual el programador esta conectado, <n> es un numero opcional en el cual se setea el nodo o dirección, y <platform> el cual es el tipo de plataforma MPR, las cuales fueron descritas anteriormente. A continuación se da una pequeña explicación a cerca de los comandos *install* y *reinstall:*

Install,<n> .- compila la aplicación para una plataforma asignada, setea los nodos dirección/ID y programa el dispositivo (Mote).

Reinstall,<n> .- setea los nodos dirección/ID y descarga el programa precompilado solo (dentro del mote), y no recompila. Esta opción es significativamente rápida.

- **Seteando el Grupo ID y la dirección de nodo para la red de sensores Mote.**

Los mensajes de TinyOs contienen un grupo ID en la cabecera, lo cual permite a diferentes grupos de Motes a compartir el mismo canal de radio, es necesario setear el grupo ID a un valor único de 8-bits para permitir a TinyOs filtrar de salida dichos mensajes. El grupo de salida por defecto es 0x7d. Se puede setear el grupo ID definiendo el símbolo preprocesado **`DEFAULT_LOCAL_GROUP`** dentro del archivo *MakeXbowLocal* el cual esta ubicado en el directorio *tinyos-1.x/contrib./xbow/apps/.*

Además el encabezado contiene información como numero de nodo el cual es un valor de 16-bits.

Exceptuando por el valor decimal **126** (el `TOS_UART_ADDR: dirección de transmisión UART 0x007E`) y 255 (el `TOS_BCAST_ADDR: direccion de transmisión de broadcast 0xFFFF`), el resto de valores entre 0,254 son permisibles. El numero 0 es típicamente reservado para el Mote que funciona como estación base. Setear la direccion de nodo es importante cuando se programa los motes para una red de sensores, la línea de comando es:

`make <platform> re/install,<n> <programmer,<port>`

Ejemplo:

Asignación de nodo dirección/ID de 38 al MICAz. El MIB510 está en el PC ubicado en el puerto serial com1.

`make micaz install,38 mib510,com1`

- **El Archivo MakeXbowlocal**

El archivo MakeXbowlocal fue desarrollado por Xbow para ser usado en aplicaciones dentro del directorio contrib/xbow/apps/. Esto provee una manera conveniente para los usuarios para cambiar el ID local del grupo, canal (frecuencia de RX/TX) y la potencia de transmisión en RF. Debe ser usado en conjunto con el archivo CC1000Const.h ubicado en opt/tinyos-1.x/contrib/xbow/tos/platforms/mica2/. para usarlos es necesario chequear que el Makefile en una aplicación particular tiene la siguiente línea:

```
include ../MakeXbowlocal.
```

Incluyendo dicha línea en las aplicaciones Makefile provocara al compilador incluir el archivo MakeXbowLocal. El archivo será adjuntado a los anexos donde se tiene los contenidos del MakeXbowLocal y donde para realizar los cambios necesarios se puede cambiar, el grupo local, el canal, la potencia de transmisión RF, el número de puerto del programador.

2.5.2 Frecuencias de Radio

Los receptores de radio en el MICAz, MICA2, y MICA2DOT soportan múltiples frecuencias. Las unidades en un canal predefinido son repartidas en cualquiera de las bandas ISM de 315MHz, 433MHz, 915MHz, o 2.4GHz. Todos los coeficientes para la sintonización de radio para el MICAz y MICA2DOT están contenidos en el archivo *CC1000Const.h* de TinyOs ubicado en *tos/platform/mica2/.*

Los usuarios deben compilar en la frecuencia de la estación base correcta porque de otra forma la comunicación va a fallar. La mejor y mas segura manera de estar seguro que se esta compilando dentro de la frecuencia correcta para cada plataforma Mote, es editando el archivo MakeXbowlocal.

Para usuarios de Mica2 y Mica2Dot, a veces la frecuencia de interés no esta disponible en dicho archivo por lo cual es necesario realizar la siguiente recomendación:

Adherir una línea de texto CFLAGS con el siguiente formato: **`CFLAGS=-DCC1K_DEF_FREQ=<Desired_Freq>.`** Dentro de una aplicación particular de *Makefile.* El valor en <desired_freq> es la frecuencia valida en Hz. además es necesario chequear y quitar el comentario (“#”) de las frecuencias en el archivo MakeXbowLocal.

Específicamente en el uso de aplicaciones en el directorio *tinyos/opt/cygwin/tinyos-1.x/contrib/xbow/apps/.*

En el caso de usuarios de MICAz, la sintaxis de CFLAGS es estrechamente diferente. Todos los 16 canales de 802.15.4 en la banda de 2.4Ghz están ubicados en el archivo MakeXbowLocal, en vez de ello se puede usar;

CFLAGS=-DCC2420_DEF_CHANNEL=*<Desired_Channel>*

Algunas veces la frecuencia de interés no esta disponible en el archivo MakeXbowlocal el cual fue desarrollado por Xbow para ser usado en aplicaciones dentro del directorio contrib/xbow/apps/. Esto provee una manera conveniente para los usuarios

Ejemplo:

Adhiriendo una sentencia CFLAGS al final de CntToLedsAndRfm en el archivo Makefile para setear el canal de radio de 802.15.4 dentro del canal 15.

```
COMPONENT=CntToLedsAndRfm
PFLAGS=-I%T/lib/Counters
XBOWROOT=%T/../contrib/xbow/tos
PFLAGS += -I$(XBOWROOT)/platform/mica2
include ../MakeXbowlocal
include $(TOSROOT)/tools/make/Makerules
CFLAGS=-DCC1K_DEF_CHANNEL=15
```

2.5.3 Visualización de los Datos

Los datos recopilados de redes sin hilos del sensor se ahorran generalmente bajo la forma de datos numéricos en una estación baja central. Hay muchos programas, como TosGUI y MonSense, GSN que facilitan la visión de estas cantidades grandes de datos.

Además, el consorcio abierto de Geospatial (OGC) está especificando los estándares para los interfaces de la interoperabilidad y las codificaciones de los meta datos que permiten la integración en tiempo real de las telas heterogéneas del sensor en el Internet, permitiendo que el individuo supervise o controle redes sin hilos del sensor con un web browser.

CAPITULO 3

ESTUDIO DEL HARDWARE (SENSORES INALAMBRICOS)

3.1 INTRODUCCION

Dentro del desarrollo del presente capítulo se podrá tener acceso a cada una de las características y bondades que nos ofrece la parte física de la implementación del proyecto, es decir, en esta sección del desarrollo del documento, se detallará las utilidades, características y principios de funcionamiento de cada uno de los componentes implícitos en la implementación planteada para la culminación del proyecto, es así que se va a trabajar en el desarrollo de esta sección sobre los siguientes componentes:

Tabla 3.1: Listado de Componentes de la Red Sensorial

	COMPONENTE	DESCRIPCIÓN
A	MPR2400	Sensor de Temperatura, Luz, Aceleración
B	MIB510	Tablero de Programación con Interfaz Serial RS-232
C	MIB520	Tablero de Programación con Interfaz USB
D	MIB600	Tablero de Programación con Interfaz Ethernet

De tal forma, a continuación se procederá al detalle, análisis y explicación del funcionamiento de cada una de las características de los componentes antes mencionados en la *Tabla3.1* para la verificación de la parte física, es decir del Hardware del Sistema.

3.2 MPR2400

Figura 3. 1: Mote MPR2400, Sensor Inalámbrico con Antena Estándar

El sensor físico MPR2400, de las siglas en inglés "*Mote Processor Radio*", justamente por ser la última generación de Sensores de variables disponibles creada por la empresa CrossBow para el mercado, conocidos, a nivel industrial y de implementación como **MICAz**. El MPR2400 trabaja a frecuencias de entre los 2400 MHz hasta los 2483.5 MHz, una banda bastante óptima para el uso de ambientes industriales e incluso, para aplicaciones del hogar como seguridad o anticipación de desastres. Entre sus principales características de funcionamiento tenemos que:

a) Usa el *Chipcon* CC2420,

b) Basado en IEEE 802.15.4,

c) Posee integrado un emisor/receptor de radio frecuencia ZigBee con un Microcontrolador Atmega128L,

d) Un conector de 51 pines de tipo I/O,

e) Y una memoria flash, lo que lo hace compatible con cualquier versión anterior de software usados para los MICA2.

3.2.1 Diagrama de Bloques

Dentro de la configuración externa y de funcionamiento de los MPR2400, podemos encontrar una serie de etapas de orden necesario y coordinado, todo para el correcto funcionamiento, así:

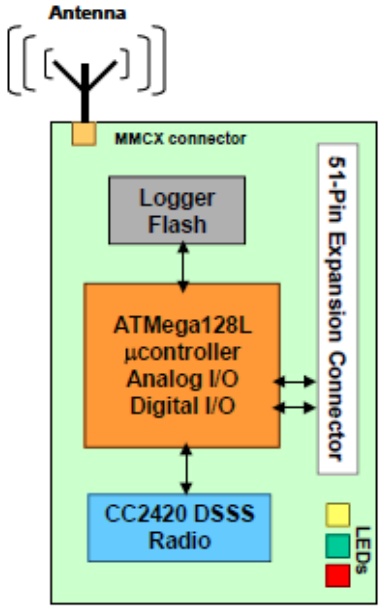


Figura 3. 2: Diagrama de Bloques de MPR2400[2]

Donde es claro observar, cada una de las características de los Sensores, comenzando por una **Antena,** la misma que permite la comunicación de los Sensores Secundarios con el Sensor Base o Estación, bajo las normas antes mencionadas. Se debe tener mucho cuidado al momento de dotar de una antena a un dispositivo, ya que de esto depende el correcto funcionamiento del mismo, así como el aprovechamiento de sus máximas capacidades. El Rango y su operación son grandemente afectados por la elección de una antena y su colocación en un ambiente adecuado. Además, en cuanto a este mismo tema, se debe tener en cuanta que hay que cumplir con ciertas especificaciones y normas que constan pata el cumplimiento del artículo 15 de la regulación de radiadores intencionales de la FCC. A causa del pequeño tamaño de los dispositivos, lo usual es una antena de alambre aislado de cuarto de onda para la frecuencia de interés. Este tipo de antenas es comúnmente llamada antena de monopolo, y su ganancia es dependiente del plano de tierra. La antena deberá tener un Terminal coaxial y un conector del tipo MMCX, con un plano de tierra que puede estar recubierto de Aluminio como se muestra en la siguiente Figura.

[2] MPR – MIB_Series_User_Manual.pdf

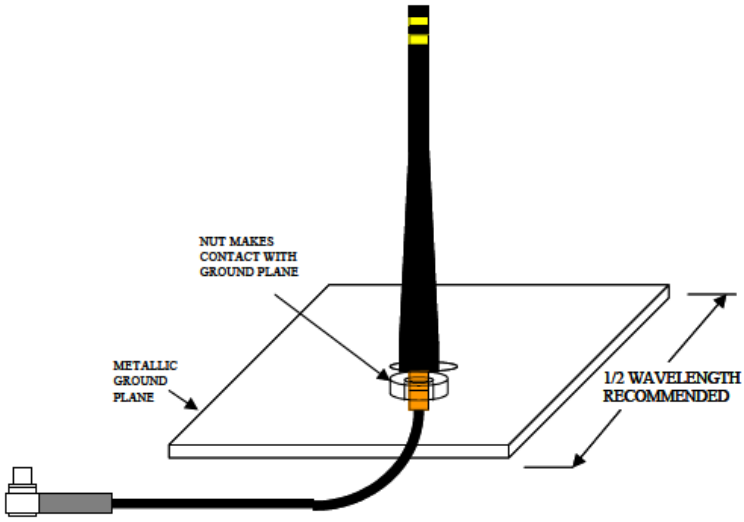


Figura 3. 3: Esquema de Conexión de Antena

El conector **MMCX**, un estándar de conexión de la antena para este tipo de tecnología del cual se comentará más adelante en el documento. Las generaciones de Sensores MICA2 y MICAz tienen un conector de este tipo para adjuntar la conexión de una antena externa. Básicamente, existen dos proveedores de este tipo de conectores Johnson Components y Hirose Electric Ltd. Una de sus principales ventajas es que este tipo de conectores soportan dos estándares esenciales de cable coaxial, el RG178 /U y el RG 316/U.

La memoria **Flash**, con una capacidad de 4 Mbit la que se encuentra adjunta al microcontrolador para hacer compatible al Sensor con versiones anteriores de programadores como MICA2; a continuación se puede observar justamente el Microcontrolador con entradas/salidas tanto análogas como digitales; y que se encuentra con posibilidades de almacenar Mediciones del Usuario, Datos y otro tipo de información definida por el mismo usuario. Este chip es soportado por TinyOs, el cual lo usa como un micro archivo del sistema. La capacidad de este dispositivo es de hasta 100,000 mediciones o lecturas y es esencial para la programación Sobre-Aire, una de las capacidades ofrecidas por TinyOs.

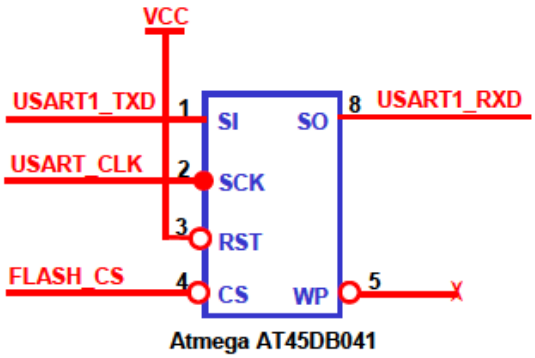
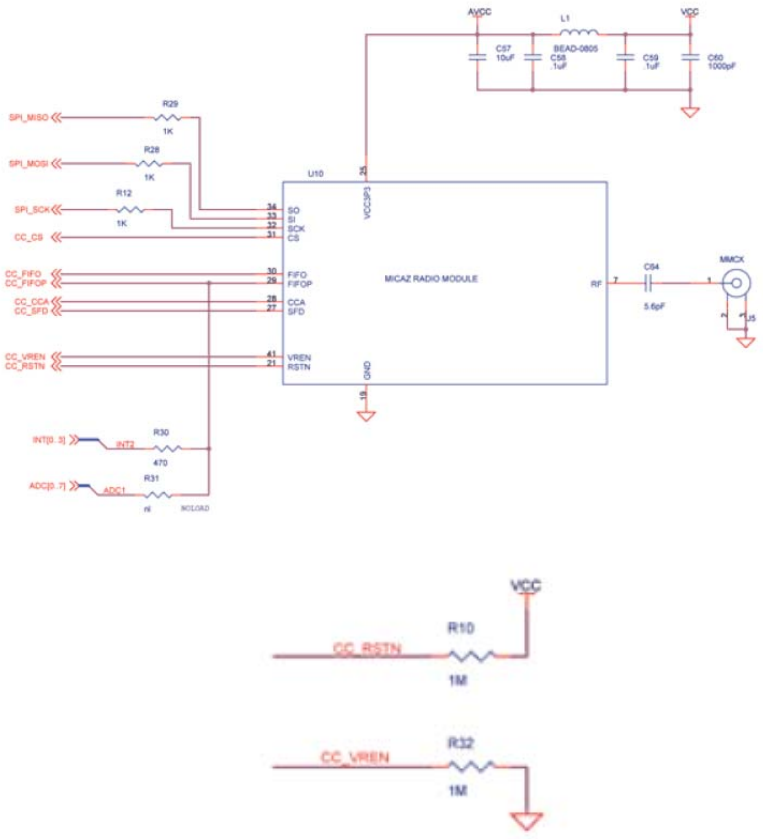


Figura 3. 4: Memoria Flash

En el cuadro azul, se tiene justamente el Tablero de Radio **CC2420DSS**, cuya utilidad es la de encargarse en su configuración de la parte concerniente al procesamiento de la información para la comunicación del sistema, es decir, que este bloque es el encargado del éxito de la transmisión y recepción de datos a través y desde el MPR2400, en su estructura se presenta una serie de componentes que permiten la configuración de los parámetros de comunicación para posteriormente interactuar con otros componentes como el mismo Microcontrolador o la Antena del Sensor.



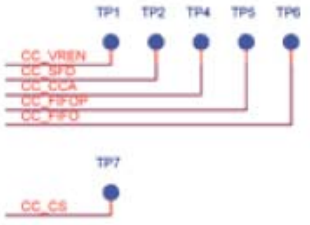


Figura 3. 5: Diagrama de Conexiones del Módulo de Radio CC2420DSS[3]

El conector de 51 Pines, es esencial para la interacción con la parte de la programación del Sensor, es decir, es la parte de la conexión Física entre el Tablero de Programación de cualquiera de las interfaces y el MPR2400, un conector de Expansión implementado en el Sensor para interacción con los sistemas de programación.

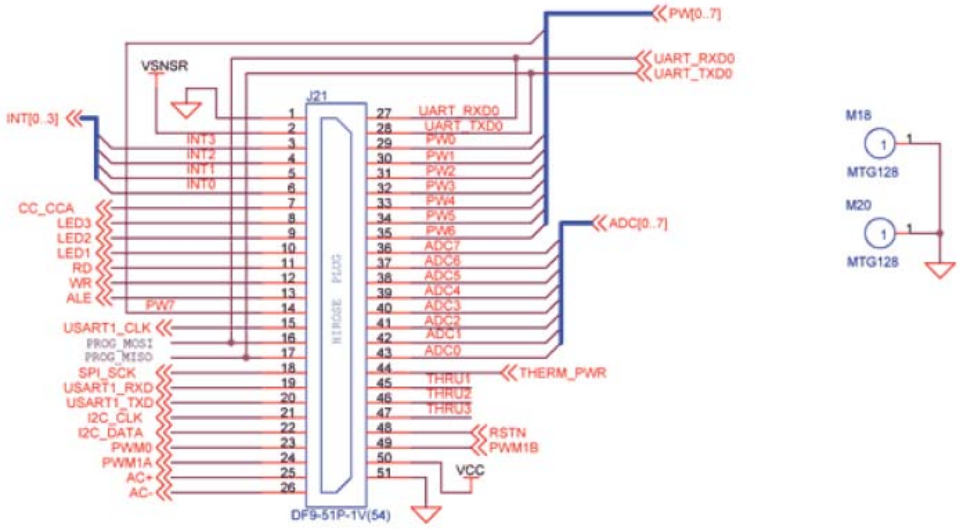


Figura 3. 6: Pines del Conector de Expansión[3]

Finalmente, una parte primordial para el funcionamiento del MPR2400, la alimentación del dispositivo, que por ser una tecnología meramente inalámbrica, es una conexión a baterías de poder, baterías 2AA que permiten la operación del equipo de forma independiente.

[3] MPR – MIB_Series_User_Manual.pdf

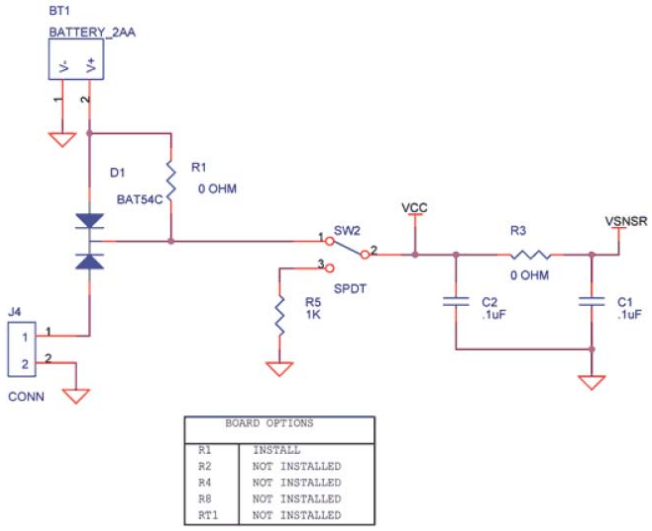


Figura 3. 7: Esquema de Alimentación del MPR2400

3.3 MIB510

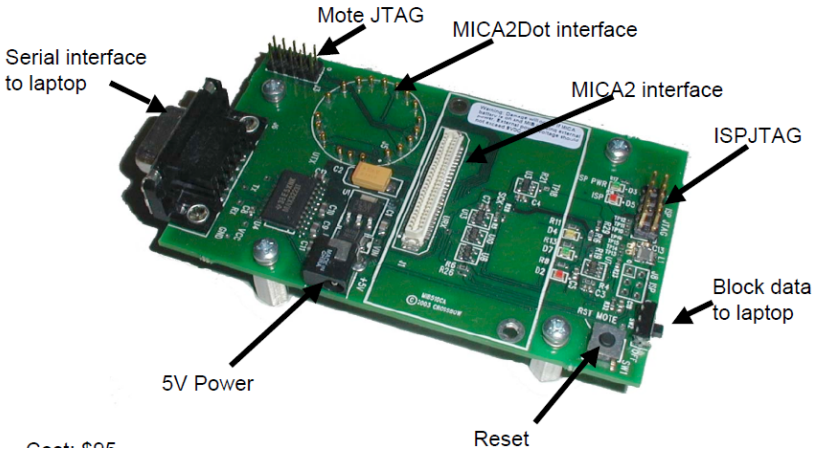


Figura 3. 8: Placa del Módulo de Programación MIB510

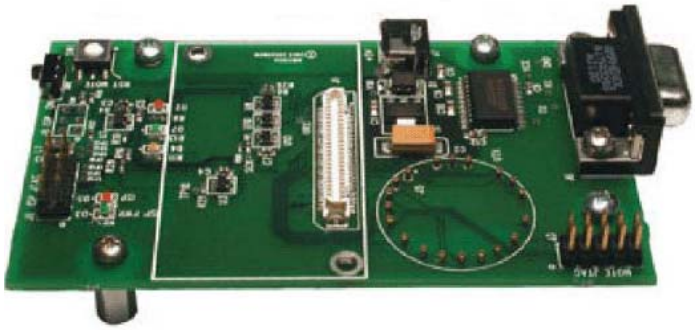


Figura 3. 9: Vista Superior de Placa del MIB510

La placa de desarrollo MIB510 actúa como interfaz entre el PC y los motes a través del puerto serie, permitiendo la programación de los dispositivos acoplados. La placa MIB510 tiene un procesador (ISP) Atmega16L mediante el cual se programan los motes. El código se descarga al ISP a través del puerto serie RS-232, y es el ISP el que programa el código en el mote. Para programar los motes se ha de tener instalado en el PC el sistema operativo TinyOS, descrito en la sección dedicada al Software. Esta placa tiene conectores para motes MICAz, MICA2 y MICA2DOT.

El ISP corre bajo condiciones normales a una tasa de baudios de 115.2 kbaudios, y principalmente de lo que se encarga es de continuamente monitorear los paquetes provenientes del puerto serial para la organización de una forma especial de modelado multi-byte. Una vez que este modelo es detectado, éste deshabilita tanto la transmisión como la recepción de datos o información a través del serial de los MOTES, luego toma control de dicho puerto.

3.3.1 Programación de MOTES usando MIB510

En primer lugar, para la programación de los MOTES a través de este tablero de programación, se debe tener en cuenta la necesidad primordial de la instalación de TinyOs en la PC que va a ser nuestro servidor o cliente dentro de los parámetros de red planteados.

Luego, la operación de la placa MIB510 dependerá, en cuanto al software, a la versión de MOTE con la que se trabajará, en el caso del presente proyecto, por ejemplo, será a través de los MPR2400 (MICAz) antes mencionados, de tal forma que la manera de crear, compilar y bajar programas a los módulos, será precisamente una expresa fórmula de las versiones de equipos del sistema a ser implementado.

3.3.2 Interfaces para MICAz, MICA2, y MICA2DOT

Básicamente, en su estructura, el MIB510 tiene conectores para los tres modelos o versiones de equipos (MICAz, MICA2 y MICA2DOT). En el caso especifico de los MICAz y MICA2 existe otro conector en uno de los extremos finales de la placa que permite la conexión simultánea de los sensores junto con los MOTES. Para la versión de

MICA2DOTs con adaptadores de batería, también se los puede implementar sobre uno de los conectores adaptados de la placa del MIB 510.

3.3.3 Reset

El botón "RST MOTE" lo que hace, es justamente, provocar un reset tanto del ISP como del procesador del MOTE. Es una operación secuencial, esto debido a que al presionar dicha tecla, en primer lugar se resetea el ISP, y a continuación de esto, cuando se enciende de nuevo el ISP, resetea el procesador del MOTE.

3.3.4 JTAG

La placa del tablero de programación, el MIB510, cuenta con un conector J3 ("MOTE JTAG"), el cual permite la conexión al JTAG del microcontrolador Atmel para la generación de una depuración dentro del circuito mismo, es decir a nivel de Hardware. Este conector permite la alimentación a la entrada JTAG, por lo que no es necesaria una fuente de alimentación externa.

Pero se debe tener en cuenta que la placa viene con un conector JTAG para el procesador ISP, pero solamente de uso de los fabricantes, por lo que hay que tener cuidado con su uso inadecuado.

3.3.5 Alimentación

La placa del MIB510 tiene en su estructura una regulador de tensión para poder trabajar con su propia alimentación nominal con las opciones de operar rangos de entre 5 a 7 VDC, además, que por medio de la alimentación que recibe, logra también proporcionar una tensión nominal regulada de 3 VDC a las versiones MICAz, MICA2, y MICA que sean conectadazas a la misma para la operación con las distintas interfaces. Es por ello que el MIB510 se encuentra proporcionado con una pared de poder para su protección y de los equipos que sean conectados a la misma.

Además, podemos destacar de la conformación del MIB510, la existencia de un monitor de baja tensión que permite deshabilitar la reprogramación en caso de que los rangos de voltaje de alimentación sean peligrosamente bajos para la correcta operación del

equipo. A manera de aviso, cuando existe un adecuado nivel de voltaje de programación, se enciende un LED denominado con “ISP PWR”. Si el valor de voltaje de alimentación va hasta por debajo de 2.9 V, entonces el mencionado LED (“ISP PWR”) de color verde, comenzará a parpadear y entonces el circuito procede a la deshabilitación de cualquier intento de descargar programa alguno a la plataforma instalada, y en caso de que el voltaje sea muy por debajo del nivel necesario, entonces el LED dejará de parpadear y se apagará por completo, evitando así la operación completa de la placa de por sí.

3.3.6 Interfase RS-232

La interfase RS-232 es un estándar de comunicación de canal único bi-direccional con un conector DB9 que va a permitir la conexión con un puerto externo del Computador. Éste usa simplemente, para el caso de la aplicación presente, las líneas de transmisión y recepción.

3.4 MTS300/310

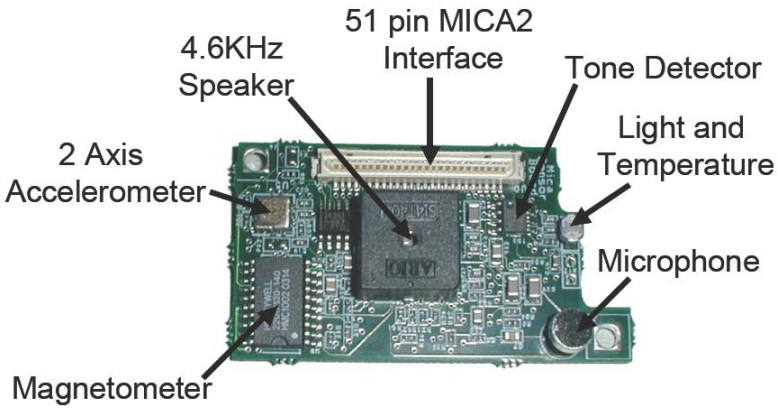


Figura 3. 10: Vista Superior de Placa del MTS310

Incluidos en el Kit de desarrollo, y con total compatibilidad con los módulos Motes 2400, tanto los MTS300CA como los MTS310CA y los MTS300CB/MTS310CB, son parte del análisis de esta sección, esto debido a que solamente algunas pequeñas diferencias son las que se pueden denotar entre los dispositivos nombrados, fuera de eso, su operación es muy semejante.

Por lo tanto, para el MTS300 y MTS310 se puede especificar características como la *flexibilidad*, es decir que son tableros de sensamiento muy flexibles, esto debido a la gran variedad de magnitudes posibles a ser medidas o captadas. Dichas modalidades pueden ser fácilmente explotadas en redes sensoriales de desarrollo por su capacidad de cambiar la variable que miden, de tal forma que se puede emplear en aplicaciones como detección de vehículos, anticipación sísmica de bajo nivel, movimiento, acústica, robótica, entre otras aplicaciones.

Debido, justamente a este amplio gama de aplicaciones, es que esta sección de a continuación está meramente dedicada al análisis y explicación, así como reconocimiento de cada una de las características y componentes del circuito que hacen posible dichas operaciones y bondades del MTS300/310.

3.4.1 Micrófono

Principalmente, el circuito del micrófono tiene dos operaciones básicas, en primer lugar, permite reconocimiento de amplitudes y distancias a través de rangos acústicos; y la segunda operación es para grabado y medición de audio. El circuito básico del micrófono comprende un Pre-amplificador (U1A-1), un amplificador de segundo nivel con control de variación de resistencia digital (U1A, PT2).

En sí, lo que hace el circuito es amplificar la baja salida del micrófono, dicha salida puede ser alimentada directamente dentro del convertidor Análogo/Digital (ADC) simplemente usando el selector "*Microphone Output*" (MX1) para conectar la salida del micrófono a la señal del ADC. Esta configuración es muy útil dentro de aplicaciones generales de grabados acústicos y mediciones de la misma naturaleza. El Audio grabado en archivos es almacenado en la memoria Flash del modulo MICAz, MICA2, o MICA para más tarde ser descargados en la PC para su uso final, dependiendo de las características del proyecto.

La salida del micrófono de la segunda etapa (*mic_out*) es encaminada a través de un filtro activo (U2) y luego dentro de un detector de tonos (TD1). El circuito integrado utilizado como detector de tonos, el LM567 CMOS, convierte a la señal de salida análoga

del micrófono en una salida de bajo nivel de tipo digital a INT3 cuando se encuentra presente un tono de 4 kHz. El circuito *Sounder* es el que puede generar en su estructura el mencionado tono.

La más práctica de las formas de usar el circuito *sounder* y el detector de tonos es en aplicaciones de mediciones de distancia acústica, es decir que el MOTE remoto lo que hace es generar pulsos auditivos y enviar un paquete RF vía radio al mismo tiempo, luego, el MOTE base o la estación base, recibe dicha señal y a través del cálculo del tiempo de llegada del mensaje, se puede estimar con bastante precisión de por medio la distancia de la cual fue emitida la señal. A través del incremento de un contador en el procesador de la Base, se obtiene la señal de onda del audio así como el *tiempo de vuelo* de la misma, tiempo que puede ser transformado en la distancia aproximada de la que ya se habló.

3.4.2 Sounder

Simplemente es un generador resonador piezoeléctrico de una frecuencia simple de 4 kHz. La circuitería del manejo y control de frecuencia se encuentra construida dentro de dicho resonador. La única señal que requiere para ser encendido o apagado, es la de alimentación (*Sounder_Power*), señal que es controlada a través de un interruptor de poder (P1) y ajustada por la línea de hardware PW2.

3.4.3 Luz y Temperatura

Los sensores de luz y temperatura comparten el mismo canal de conversión del ADC1. Solamente se puede encender uno de los dos al tiempo, debido que al encender los dos al mismo tiempo, las medidas serían afectadas con errores falsos y bastante significativos. Tanto el MTS300 como el MTS310 tienen un sensor de luz y un termistor en su estructura.

El sensor de luz es una simple fotocelda de tipo CdSe. La máxima sensibilidad de la mencionada fotocelda es a la longitud de onda de la luz de 690 [nm]. La resistencia típica, mientras se expone a la luz, es de 2 kΩ; mientras que mientras se mantiene en la oscuridad es de 520 kΩ. En orden de utilizar el sensor de luz, la señal de control digital debe ser encendida a través de PW1. Luego, la salida del sensor deberá ser conectada al canal 1 del

convertidor análogo – digital (ADC1). Mencionadas las condiciones de resistencia, frente a la luz, la salida del circuito nominal es cercana al valor de voltaje de la alimentación (VCC) o de escala-completa, mientras que cuando se encuentra en condiciones de oscuridad, el valor de la salida del circuito es de case cero o cercano a GND. La potencia es controlada al sensor de luz a través de la señal INT1.

Por otro lado, el termistor (Panasonic ERT-J1VR103J) en el MTS300 y en el MTS310 es un área montada sobre el espacio correspondiente a RT2. Éste es configurado de tal forma que se lo puede comparar con un simple divisor de voltaje con una lectura nominal de media escala de hasta 25°C. La salida del sensor de temperatura se encuentra disponible dentro de la composición del circuito en el conversor ADC1.

Para el MTS300CA y el MTS310CA, el voltaje de alimentación del termistor es controlado a través del ajuste de la señal INT2.

Para el MTS300CB y el MTS310CB, el voltaje de alimentación del termistor es controlado a través del ajuste de la señal PW0.

Tabla 3. 1: Voltaje, Resistencia Vs. Temperatura

Temperatura (°C)	Resistencia (Ohms)	ADC1 (% of VCC)
-40	427,910	2.3%
-20	114,200	8.1%
0	35,670	22%
25	10,000	50%
40	4090	71%
60	2224	82%
70	1520	87%

- **Conversión a Unidades de Ingeniería**

La salida del ADC de los MOTES puede ser transformada a grados Kelvin, simplemente usando la siguiente aproximación, en el rango de 0-50 °C:

$$1/T(K) = a + b \times \ln(R_{thr}) + c \times [\ln(R_{thr})]^3$$

Ecuación 3. 1: Aproximación Conversión

donde:

Rthr = R1(ADC_FS-ADC)/ADC

a = 0.00130705

b = 0.000214381

c = 0.000000093

R1 = 10 kΩ

ADC_FS = 1023

ADC = medida de la salida del ADC.

3.4.4 Acelerómetro Axial (MTS310CA/MTS310CB Solamente)

El acelerómetro es una superficie tipo MEMS, micro maquinizada en dos ejes, un dispositivo de ± 2 g. se caracteriza por una capacidad de detección o sensibilidad muy baja (< 1mA) y una resolución de 10 bits. El sensor puede ser usado para detección de movimiento, de vibraciones, y/o para mediciones de carácter sísmico. La alimentación del acelerómetro es controlada a través de la señal PW4, y la señal análoga es muestreada en ADC3 y ADC4. El acelerómetro, en la locación U5 es un ADXL202JE y su completa descripción se encuentra detallada en la hoja técnica provista por http://www.analog.com. A continuación se muestra un pequeño resumen de las especificaciones técnicas del acelerómetro.

Tabla 3. 2: Resumen de Especificaciones de ADXL202JE

Canales	X (ADC3), Y (ADC4)
Rango G	±2 g (1 g = 9.81 m/s2)
Ancho de Banda	DC-50 Hz (controlado por C20, C21)
Resolución	2 mG (0.002 G) RMS
Sensitividad	167 mV/G ±17 %
Offset	2.5 V ±0.4 V

La sensitividad del ADXL202 y su *offset* tienen una muy ancha tolerancia inicial. Simplemente, usando una calibración del campo gravitacional de la tierra, se puede aumentar u optimizar grandemente la precisión del sensor ADXL202 sensor. Y mediante la

rotación en el rango de +1 G and a –1 G del sensor, el offset y la sensitividad pueden ser calculadas en el rango de hasta 1 %.

3.4.5 Magnetómetro de Dos Ejes (MTS310CA/MTS310CB Solamente)

El circuito del magnetómetro es un sensor de silicón que tiene un único puente de resistencias cubierto por una altamente sensitiva cubierta de tipo *NiFe*. La mencionada capa o cubierta causa que la resistencia del puente resistor del circuito sea alterado por variaciones, justamente en los rangos de magnetismo generados. El puente es altamente sensible y puede obtener mediciones muy precisas sobre el campo de la Tierra y otros campos magnéticos menores. Una muy útil aplicación de este sistema o dispositivo es la detección vehicular. Pruebas exitosas realizadas sobre el dispositivo, han logrado detector alteraciones en el campo magnético, generado por automóviles, dentro de un radio de hasta 15 pies.

El sensor es de la casa *Honeywell*, cuya descripción es **HMC1002**. Similar al caso anterior, una detallada descripción del equipo se encuentra en la dirección http://www.ssec.honeywell.com. Las mejoras implementadas en el equipamiento del presente proyecto, es la amplificación de las salidas de cada uno de los ejes (X, Y), las mismas que son amplificadas por U6 y U7. Dicha señal amplificada se encuentra disponible en ADC5 y ADC6. La potencia de alimentación es controlada mediante la señal PW5. Y cada uno de los amplificadores en cuestión (U6, U7) pueden ser sintonizados mediante la variación o manipulación del potenciómetro digital PT1 que es controlado a través del bus I2C.

Dentro de las especificaciones de seguridad, se debe tener muy en cuenta la extremada sensibilidad de la coraza NiFe del sensor magnético. Además, hay que tener presente su saturación, ésta ocurre cuando el sensor es expuesto a un largo campo magnético durante un tiempo demasiado grande. Desafortunadamente, el circuito del MTS310 no cuenta con una solución de recuperación automática de tipo *set/reset*. Esta limitación es la que impide el uso del sensor de magnetismo en aplicaciones realmente útiles.

3.4.6 Encendiendo y Apagando los Sensores

Cada uno de los sensores de la placa tienen un control de alimentación. Es estado predeterminado de cada uno de los sensores es estar apagados; de esta manera lo que se busca es optimizar el uso de energía por parte del circuito, es decir evitar el desperdicio de energía de la placa del sensor.

Ahora bien, para poder acceder al encendido de cada uno de los sensores, las señales de control son utilizadas para accionar como interruptores para su efecto. La tabla que se muestra a continuación, enlista cada una de las formas de seteo del control de encendido de los sensores.

Tabla 3. 3: Listado de Seteo de Señales de control

Sensor/Actuador	Control Signal
Sounder	PW2
Micrófono	PW3
Acelerómetro	PW4
Magnetómetro	PW5
Temperatura (RT2)	INT2/PW01
Foto Celda (R2)	INT1
Temperatura (RT2) (MTS300CB/MTS310CB)	PW0

Dentro de las señales descritas en la Tabla anterior, hay que tener presente que solamente una de las dos señales INT1 y INT2/PW0 puede y debe estar encendida al mismo tiempo.

Para los sensores MTS300CA y MTS310CA, la alimentación RT2 es controlada a través del seteo de la señal INT2. Mientras que para los sensores MTS300CB y MTS310CB, la misma señal de alimentación es controlada a través de PW0.

CAPITULO 4

DISEÑO DE LA RED DE SENSORES INALAMBRICOS

Una LoWPAN es una red de comunicaciones de bajo coste que permite conectividad sin hilos a elementos de funcionalidades reducidas en Redes inalámbricas multisalto, aplicaciones sin grandes requerimientos de velocidad. Estas redes cumplen el estándar IEEE 802.15.4, del cual se habla en el siguiente apartado. Ofrecen soporte para direcciones de 16 o 64 bits, su ancho de banda es reducido (250 kbps, 40 kbps y 20 kbps para las capas físicas definidas de 2.4 GHz, 915 MHz y 868 MHz respectivamente) y permite topologías mesh o en estrella. Están orientadas a dispositivos como los sensores: de bajo coste, con capacidades de procesado, memoria de almacenamiento y batería limitados.

En el ámbito de las LoWPANs existen varios problemas y requerimientos: - Tamaño limitado de paquete: el máximo tamaño de paquete en capa física es de 127 bytes, y en capa MAC de 102 bytes.

- Conectividad IP: el gran número de dispositivos plantea la necesidad de un espacio de direccionamiento grande. Además es aconsejable una auto configuración sin estado y solucionar la interconexión con otras redes IP, incluyendo Internet. IPv6 propone soluciones a estos problemas, pero hay que tener en cuenta el limitado tamaño de paquete.

- Topologías: dado que las topologías mesh implican encaminamiento multisalto, el protocolo de encaminamiento debe minimizar el overhead así como los requerimientos de memoria y procesado.

- Gestión y configuración limitada: puede darse el caso en el que dada la localización de algunos dispositivos, éstos sean difíciles de acceder. Dados el caso de hallarse obstáculos o interferencias causadas ya sea por equipos adversos o el medio en si, es indispensable que la configuración de los protocolos debería minuciosa tomando en cuenta todo este tipo de inconvenientes a presentarse.

- Descubrimiento de servicios: las LoWPANs requieren protocolos simples de descubrimiento para descubrir, controlar y mantener los servicios que proporcionan los dispositivos.

- Seguridad: debe considerarse dependiendo de las necesidades de cada aplicación. IEEE 802.15.4 proporciona seguridad a nivel de enlace mediante AES (Advanced Encryption Standard). Debido a la naturaleza de los dispositivos, deberían descartarse soluciones que impliquen un excesivo ancho de banda o nivel de cálculo.

4.1 ANÁLISIS DE LA TECNOLOGÍA DE COMUNICACIÓN

En la IETF existe el grupo LoWPAN, que en colaboración con otros grupos como la Zigbee Alliance, centra su trabajo en las redes LoWPAN y más concreto en su uso junto con soluciones basadas en el protocolo IPv6. Entre sus objetivos se encuentra la adaptación de IP a los formatos de paquete y la interoperabilidad, la gestión de direcciones y de la red,

el encaminamiento mediante topologías adaptables dinámicamente, la seguridad, la Optimización del protocolo de encaminamiento en redes de sensores IEEE 802.15.4 descubrimiento de dispositivos y otras consideraciones en cuanto a la implementación.

Hasta ahora ha proporcionado el draft "LoWPAN: Overview, assumptions, problem statement and goals", en el que se presentan las características y problemas de la transmisión IP en redes 802.15.4, y el "Transmission of IPv6 Packets over IEEE 802.15.4 WPAN Networks", en el que definen los formatos básicos de los paquetes y una subcapa IP de adaptación para la transmisión de paquetes IPv6.

Además también están trabajando en la aplicación de los protocolos de encaminamiento de redes MANET, como por ejemplo el AODV, OLSR y DYMO, sobre redes LoWPAN.

4.1.1 Tecnología radio IEEE 802.15.4

La plataforma MICAz (ver apartado 3.2) usada en el proyecto funciona en la banda global ISM1 de 2.4 GHz y soporta el estándar IEEE 802.15.4 y Zigbee, una arquitectura de protocolos basada en 802.15.4 elaborada por la Zigbee Alliance.

En los últimos años las redes inalámbricas han experimentado un gran crecimiento. El punto de mira de las estandarizaciones, tal y como demuestran los estándares de la serie 802.11, se ha dirigido hacia aplicaciones de altas tasas de datos y de largo alcance. De hecho Bluetooth ha sido el primer estándar dirigido hacia aplicaciones sin una gran necesidad de velocidad, pero la complejidad del protocolo ha hecho que resulte costoso e inapropiado para algunas aplicaciones con requerimientos de poco consumo de energía y

bajo coste. A medida que en el mercado van apareciendo más dispositivos de mayor calidad y menor coste, las WPANs han ido ganando en importancia.

Históricamente se han dedicado más esfuerzos a estandarizaciones para las aplicaciones de mayores necesidades de tasa, mientras que para el resto el principal obstáculo ha sido la falta de estándares globales, hasta la aparición del 802.15.4.

El 802.15.4 define la capa física (PHY) y la de acceso al medio (MAC) para dispositivos simples que consumen una potencia mínima y que operan en una área de 10 m o menos. Ofrece soporte para topología en estrella y peer-to-peer multisalto cuando los enlaces superan los 10 m.

- **Capa PHY**

Dentro de la capa física se definen dos opciones (las dos basadas en DSSS2): la banda global de 2.4 GHz y la de 868/915 MHz, disponibles en Europa y Norte América respectivamente. La banda de 2.4 GHz tiene 16 canales con una tasa de 250 kbps, la de 915 MHz tiene 10 con una tasa de 40 kbps y la de 868 MHz un canal de 20 kbps. La siguiente tabla muestra un resumen de las dos bandas:

Tabla 4.1: Resumen de las características de la capa PHY

PHY (MHz)	Banda de frecuencia (MHz)	Tasa de chip (Chips/s)	Modulación	Tasa de bit (kbps)	Tasa de símbolo (ksímbolos/s)
868 / 915	868 – 868.6	300	BPSK	20	20
	902 - 928	600	BPSK	40	40
2450	2400 – 2483.5	2000	O-QPSK	250	62.5

802.15.4 define dos tipos de dispositivos: FFD (Full Function Device) y RFD (Reduced Function Device). Los dispositivos FFD pueden operar en tres modos: como coordinador de PAN (Personal Area Network), como coordinador o como dispositivo, mientras que los RFD están pensados para aplicaciones muy simples.

Las características que proporciona la capa PHY son la activación y desactivación del transceptor radio, la detección de energía, el indicador de calidad del enlace (LQI, Link Quality Indicator), la selección de canal, el CCA (Clear Channel Assessment) y la transmisión y recepción de paquetes a través del medio físico.

- **Capa MAC**

Las características que proporciona la subcapa de acceso al medio son la gestión de beacons, el acceso a los canales, la gestión de time slots garantizados (GTS, Guaranteed Time Slots), el envío de tramas con reconocimiento (ACK) y la asociación y disociación.

Se caracteriza por la baja complejidad con respecto a Bluetooth; como comparación, en 802.15.4 la capa MAC proporciona 26 primitivas, mientras que Bluetooth tiene alrededor de 131 primitivas en 32 eventos.

Existen cuatro tipos diferentes de tramas MAC: tramas de beacon, de datos, de ACK y de comandos. Las redes LoWPAN pueden utilizar uno de los dos mecanismos de acceso a los canales dependiendo de la configuración de la red. En las redes beacon con supertramas, una portadora fragmentada en varios intervalos de tiempo (slots) permite múltiples accesos mediante el mecanismo CSMA-CA rasurado (Carrier Sense Multiple Access – Collision Avoidance). Los dispositivos que quieren transmitir durante su periodo

de acceso esperan a que empiece el siguiente slot y determinan si algún otro dispositivo está transmitiendo en ese mismo slot. En redes beaconless se utiliza el CSMA-CA no ranurado.

Clasificación De Protocolos Mac

Existen gran variedad de protocolos, que fueron desarrollados con anterioridad para otros tipos de redes, y que ahora han sido implementados sobre redes inalámbricas de sensores. Además, diversos grupos de trabajo han creado otros protocolos ad hoc para este tipo específico de redes, lo que nos da un amplia gama de protocolos, que se enumeran a continuación según su tipo. Protocolos de Acceso Múltiple por Detección de Portadora (CSMA. Carrier Sense Multiple Access) o ranurados:

- Sensor MAC (S-MAC)
- 802.15.4
- Timeout MAC (T-MAC)
- Berkeley-MAC (B-MAC)
- Power-Aware Multi-access protocol with signaling (PAMAS) Protocolos de Acceso multiplexado por división de tiempo (TDMA. Time Division Multiplex Access):
- Traffic-Adaptive MAC (TRAMA)
- Low-Energy Adaptive Clustering Heirarchy(LEACH)
- Power Aware Clustered TDMA (PACT)
- Bit-Map Assisted (BMA)
- Proposed Gateway MAC (G-MAC)
- SPRIME (SupSlot Period Reservation & Inter-Master Estimation)

- SMACS (Self-Organized MAC for Sensor networks)

Otros:

- DMAC (Dynamic Topology MAC)
- CMAC (Spatial Correlation-based Collaborative)
- DSMAC (Dynamic Duty Cycle for WSN)
- SmartNode (*send with same power as received)
- STEM (Sparse Topology and Energy Management)
- DPSM (Dynamic Power Saving Mechanism)
- MACA (MultiAccess Collision Avoidance)
- ZMAC (Hybrid MAC for WSN)

4.1.2 PROTOCOLOS DE ENRUTAMIENTO PARA WSN

Los nodos no tienen un conocimiento de la topología de la red, deben descubrirla. La idea básica es que cuando un nuevo nodo, al aparecer en una red, anuncia su presencia y escucha los anuncios broadcast de sus vecinos. El nodo se informa acerca de los nuevos nodos a su alcance y de la manera de enrutarse a través de ellos, a su vez, puede anunciar al resto de nodos que pueden ser accedidos desde él.

Transcurrido un tiempo, cada nodo sabrá que nodos tiene alrededor y una o más formas de alcanzarlos. Los algoritmos de enrutamiento en redes de sensores inalámbricas tienen que cumplir las siguientes normas:

- Mantener una tabla de enrutamiento razonablemente pequeña
- Elegir la mejor ruta para un destino dado (ya sea el más rápido, confiable, de mejor capacidad o la ruta de menos coste)

- Mantener la tabla regularmente para actualizar la caída de nodos, su cambio de posición o su aparición
- Requerir una pequeña cantidad de mensajes y tiempo para converger

- **Modelos De Enrutamiento**

a) Modelo de un salto

Este es el modelo más simple y representa la comunicación directa. Todos los nodos en la red transmiten a la estación base. Es un modelo caro en términos de consumo energético, así como inviable porque los nodos tienen un rango de transmisión limitado. Sus transmisiones no pueden siempre alcanzar la estación base, tienen una distancia máxima de radio, por ello la comunicación directa no es una buena solución para las redes inalámbricas.

b) Modelo Multi-hop

En este modelo, un nodo transmite a la estación base reenviando sus datos a uno de sus vecinos, el cual está más próximo a la estación base, a la vez que este enviará a otro nodo más próximo hasta que llegue a la mota base. Entonces la información viaja de la fuente al destino salto a salto desde un nodo a otro hasta que llega al destino. En vista de las limitaciones de los sensores, es una aproximación viable.

Un gran número de protocolos utilizan este modelo, entre ellos todos los MultiHop de Tmote Sky y Telos: MultiHop LQI, MintRoute, Router, etc. Éste modelo será el utilizado en nuestra red, ya que dado el número de nodos, el tipo de sensores y el software del que disponemos, hacen del modelo MultiHop una solución óptima.

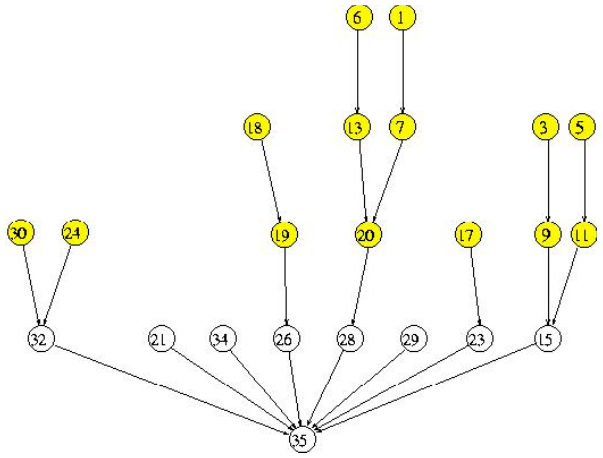


Figura 4.1. Ejemplo de una red de sensorial donde la base es el nodo 35, el resto de sensores envían sus datos ha ella con una profundidad máxima de 4 saltos (hops) para los nodos 1 y 6.

b) Modelo esquemático basado en clústeres

Algunos otros protocolos usan técnicas de optimización para mejorar la eficacia del modelo anterior. Una de ellas es la agregación de datos usada en todos los protocolos de enrutamiento basados en clústeres. Una aproximación esquemática rompe la red en capas de clústeres. Los nodos se agruparán en clústeres con una cabeza, la responsable de enlutar desde ese clúster a las cabezas de otros clústeres o la estación base. Los datos viajan desde un clúster de capa inferior a uno de capa superior. Aunque, salta de uno a otro, lo está haciendo de una capa a otra, por lo que cubre mayores distancias. Esto hace que, además, los datos se transfieran más rápido a la estación base.

Teóricamente, la latencia en este modelo es mucho menor que en la de MultiHop. El crear clústeres provee una capacidad inherente de optimización en las cabezas de clúster. Por tanto, este modelo será mejor que los anteriores para redes con gran cantidad de nodos en un espacio amplio (del orden de miles de sensores y cientos de metros de distancia).

4.1.3 MODELO POR CAPAS DE LAS WSN

Como con otras redes, el modelo de capas de WSN esta basado en el modelo de referencia OSI como se muestra a continuación:

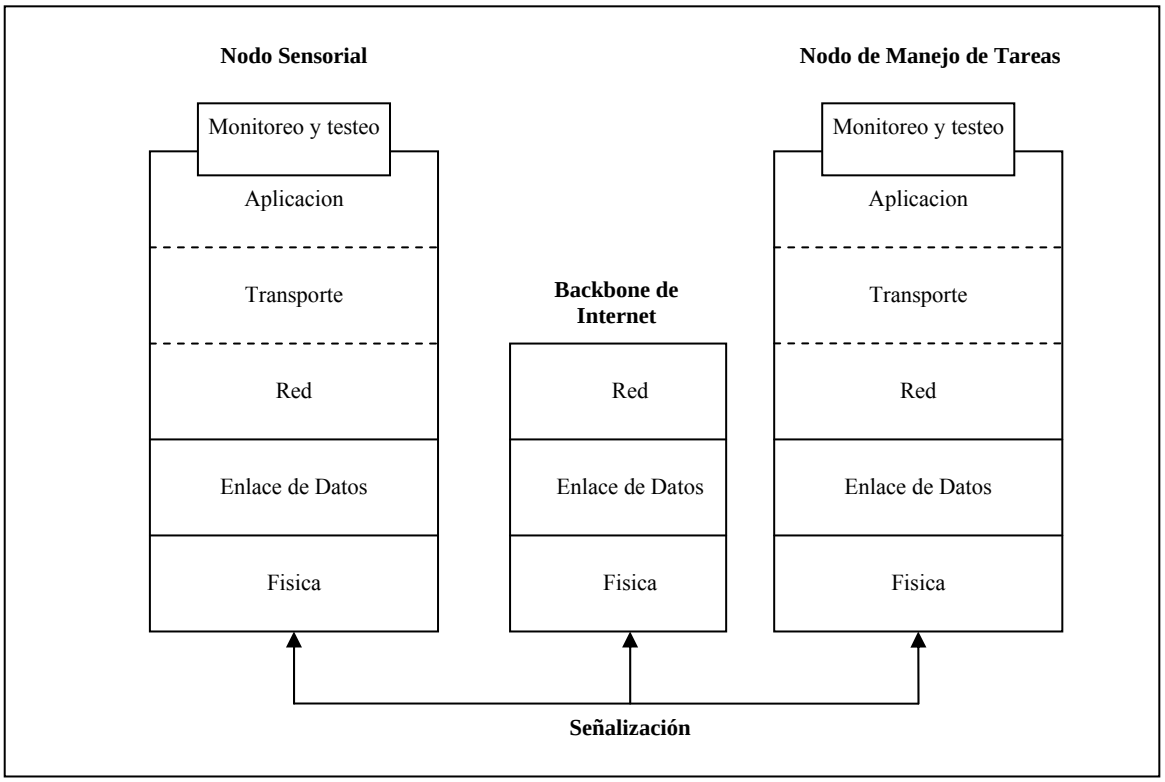


Figura 4.2. Modelo de Capas de redes de sensores genéricas

- **Capa Física**

Es la responsable de transmitir bits individuales para modulación y técnicas de propagación en el espectro, sobre bandas de frecuencias destinadas. En WSN se emplea usualmente esquemas de modulación simples tales como BPSK (Binary Phase Shifting Keying) o QPSK los cuales son suficientes para proveer bajas tasas de transmisión. Además, son usados también en esquemas DSSS (Direct Sequence Spread Spectrum). Estos son comúnmente usados en bandas de frecuencia medicas e industriales (ISM) a

900Mhz o 2400Mhz (zigbee), o longitudes de onda de infrarrojo dedicadas a comunicaciones en una sola línea de vista.

- **Capa de enlace de datos**

Es la responsable de la transmisión de paquetes de datos. En conexiones inalámbricas, la subcapa MAC provee el protocolo para acceso hacia los canales de comunicación comunes. Debido al consumo de energía y a los requerimientos de organización, los protocolos MAC convencionales son evitados por esto muchos protocolos MAC para redes sensoriales son propuestos. Además, varios modos de seguridad pueden ser incorporados dentro de los protocolos de capa MAC.

Por ejemplo, 802.15.4 (zigbee MAC) provee servicios para encriptación de datos, integridad de tramas, y control de acceso a través de Encriptación Standard Avanzada (AES) para modos de operación seguros. Técnicas de ruteo dedicadas tales como SPIN, y LEACH son propuestas para direccional estas entradas.

- **Capa de aplicación**

Provee varios servicios destinados hacia aplicaciones de WSN. Esto incluye protocolos tales como SMP (Sensor Management Protocol), TADAP (Protocolo de Asignación de tareas y advertencia de datos).

SMP, permite la interacción con los nodos incluyendo la búsqueda de localización, apagado, configuración de red, y sincronización en el tiempo para aplicaciones de manejo de redes.

TAADAP, provee el software de usuario con una interfaz que permite a los usuarios expresar su interés sobre las funciones de los nodos de sensores los nodos de sensores pueden advertir también sobre los datos disponibles de los usuarios.

- **Capa de Red**

Reparte técnicas de ruteo eficiente, las cuales son esenciales para preservar energía. El ambiente sin control operativo, con fallas comunes aleatorias de los nodos sensoriales, complican aun mas al router.

4.2 PARÁMETROS DE DISEÑO DE LA RED.

Cada nodo de la red consta de un dispositivo con microcontrolador, sensores y transmisor/receptor, y forma una red con muchos otros nodos, también llamados motas o sensores. Por otra parte, un sensor es capaz de procesar una limitada cantidad de datos. Pero cuando coordinamos la información entre un importante número de nodos, éstos tienen la habilidad de medir un medio físico dado con gran detalle.

Con todo esto, una red de sensores puede ser descrita como un grupo de motas que se coordinan para llevar a cabo una aplicación especifica. 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.

Las redes de sensores están formadas por un pequeño número de nodos que estaban conectados por cable a una estación central de procesado de datos. Hoy en día, sin embargo, nos centramos más en redes de sensores distribuidas e inalámbricas. Pero, por qué distribuidas e inalámbricas: cuando la localización de un fenómeno físico es

desconocida, este modelo permite que los sensores estén mucho más cerca del evento de lo que estaría un único sensor.

Además, en muchos casos, se requieren muchos sensores para evitar obstáculos físicos que obstruyan o corten la línea de comunicación. El medio que va a ser monitorizado no tiene una infraestructura, ni para el suministro energético, ni para la comunicación. Por ello, es necesario que los nodos funcionen con pequeñas fuentes de energía y que se comuniquen por medio de canales inalámbricos.

Otro requisito para las redes de sensores será la capacidad de procesamiento distribuido. Esto es necesario porque, siendo la comunicación el principal consumidor de energía, un sistema distribuido significará que algunos sensores necesitarán comunicarse a través de largas distancias, lo que se traducirá en mayor consumo. Por ello, es una buena idea el procesar localmente la mayor cantidad de energía, para minimizar el número de bits transmitidos.

En el laboratorio avanzado de telecomunicaciones SAT, del Departamento Eléctrica y Electrónica de la ESPE, vamos a desarrollar un sistema de programación en red para redes de sensores inalámbricas que ofrecerá las mejores prestaciones posibles ya que se trata de un área industrial reducida al tratarse de un cuarto de telecomunicaciones se requieren tomar en cuenta parámetros en los cuales los equipos funcionaran sin problema a dentro de la temperatura necesaria para que estos no se deterioren ni sufran danos drásticos.

Las WSNs se esperan que sean tomadas para aspectos industriales principalmente, ya que su confiabilidad, disponibilidad y servicio (RAS), se convierten en un factor critico. En los sistemas de redes tradicionales, proveer suficiente RAS puede ser absorbido por el costo de la red. De todas maneras los diseñadores de la red tienen que enfrentar dos objetivos fundamentalmente conflictivos: minimizar el total del costo de la red y proveer redundancia como protección hacia interrupciones de servicio mayores.

La redundancia física es una técnica usada comúnmente usada para asegurar la confiabilidad del sistema. Tomando lugar múltiples nodos independientes, la red se protege de fallas en puntos únicos de hardware y software. Para accesibilidad y servicio, es necesario realizar testeo y diagnostico para señalar y reparar los componentes dañados que pueden ser físicamente inalcanzables.

Existen limitaciones severas con relación al costo y a la energía transmitida dentro de las WSNs tienen un impacto negativo a la accesibilidad de los nodos y a la integridad de los datos transmitidos. Tradicionalmente los protocolos de comunicación de la capa de transporte bien definidos están siendo usados para asegurar la integridad de transmisión de datos fin a fin. La mayor parte de los WSNs son usados principalmente para recolección ambiental de datos para datos relativamente no críticos, como temperatura del ambiente como es el caso del proyecto.

Perder una pequeña porción de la información o corromper resultados en medición no representan un problema sobre el largo período de medición que se va a obtener, específicamente hablando de datos como la temperatura. Sin embargo, el testeo remoto y la reparación son extremadamente difíciles cuando la integridad de la transmisión de datos no

se garantiza. Como resultado la confiabilidad, la accesibilidad y el servicio se ven severamente afectados por estas restricciones.

4.3 DISEÑO Y CONFIGURACIÓN DE LA RED

Al diseñar una red de sensores inalámbricos es importante tomar en cuenta el sistema operativo y el software que se va a emplear para el diseño y gestión de la misma. TinyOs, como se mencionó anteriormente es el software con el cual se programara los sensores para realizar la red, su ultima versión incluye librerías que proveen un router multi-hop y ad-hoc para aplicaciones de sensores en red.

Una red de sensores inalámbricos esta formada por tres componentes: los nodos de sensores, el nodo de gestión de tareas o base y el backbone de interconexión.

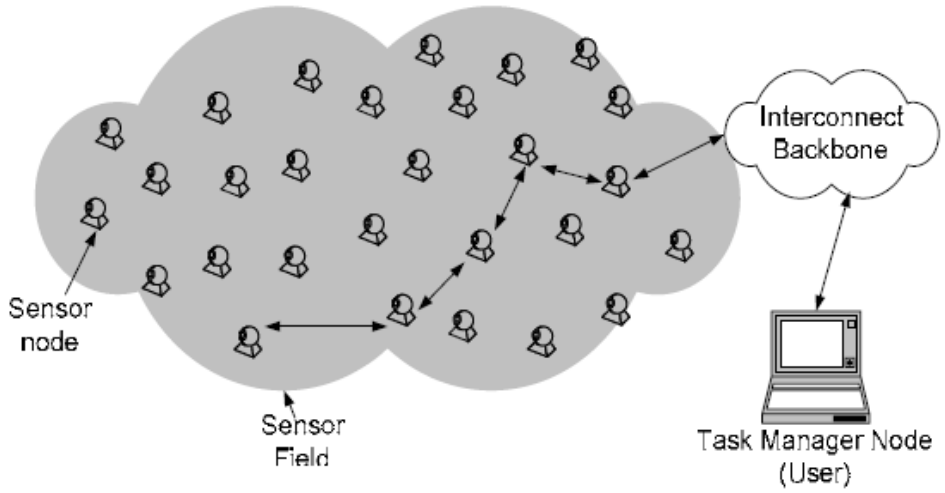


Figura 4.3: Red De Sensores Inalámbricos Con Sus Componentes

Cada nodo sensor puede contener varios sensores y actuadotes que son usados para recolectar la información y controlar los procesos físicos. Los datos recolectados son

transferidos hacia el usuario a través de la red. Además de recolectar datos y controlar los actuadotes, un nodo debe realizar algunos cálculos con respecto a la medición de datos. Una comunicación directa entre nodos individuales también podría ser requerida.

El nodo agente de tareas (base o usuario) dependiendo de cómo se lo quiera denominar realiza tareas con respecto al almacenamiento de datos, análisis y despliegue de los mismos, en adición realiza el control de la interfase que interconecta hacia el backbone. Debido a limitaciones menos estrictas, se puede realizar tareas significativamente mas complejas que en los nodos WSN.

En general, las redes sensoriales inalámbricas deben encontrarse con requerimientos de medida en tiempo real y proveer un sistema robusto. Los requerimientos generales para las redes inalámbricas deben incluir lo siguiente:

a) *Consumo de baja potencia:* los nodos están potenciados usualmente por baterías. El reemplazo manual de las baterías no es posible usualmente, lo cual hace que los nodos dependan del tiempo de vida de la batería. Como resultado, la minimización del consumo de energía de esta manera se convierte en crítico el alcanzar un sistema robusto.

b) *Escalabilidad*: las WSNs con una gran cantidad de nodos pueden convertirse en comunes. De esta manera, se convierte en estacionaria en algunos casos, los sensores móviles podrían ser usados en aplicaciones ambientales o militares. La escalabilidad del sistema desde ahí se convierte en una importancia mayor.

c) *Habilidad de organización propia:* las WSNs pueden ser largas en tamaño y trabajo en el ambiente de trabajo que causa el incremento de fallas en los nodos individuales. Mecanismos son necesarios para juntar la red de manera aleatoria, de esta forma se reorganiza la red de acuerdo a fallas desde la habilidad esencial de organización a si mismo.

d) *Habilidad de duda:* debido al tamaño de la red, la cantidad de información agregada debería ser muy extensa como para poder ser transmitida a través de toda la red. Por esto, la recolección de información de una región especifica o de nodos específicos puede ser aplicada. Ciertos nodos WSN necesitan ser dedicados para la recolección de información a partir de ciertas regiones, creando un sumario y direccionando la información. La función de duda es usada para identificar la recolección de nodos y sus correspondientes regiones.

Como en otros sistemas de redes, las redes de sensores inalámbricas tienen sus propios mecanismos como se dijo anteriormente SMP, para asegurar, una operación confiable sobre toda la red inalámbrica. Así mismo, un bajo costo en hardware, facilita la redundancia en el significado de que la red se ira deteriorando de acuerdo a la cantidad de sensores que sean colocados como nodos dentro de la red.

Una manera de solución a una red optima seria el testeo y diagnostico de la infraestructura para nodos de sensores individuales. El objetivo es el identificar los nodos que fallan y repararlos remotamente por medio de el hardware redundante embebido, o posiblemente descargando y cargando un software programable dentro del hardware.

Una vez dicho esto se tomara en consideración los siguientes parámetros para realizar la configuración para el monitoreo y gestión de la red sensorial:

4.3.1 Topología de la red

Una propiedad importante de una red de sensores es el diámetro, esto significa, el máximo numero de saltos que sen tendrá entre los nodos de la red. En su forma más simple, una red de sensores forma una red de saltos simples, con cada nodo de sensor siendo capaz de comunicarse directamente con cada nodo.

Así de esta forma al tratarse de una red de sensores para un laboratorio industrial destinado a las comunicaciones avanzadas, se realizara una red tipo malla simple con un máximo de 4 a 5 nodos y una estación base, están podrán comunicarse entre si, pero realmente lo que nos interesa es la información que estas envían a la estación base con relación a la variable temperatura de cada nodo sensorial.

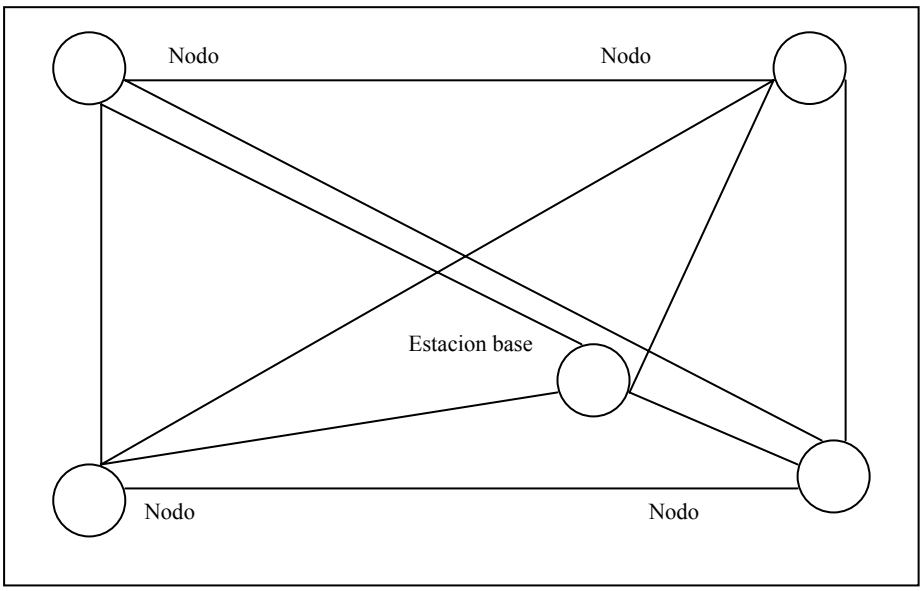


Figura 4.4: Descripción de la topología de la red sensorial dentro de cuarto de telecomunicaciones avanzadas.

4.3.2 Cobertura

El rango efectivo de comunicación de los nodos colocados dentro de la red entre si, define el área de cobertura del nodo sensorial. Así la medida de la cobertura es el grado de cobertura que tiene cada de acuerdo al área de interés. La cobertura varía a través de la red de acuerdo a la precisión requerida y de acuerdo a la ubicación física. Al tratarse de la implementación dentro de un cuarto de telecomunicaciones avanzadas, entonces el área de cobertura únicamente será dentro de tal cuarto ya que la implementación y el objetivo de la red es monitorear únicamente dicho cuarto, el rango de cobertura vendrá dado únicamente por el área del cuarto siendo un cuarto donde la máxima distancia entre nodos es de 7 m, del nodo mas lejano a la estación base.

4.3.3 Conectividad

Los rangos de conectividad y las locaciones físicas de cada sensor inalámbrico individual definen la conectividad de la red. Si siempre existe una conexión de red (posiblemente sobre múltiples saltos) entre cualquier nodo, la red se dice estar conectada. Para propósito de diseño de la red es indispensable que la red este conectada con el nodo o estación base la cual brindara la información necesaria para el monitoreo y gestión de la red.

Así la conectividad se dice que esta influenciada por los protocolos de comunicación y los métodos de recolección de datos, en este caso se sabe que la variable a ser colectada será la temperatura, además de esto es necesario recolectar de cada nodo la información sobre el numero de nodo para tener certeza de la ubicación sobre la que provienen dichos

datos, esto se describirá dentro del programa ya que la red será diseñada por medio del sistema operativo TinyOs a través de ventanas Cygwin.

4.3.4 Tamaño De La Red

Esto indica el número de nodos participantes dentro de la red esto es dado principalmente por los requerimientos relacionados con la cobertura y conectividad de la misma y por el tamaño del área de interés. Por lo tanto al ser un cuarto de comunicaciones avanzadas rectangular donde la máxima distancia entre un nodo y la estación base será de 7m, entonces se considera instalar dentro de la infraestructura una cantidad de 4 nodos para monitorear la variable temperatura, así se programara 4 nodos y una estación base dentro de la red.

4.3.5 Tiempo De Vida

Dependiendo de la aplicación los requerimientos de tiempo de vida de un sensor dentro de una red deben variar de acuerdo a horas de trabajo continuo a través de los anos, al tratarse de sensores inalámbricos y una vez analizado el hardware se conoce que su duración es alta debido a su bajo consumo de energía, por lo tanto al tratarse del monitoreo único de la variable temperatura, el sensor esta destinado a un trabajo continuo de 5 anos a penas con 2 baterías de 1.5V

4.3.6 Programación de la red

Una vez vistos parámetros importantes de diseño de la red se procederá a programar tanto cada uno de los nodos así como la estación base para sensar la variable temperatura dentro del sistema operativo TinyOs, el programa esta anexado al final dentro de la sección anexos ANEXO B y detallada cada una de las variables.

Para la programación de la red es necesario tomar en cuenta ciertos parámetros importantes dentro de lo que tiene que ver con las redes sensoriales inalámbricas esto es con respecto a la forma de programación de la estación base y de cada uno de los nodos sensoriales, para esto se realizaron 2 programas, uno para la estación base que será la encargada de receptar toda la información que cada uno de los nodos ubicados en el cuarto de telecomunicaciones avanzadas enviara este programa se encuentra en la sección anexos esta dentro de la carpeta XMESHBASE de las aplicaciones de Xbow, de esta manera como se vio anteriormente, en la sección de software, para programar, compilar e instalar dicho ejecutable en el sensor es necesario tipiar el siguiente texto dentro de una ventana Cygwin:

```
Cd opt/moteworks/apps/xmesh/xmeshbase
make micaz
make micaz install,0 mib510,com1
```

Nótese que al tratarse de la estación base el instante de programar el sensor por medio del mib510 la cual es la plataforma de programación serial, se da como id del nodo, el numero 0, el cual esta explicado dentro del programa para que la estación base y receptora de la información para la interfaz y control final del ambiente pueda ser llevado a cabo normalmente y sin problemas, el código fuente para este archivo esta ubicado en anexos como XMeshBaseM.nc y XMeshBase.nc en donde el primer archivo contiene las instrucciones de recepción en RF proveniente de los nodos y el segundo contiene las declaraciones de las interfaces o funciones que serán llamadas para realizar las acciones necesarias como estación base.

El momento de compilar los archivos de la red se llama a un archivo que es configurable el cual contiene la información básica que será escrita dentro de los sensores

es el archivo MakeXbowLocal, el momento de compilar los programas para los sensores se llama este archivo directamente incrustando características necesarias dentro de los sensores este esta ubicado en la direccion /opt/moteworks/contrib/xbow/apps/ y tiene la siguiente descripción detallada:

```
############################################################################
#
# MakeXbowlocal - Local Defines related to apps in contrib/xbow directory
#
############################################################################
############################################################################
# Settings para la programación sensorial
# si se conecta el mib510 las características de Puerto y  com están por defecto
# para usa el resto de plataformas quitar los comentarios y comentar las opciones
# por defecto
# Para MIB600 usar "eprb" y proveer el hostname/IP adress
############################################################################
DEFAULT_PROGRAM=mib510  // detalla el tipo de programador (serial por defecto)
#DEFAULT_PROGRAM=mib520
#DEFAULT_PROGRAM=eprb
MIB510=COM1         // descripción de Puerto de salida por defecto
#MIB520=COM3
#EPRB=10.1.1.238
############################################################################
# se setea el grupo para los sensores
# grupo id por defecto
############################################################################
DEFAULT_LOCAL_GROUP=125  // grupo id local 125 defecto
############################################################################
# características para canales de radio (freq)
# quitar comentario únicamente en la línea escogida para la frecuencia de radio
#deseada. Seleccionar la banda deseada para la frecuencia del sensor (916,433...)
# (p.e. el canal de 433Mhz no va funcionar para un mote configurado para una
# frecuencia de 916Mhz)
############################################################################
#
# 916 MHz Band
#
# CHANNEL_00 - 903 MHz CHANNEL_02 - 904 MHz CHANNEL_04 - 905 MHz
# CHANNEL_06 - 906 MHz CHANNEL_08 - 907 MHz CHANNEL_10 - 908 MHz
# CHANNEL_12 - 909 MHz CHANNEL_14 - 910 MHz CHANNEL_16 - 911 MHz
# CHANNEL_18 - 912 MHz CHANNEL_20 - 913 MHz CHANNEL_22 - 914 MHz
# CHANNEL_24 - 915 MHz CHANNEL_26 - 916 MHz CHANNEL_28 - 917 MHz
# CHANNEL_30 - 918 MHz CHANNEL_32 - 919 MHz CHANNEL_34 - 920 MHz
# CHANNEL_36 - 921 MHz CHANNEL_38 - 922 MHz CHANNEL_40 - 923 MHz
# CHANNEL_42 - 924 MHz CHANNEL_44 - 925 MHz CHANNEL_46 - 926 MHz
# CHANNEL_48 - 927 MHz
#CFLAGS = -DCC1K_DEFAULT_FREQ=CC1K_914_077_MHZ
#CFLAGS = -DCC1K_DEFAULT_FREQ=CC1K_915_998_MHZ
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_916BAND_CHANNEL_00
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_916BAND_CHANNEL_02
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_916BAND_CHANNEL_04
. . .
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_916BAND_CHANNEL_48
#---------------------------------------------------------------------------
# 433 MHz Band
#
# CHANNEL_00 - 433.113 MHz CHANNEL_02 - 433.616 MHz
# CHANNEL_04 - 434.108 MHz CHANNEL_06 - 434.618 MHz
#
```

```
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_433BAND_CHANNEL_00
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_433BAND_CHANNEL_02
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_433BAND_CHANNEL_04
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_433BAND_CHANNEL_06
##-------------------------------------------------------------------------
# 315 MHz Band
#
# CHANNEL_00 - 315 MHz
#
##-------------------------------------------------------------------------
#CFLAGS = -DCC1K_DEFAULT_FREQ=RADIO_315BAND_CHANNEL_00
###########################################################################
###########################################################################
# potencia de radio
# - potencia de transmisión de radio por su valor en (RTP) entre 0x00 y 0xFF
# - RTP = 0 para potencies bajas; =0xFF para máxima transmisión de potencia
#-------------------------------------------------------------------------
# para Micaz y Micadot
# banda de freq: potencia de salida (dBm) RTP
# 916 MHz -20 0x02
# -10 0x09
# 0 (1mw) 0x80
# 5 0xFF
# 433 MHz -20 0x01
# -10 0x05
# 0 (1mw) 0x0F
# 10 0xFF
###########################################################################
#característica de bandera para máxima transmisión de potencia
CFLAGS += -DRADIO_XMIT_POWER=0xFF
#############################################
#
# Niveles de potencia para transmisión en RF para Micaz
#TXPOWER_MAX TXPOWER_0DBM
#TXPOWER_0DBM 0x1f //0dBm
#TXPOWER_M3DBM 0x23 //-3dBm
#TXPOWER_M5DBM 0x19 //-5dBm
#TXPOWER_M10DBM 0x0B //-10dBm
#TXPOWER_M15DBM 0x07 //-15dBm
#TXPOWER_M25DBM 0x03 //-25dBm
#TXPOWER_MIN TXPOWER_M25DBM
#########################################
CFLAGS +=-DCC2420_TXPOWER=TXPOWER_MAX
//nivel de transmisión a máxima potencia defecto
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_M0DBM
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_M3DBM
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_M5DBM
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_M10DBM
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_M15DBM
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_M25DBM
#CFLAGS +=-DCC2420_TXPOWER=TXPOWER_MIN
#########################################################
#
# Canal de selección para Zigbee
# CHANNEL_11 - 2405 MHz CHANNEL_12 - 2410 MHz CHANNEL_13 - 2415 MHz
# CHANNEL_14 - 2420 MHz CHANNEL_15 - 2425 MHz CHANNEL_16 - 2430 MHz
# CHANNEL_17 - 2435 MHz CHANNEL_18 - 2440 MHz CHANNEL_19 - 2445 MHz
# CHANNEL_20 - 2450 MHz CHANNEL_21 - 2455 MHz CHANNEL_22 - 2460 MHz
# CHANNEL_23 - 2465 MHz CHANNEL_24 - 2470 MHz CHANNEL_25 - 2475 MHz
# CHANNEL_26 - 2480 MHz
#
# 15, 20, 25 & 26 al parecer no se sobrepone ni interferencia
# Con respecto a 802.11
#########################################################
#CFLAGS +=-DCC2420_DEF_CHANNEL=11
#CFLAGS +=-DCC2420_DEF_CHANNEL=12
. . .
#CFLAGS +=-DCC2420_DEF_CHANNEL=25
```

```
#CFLAGS +=-DCC2420_DEF_CHANNEL=26
############################################################################
```

Una vez visto el archivo MakeXbowlocal, y seteadas las características necesarias para los sensores, es importante ahora realizar la red, como sabemos para ambientes industriales es necesario tener una buena comunicación entre nodos y sin perdida de información ya que se trabajan con variables indispensables para el mantenimiento correcto de los equipos, así pues se realizara la programación de una red tipo malla así todos los nodos se comunicarán entre si y buscaran la mejor vía para transmitir las variables de importancia hacia la estación base, así pues, se tomara en cuenta la característica de Multi-saltos para que la información no se pierda a través de la red.

Además se toma en cuenta que como es un laboratorio de telecomunicaciones avanzadas pequeño entonces como método recursivo y redundancia de información se tendrán máximo 5 nodos dentro de la red y una estación base, todos estos nodos enviaran como parámetro principal la temperatura.

Hay que notar que como se tratan de sensores inalámbricos que manejan varias variables al mismo tiempo hay que darles atención también a ellas así se trate de una red para el monitoreo únicamente de la variable temperatura, así, el resto de variables serán llamadas pero enceradas dentro del programa para mantener un análisis de información limpia a lo que llegue esta a la estación base para así proseguir con el control y depuración de dicha información. Así mismo, para la creación del programa se toman en cuenta todas las variables de red dentro de las funciones así como características de transmisión y recepción en RF dentro de cada uno de los nodos a ser programados.

De esta manera el programa que se encuentra descrito dentro de la sección anexos como XMTS310M.nc y XMTS310.nc que se lo ubica dentro de la carpeta ***opt/moteworks/apps/xmesh/xmts310.*** Cabe notar que el programa original ha sido cambiado con propósitos de diseño de red. De esta manera se abre una ventana Cygwin donde se ingresara la siguiente información una vez que se ha realizado y configurado el programa para la red sensorial.

```
Cd opt/moteworks/apps/xmesh/xmts310
make micaz
make micaz install,x mib510,com1
```

Donde x representara el numero de nodo, como se dijo anteriormente, para el diseño de la red dentro del cuarto de telecomunicaciones avanzadas se requiere de una cantidad de 5 nodos máximos para tener un buen control del ambiente, por lo tanto x vendrá a ser el numero de nodo (1-5) axial estos al ser encendidos se comunicaran con la estación base y enviaran la información por el puerto serial de la siguiente manera:

Tabla 4.2: Línea de Datos recibidos por la estación

Nombre del paquete de bytes	**Ejemplo**	**Notas**
Cabecera TOS=7e00007dld		
Dirección UART	7e00	
Tipo	00	
Grupo ID	7d	defecto
ID de la plataforma	82	MIB510
Id del paquete	01	
Id del nodo	01	
Reservado		
Datos	xxxxxx7e	Depende del sensor y de la programación de las variables

CAPITULO 5

DESARROLLO DE LA RED, INTERFAZ Y AMBIENTE

5.1 AMBIENTES INDUSTRIALES

Para el proyecto existe el interés en monitorear la temperatura en el ambiente del cuarto de comunicaciones, en orden de tener un control automatizado de la temperatura mediante el uso de ventiladores ambientales. En un ambiente de telecomunicaciones es necesario tomar en cuenta que la densidad de nodos máximo llegara a cinco tomando en cuenta las características de la habitación y estarán separados entre si máximo en una distancia de 5 metros por lo tanto se colocara un sensor de temperatura en cada esquina del cuarto de comunicaciones y uno en el medio del aula así se espera obtener una temperatura promedio. Esto se toma en cuenta ya que la distancia es un factor importante ya que el coste de la comunicación aumenta con la distancia.

Las WSN han sido creadas para monitoreo de varios parámetros como temperara, luz, humedad, magnetismo, los cuales son parámetros importantes dentro de un ambiente de comunicaciones, de esta manera se debe tener mediante el monitoreo de temperatura, un promedio de temperatura deseada para el ambiente reducido.

5.2 ADQUISICIÓN DE DATOS

En esta sección se presentará la forma en que la información vital para el funcionamiento del sistema se adquiere directamente desde cada uno de los equipos, es

decir, la manera en la que la información es recopilada, el dónde, el cómo y el por qué de dichas señales.

Existiendo tres maneras distintas de adquisición de datos desde los sensores hacia la PC, o centro de monitoreo, para el presente proyecto se ha escogido la vía Serial, es decir, usando como Estación Base el módulo **MIB510**, como sensores, a los tableros de sensor **MICAz** (MTS310CB). Por lo tanto, una vez determinado el equipamiento a ser utilizado, lo siguiente es distinguir la, o, como es el caso, las interfaces de monitoreo para recepción y visualización de la información recibida; siendo en el proyecto las plataformas *MoteView* y la interfaz propia desarrollada en *Visual Basic*.

Vía serial, la información desde cada uno de los nodos, como es explicado claramente en cada una de las hojas técnicas de los sensores, viene codificada en tramas de hasta 36 bytes, dependiendo de la cantidad de información enviada a través de los mismos. Así mismo la velocidad de transmisión es la que determina el tiempo de adquisición de datos así como el refresco de la información obtenida.

5.3 DISEÑO DE LA INTERFAZ DE MONITOREO

Para el desarrollo de la interfaz de monitoreo y control del sistema, se ha escogido de la variedad de soluciones en software disponibles en el mercado, la opción de ***Visual Basic***, esto a causa de su acceso de manera sencilla y versátil al puerto de comunicaciones de la PC, el RS-232, es decir el puerto Serial de comunicaciones, además, claro está de la forma amigable de presentar las características de programación para el desarrollo de cualesquiera que sea el uso o tecnología que permita la recepción y/o transmisión de datos.

De tal forma, que, en la elaboración de cada una de las características constitutivas de la interfaz, se debieron tener en cuanta cuestiones de seguridad, almacenamiento de información, opciones de visualización, acceso dinámico al detalle del estado de cada uno de los equipos y dispositivos del sistema; entre otras características que a lo largo de la explicación de las líneas de código del programa se irán presentando en conjunto con la respectiva explicación del mismo.

En el desarrollo del código del programa mencionado, se debió tener en cuenta la necesidad del análisis detallado de la trama de transmisión de cada uno de los nodos a la estación base, es decir, dependiendo del tipo u origen de la trama, que en el presente caso es de tipo serial, se han de especificar cada una de las partes o elementos de las que esta constituida la trama, de los estudios realizados sobre la trama obtenida, se pudo determinar que consta de hasta un máximo de 32 bytes, conformados de tal forma que se encuentra en su estructura una parte correspondiente a la cabecera de la cadena.

Un identificador de nodo, la trama intermedia conforma la parte de la información receptada por cada nodo, de forma codificada, y finalmente un carácter indicador de fin de trama. Dichos análisis realizados sobre la trama se realizaron, para el proyecto, a través de la recepción de la misma por el Puerto Serie de la PC, y por medio del Hyper-Terminal de la misma, logrando obtener una trama de la siguiente forma:

Para lograr la simplificación de la cadena de caracteres, y por la necesidad de obtener solamente un tipo de variable en el proyecto, la temperatura, es que por medio de la programación de cada uno de los sensores nodos de la red, se limitó a los mismos a la recepción de dicha variable únicamente, con lo que se permitió la reducción de la cadena a

17 caracteres, con lo cual, la determinación de la forma en que se encuentran distribuidos los datos e información recibidos se redujo tremendamente, constando en ese caso simplemente de los caracteres correspondientes a la cabecera de identificación, al número de nodo, al dato de temperatura codificado; y al carácter de fin de cadena, así:

La forma en que los datos se encuentran distribuidos en la trama serial, ya reducida fue determinada de manera experimental, a través de la identificación de cada uno de los caracteres de la cadena total, y su relación a lo programado en cada uno de los sensores es que se puedo especificar la forma exacta de codificación de cada uno de los datos, de tal forma que se logro determinar que:

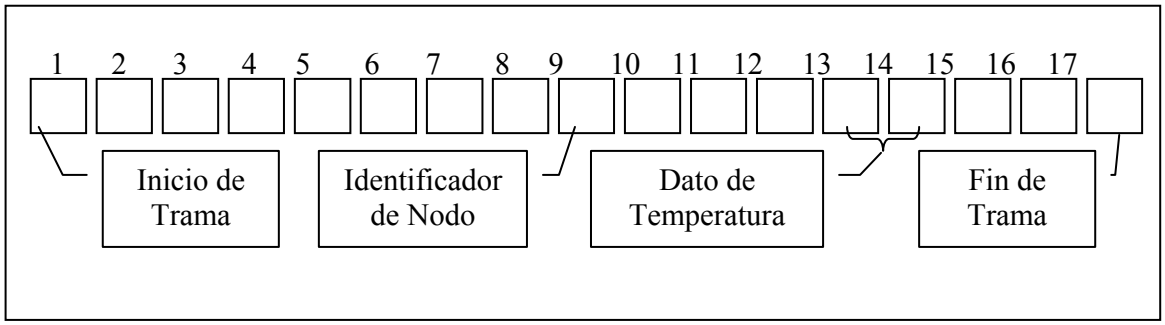


Figura 5.1: Trama Del Paquete Serial Receptada Por La Base WSN

Como se obtiene la información directamente desde el puerto serial, en forma de caracteres especiales del Código ASCII, entonces es necesaria la transformación de los mismos a hexadecimal, y finalmente a decimales para poder obtener los datos correctos de la mencionada trama.

En el caso del byte identificador de la red, se vuelve sencilla su interpretación debido a que consta de un solo dígito, por lo que a través de las transformaciones podemos simplemente obtener el valor de su nodo; sin embargo, en el caso de la temperatura, la

transformación no es suficiente, esto debido a que para el caso, este dato consta de dos bytes de ASCII, una vez que se transforma se obtienen con ellos, 4 datos hexadecimales, y su interpretación fue sometida a una serie de pruebas experimentales, a través de las cuales se logró determinar que la forma en que viene dada la temperatura es la siguiente:

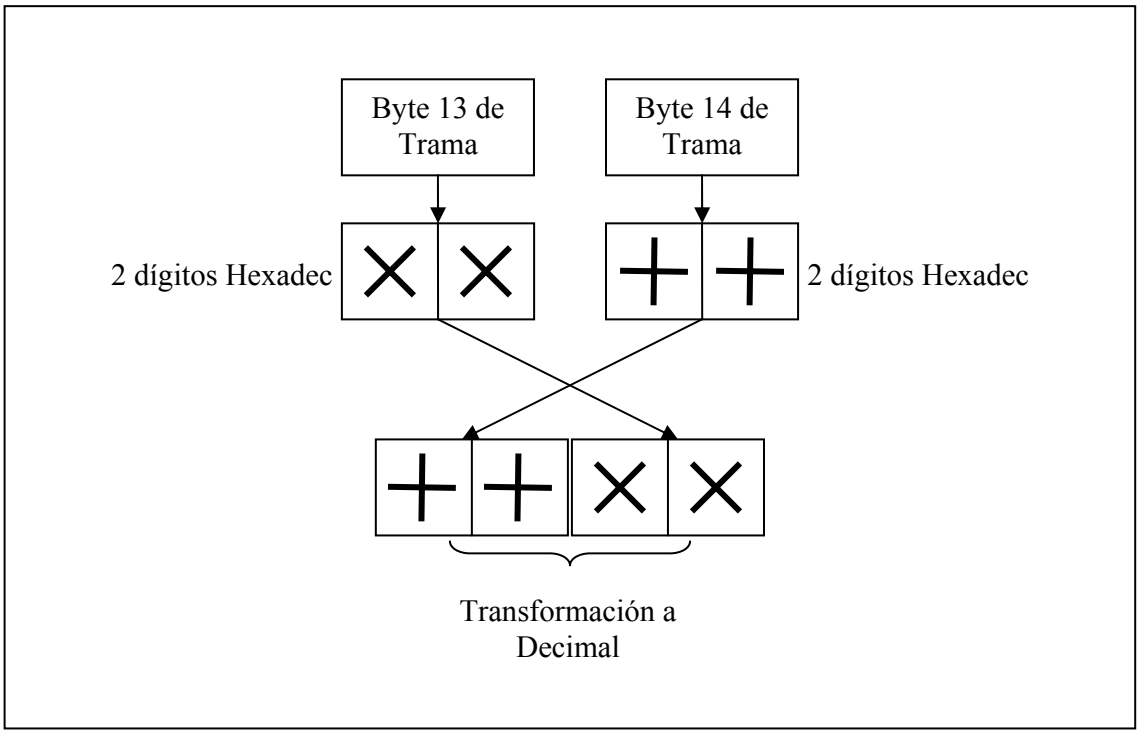


Figura 5.2: Descripción de la validación realizada para la obtención del parámetro temperatura

Finalmente, en el proceso de obtención de la temperatura, una vez que se tiene la información en forma decimal, procedemos a la transformación de las unidades crudas de ingeniería a unidades reales, que para el caso vienen dadas en grados Centígrados, esto a través de la regla de tres obtenida por medio de una serie de pruebas y tabulaciones que concluyeron en la siguiente expresión:

$$Temperatura(^{o}C) = {(19.78 * 448)}/{UnidadCruda}$$

Ecuación 5. 1: Transformación de Unidades Crudas

Determinada la forma en que la información era receptada en los nodos, se procedió a la utilización de los parámetros necesarios para ser visualizados en pantalla, es decir, la temperatura y el número de nodo; de tal forma que con dicha información, se logra la implementación de cada una de las pantallas de visualización de la interfaz implicada, sobre lo cual se detalla a continuación.

Una vez que se maneja ya la parte correspondiente a la adquisición de datos, entonces es posible acceder a la codificación de un programa en el cual se puedan apreciar todas las bondades del equipamiento adquirido, es decir, a través de la identificación de cada uno de los nodos programados dentro de la red, simplemente resta la presentación de la información receptada en las pantallas a las que el usuario final se prestará a usar.

El uso de varias cajas de texto posibilita que se desplieguen en pantalla la información como el número de nodo de cada sensor, con el cual hayan sido programados, además del dato de temperatura actual, una serie de validaciones para evitar errores de carácter interno como al apertura múltiple del puerto de comunicaciones o como el intento de trabajo con un puerto no existente, así como la validación de distinta información.

En el caso, por ejemplo, de la salida de los datos para poder lograr la activación de los efectores finales que son los ventiladores, esto logrado a través de un circuito de control sobre el cual se detalla más adelante en el texto, de tal forma que, en lo que concierne al programa por ejemplo, esta salida de datos e logra a través de la generación de señales al momento de sobrepasar valores determinados por el usuario como por ejemplo, por defecto del programa, será de 24° centígrados, señales que varían por efecto de control entre potencias de 2 de acuerdo al número de identificación del nodo, 2^{NodeId}, basados en esta

expresión la salida acumulada será conformada por valores de entre 1 hasta 31, para efecto de controlar los cinco ventiladores expuestos en el área física de desarrollo como se muestra en la parte de la topología de la interfaz de control.

5.3.1 Diagrama de Flujo del Sistema

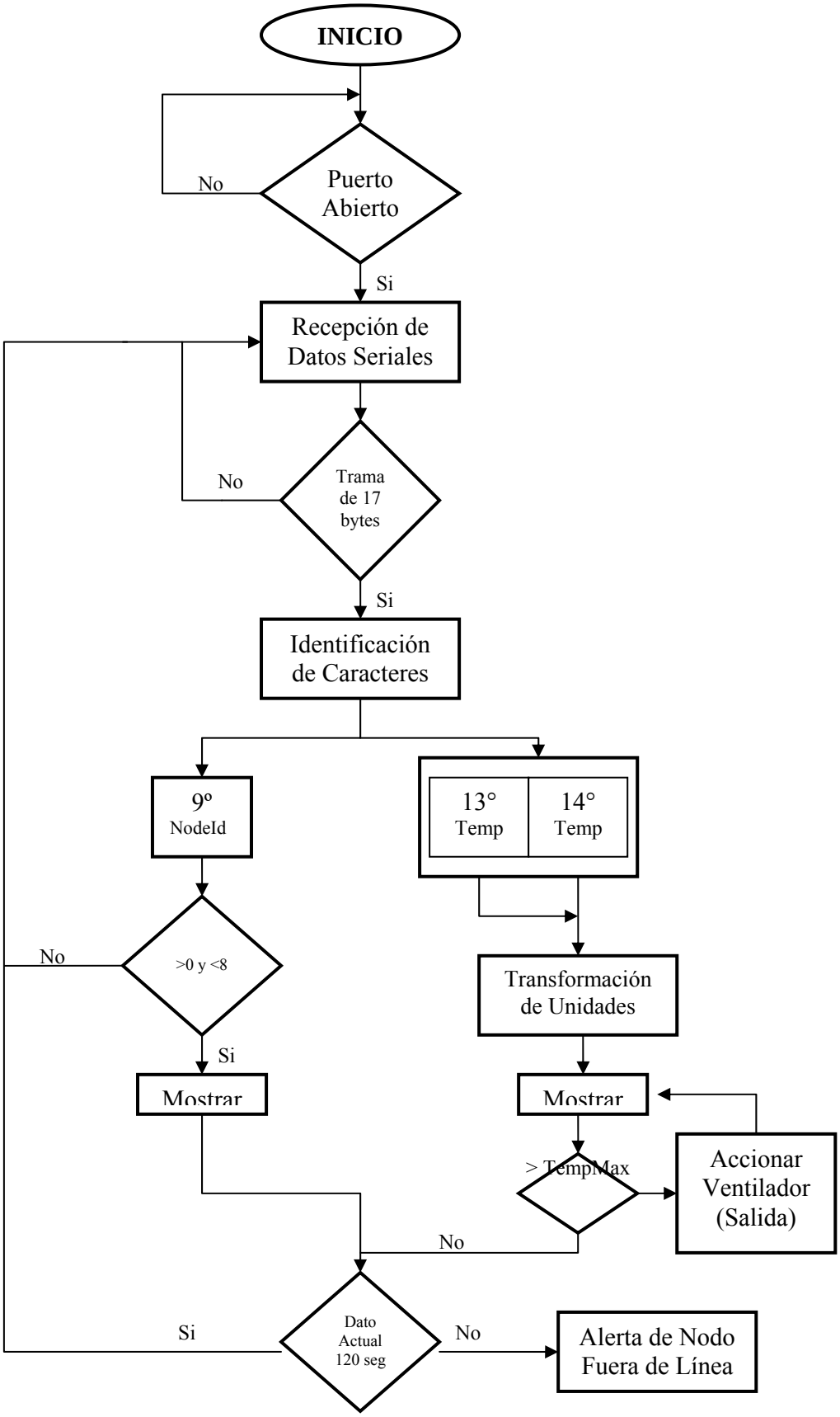


Figura 5. 3: Diagrama de Flujo del Sistema

5.3.2 Detalle de la Interfaz de Monitoreo y Control

Superada la parte de análisis y obtención de la información en l pantalla, lo siguiente corresponde a la visualización de dichos datos en la pantalla del usuario, de tal forma, que basados en la operación típica de los Sensores Inalámbricos dispuestos en la red constituida, las principales características son las siguientes, esto en la interfaz de monitoreo y control desarrollada.

- **Pantalla de Presentación**

Esta pantalla, mostrada al acceder al menú 'Puerto', nos permite acceder a la configuración de las características que tienen que ver con la comunicación, claro esta como es el caso, en lo que tiene que ver con el puerto serial de la PC, es decir lo que se refiere a la velocidad de transmisión, el puerto al cual esta conectada la estación Base, y otras características que son explicadas a detalle a continuación:

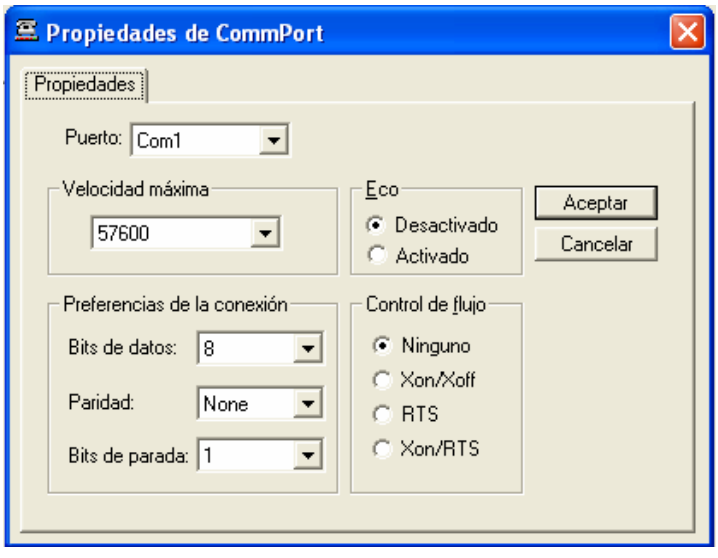


Figura 5. 4: Cuadro de Propiedades del Puerto Serial

En primer lugar, tenemos la posibilidad de cambiar el puerto a través del cual se va a realizar la recepción de la trama serial con la información necesaria para el manejo de la

temperatura en la Red. A continuación tenemos la posibilidad de ajustar la velocidad de transmisión de datos entre los nodos, que por cuestiones de programación de los MOTE's debe estar ajustada a 57600, pero puede ser variada para efectos de recepción de la información.

En la parte que corresponde a las Preferencias de Conexión, tenemos opciones como la selección de los bits de datos, la Paridad y el número de bits de parada. Consta a demás de la opción de seleccionar la activación o desactivación de la recepción de Eco, es decir la réplica de los caracteres ingresados. Y, finalmente, la opción de seleccionar la forma de Control de Flujo para el puerto.

Todas las características mostrada en esta pantalla de ajustes del puerto, para efectos del monitoreo de variables en el sistema, por defecto se encuentran ajustadas a los valores típicos de operación, cualquier cambio podría causar demora o falta de precisión en la información receptada.

- **Menús y Botones de Acceso Rápido**

Una vez que se han ajustado cada una de las propiedades del puerto de comunicaciones serial, ya se puede proceder a la operación de la interfaz en sí, para lo cual nos valemos de las propiedades esenciales que se encuentran implícitas en la estructura visual de la misma, de tal forma que para el detalle de cada una de las propiedades de acceso, se detalla el funcionamiento y operación de cada botón de acceso o menú en la pantalla de usuario.

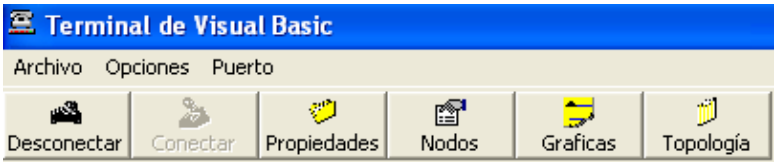


Figura 5. 5: Menú de Acceso

Desconectar; Botón de Acceso directo a la opción de desconexión del puerto de comunicaciones escogido en los ajustes o ventana de propiedades, es decir, internamente, en el programa lo que se realiza a través de esta opción es cerrar el mismo para evitar el flujo de información.

Conectar; Botón de acción contraria al caso anterior, es decir, el acceso directo que nos permite abrir el puerto de comunicaciones escogido en las propiedades, hay que tener en cuenta que los dos botones, tanto el de conexión como de desconexión, complementariamente se activan y desactivan cuando uno de los dos es presionado, esto para evitar la apertura del puerto una y otra vez, o el caso contrario, lo que podría generar algunos errores.

Propiedades; el acceso directo a la ventana de Propiedades del puerto de Comunicaciones, es decir, esta opción nos permite el acceso a la ventana del literal anterior para los ajustes del puerto.

Nodos; opción que nos muestra la pantalla Principal de Visualización, explicada más adelante.

Gráficas; lo que nos permite acceder a la pantalla de visualización de las gráficas del dato Temperatura en función del tiempo, igualmente explicada a continuación.

Topología; acceso a la pantalla de visualización de la distribución de la Red en el área especificada, como se detalla más adelante.

Igualmente, cada una de las funciones explicadas en esta sección, se encuentran distribuidas en cada uno de los menús de la barra principal, repartidos en las tres opciones: *Archivo, Opciones* y *Puerto.*

- **Pantalla Principal de Visualización**

Es la primera pantalla de visualización que se muestra en la interfaz de monitoreo y control, es aquella en la que tenemos la posibilidad de observar detalladamente numérica y gráficamente cada una de las mediciones de los sensores conectados a la Red.

Dentro de las características de la presente pantalla, se debe destacar que originalmente, al abrir el programa, se nos muestra totalmente en blanco, cada una de las mediciones y datos de temperatura se van mostrando de acuerdo al su conexión, es decir van apareciendo según se van encendiendo o anexando a la Red de Sensores.

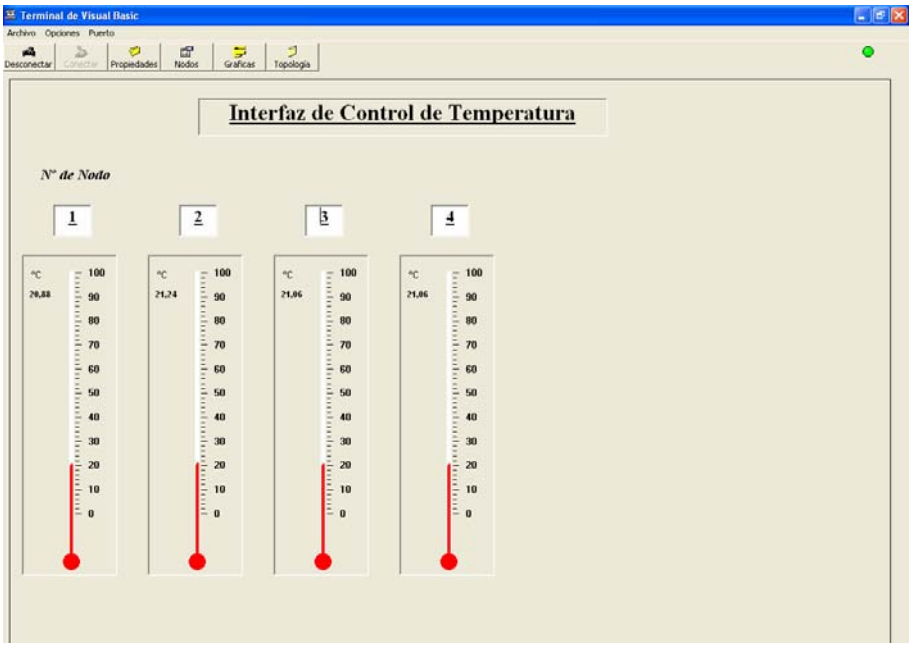


Figura 5. 6: Pantalla Principal de Visualización

Como se muestra en la Figura, la medición de la Temperatura es desplegada en valor numérico y en forma visual, a manera de un termómetro gráfico, en el cual podemos apreciar por nivel el valor mostrado, el especio de la pantalla nos es suficiente para poder tener al mismo tiempo un máximo de 7 sensores en el sistema, esto debido a la cantidad de equipamiento disponible.

Además, se debe destacar la opción de la interfaz de detectar por tiempo cuando alguno de los sensores del sistema se desconecta o deja de enviar información a la estación base, de tal forma que esto se puede apreciar visualmente a través de la coloración distinta en esta pantalla del termómetro correspondiente, así como se muestra en la Figura de a continuación.

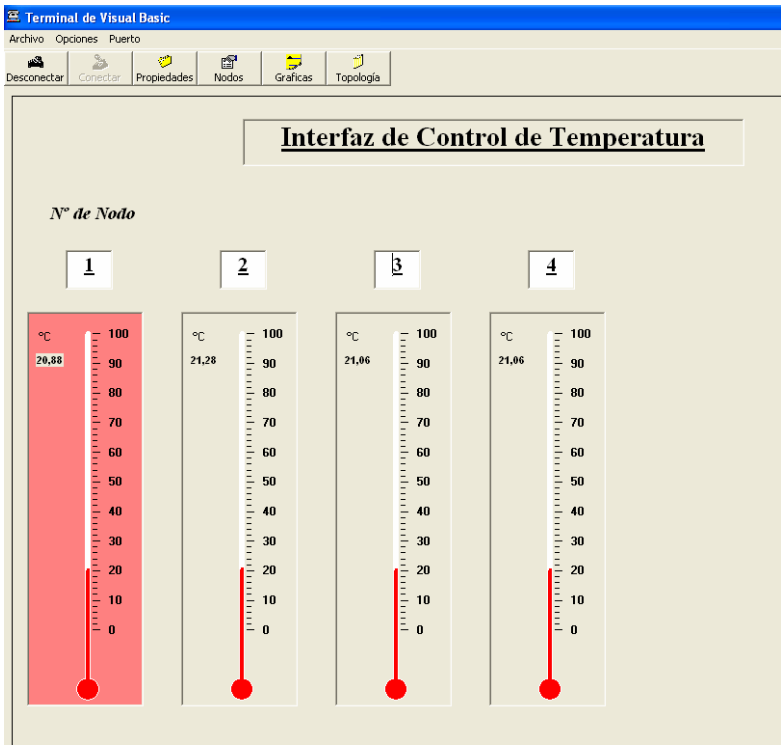


Figura 5. 7: Desactivación de Nodos

5.3.3 Pantalla de Historial Gráfico de Datos en el Tiempo

Dentro de las opciones de visualización, se ha incluido también la posibilidad de observar el desempeño de cada uno de los nodos de la Red Sensorial y sus mediciones a lo largo del tiempo de conexión, es decir que para este caso, lo que se permite es poder acceder a un gráfica en la que se muestren los datos de temperatura medidos por cada sensor a lo largo del tiempo de conexión, con la posibilidad de habilitar o no la grafica individual de los mismos, es decir, tenemos un color distinto para cada una de las señales, en un máximo de 7 nodos más la estación base, marcando o no las pestañas ubicadas al lado derecho del área de gráfico, podemos visualizar el desempeño temporal de cada nodo.

En general, la pantalla mostrada para esta opción se describe por si sola, muestra el área de gráfico y a un lado, de acuerdo a su activación, a las pestañas de selección de cada uno de los nodos posibles con el identificador del color que lo determina.

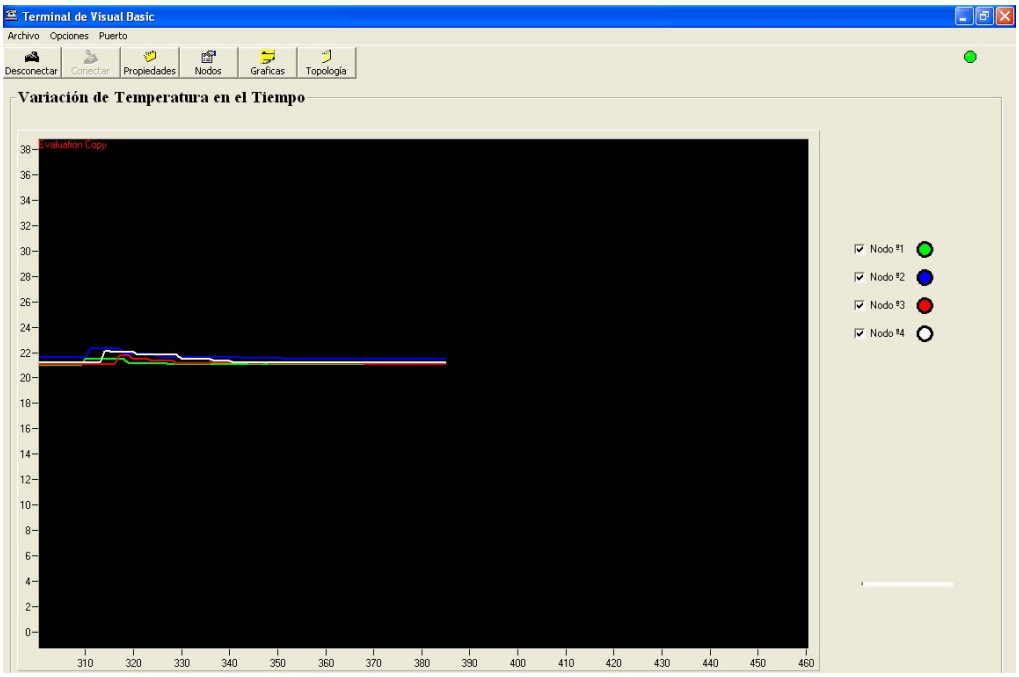


Figura 5. 8: Pantalla de desempeño en el Tiempo de las Mediciones de Temperatura

- **Pantalla de Visualización de la Topología de la Red Sensorial**

Basándose el la idea propuesta por el programa original, proporcionado por la empresa productora de la tecnología inalámbrica de sensamiento, en esta pantalla, lo que se muestra es la forma en la que los sensores se encuentra distribuidos a lo largo del área de trabajo, que para el caso, representa el área industrial de comunicaciones, en donde se va a controlar la temperatura parámetro importante para poder asegurar el correcto funcionamiento y desempeño de los equipos inmersos en su generación, como lo es el caso de PC, equipos móviles, radios entre otros.

Por lo tanto, esta parte de la visualización, dentro de la interfaz nos permite ver la forma física en que los sensores han sido distribuidos en dicha área, de igual forma que en las otras opciones de visualización, en este caso, el aparecimiento de un nuevo sensor en la red se verifica a través de imágenes de extensión GIF que representan a los nodos, así mismo con la ventaja de poder reconocer cuando alguno de los sensores ha salido de

operación ya sea por mantenimiento o por daño del mismo, en cuyo caso, la imagen GIF dejara de correr y nos mostrara un texto ayuda sobre la misma indicando que el nodo esta fuera de servicio.

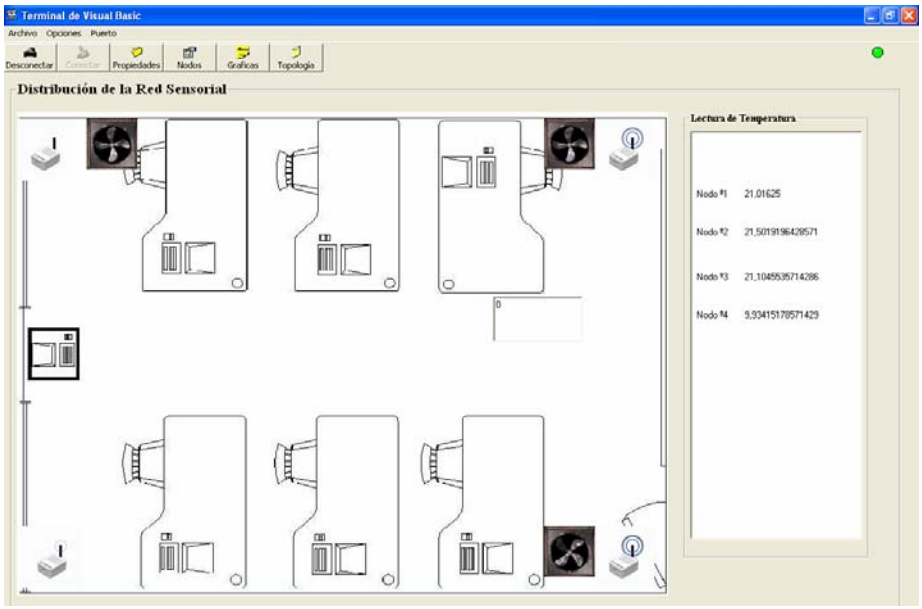


Figura 5. 9: Pantalla de Distribución de la Red Sensorial Inalámbrica

Como se puede apreciar en la Figura anterior, además de los gráficos de los nodos, también tenemos la posibilidad de acceder al valor numérico de la medida de temperatura de cada uno de los sensores activos junto con su identificador en el lado izquierdo de la interfaz, opciones que irán apareciendo al transcurso en que van apareciendo los mismo.

Finalmente, ya en lo que tiene que ver con la parte de control de la variable seleccionada, la temperatura, junto a cada una de las imágenes GIF de los nodos, una vez que se han activado las señales de salida del circuito de control, por sobrepasar los valores determinados de la variable por el usuario, aparecerán y se ejecutarán las animaciones de ventiladores, los mismo que muestran la acción física realizándose en el efector final, es decir que son la representación gráfica de lo que está sucediendo, una vez que el valor de

temperatura de alguno de los nodos activos de la Red Sensorial detecta que existe un valor por encima de lo permitido, entonces la interfaz genera una señal de salida por la cual se produce la activación de los ventiladores instalados en el área industrial de comunicaciones. De igual forma, si se logra establecer de nuevo la variable en el rango permitido dicha señal es cortada y la animación detenida.

- **Barra de Estado de Conexión**

La barra de estado, dentro de la interfaz de monitoreo y control, corresponde a una sección en la parte inferior de la pantalla en la cual se puede observar el estado y el tiempo de conexión, así como los datos de ajuste de la comunicación implementados, ya sea por el usuario o por defecto de la interfaz; datos tales como la velocidad de transmisión, BIT de parada, número de bits.

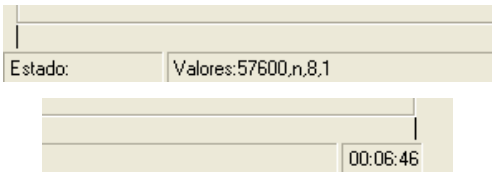


Figura 5. 10: Barra de Estado de la Conexión

5.4 INFORME DE COSTOS.

Implícito en la realización y desarrollo del proyecto que se presenta en el documento, se encuentra la inversión económica, tanto para la adquisición de los equipamientos como para la implementación de la parte práctica, generación de información y fuentes de apoyo del sistema.

Esta sección del documento presentará un informe detallado de los materiales necesarios para la culminación del mismo, es decir, los distintos materiales usados tanto

en la implementación como en la documentación, así como los medios de información utilizados.

Como base de la implementación, el equipamiento es esencial debido a su principal operación, en el presente caso, justamente serán los Sensores de características Inalámbrica son los protagonistas, de tal forma que para la parte operativa, del Kit adquirido, por las características del proyecto, se utilizaron los necesarios para la conformación de la Red necesaria.

A continuación se muestra el listado de los mencionados equipos y dispositivos, así como sus costos de mercado.

A causa de que las Redes de Sensores Inalámbricos consisten en un gran número de nodos sensores, el costo individualizado de los mismos es muy importante al momento de justificar el costo total de la Red en sí. Si el costo de la red de sensores es muy por encima que el uso de redes de sensores tradicionales, entonces su implementación no se encuentra justificada por su alto costo. Como resultado, el costo de cada sensor que va a conformar un nodo de la totalidad de la red, tiene que mantenerse en bajos precios de mercado.

Tabla 5. 1: Informe de Costos de Equipos

Nombre del Paquete	Cantidad Utilizada	Elementos del Paquete	Precio
MOTE-KIT2400CB (Paquete Comercial de Desarrollo)	1	✓ 8 MICAz tableros procesador/radio (MPR2400CA)	$3,444.00
		✓ 4 Tableros Luz / Temperatura / acústico / sísmico / magnetómetro (MTS310CB)	
		✓ 3 Tableros Luz / Temp / acústico (MTS300CB)	
		✓ 1 Tablero de Interfase Serial RS-232 PC (MIB510CA)	
		✓ Aplicaciones pre-programadas con XSensor para desarrollo fuera del paquete	

Las nuevas tecnologías y tendencias en operación de Redes Sensoriales han hecho que un sistema de radio Bluetooth rodee el precio de hasta los 10$. Así mismo, el precio de un PicoNode se encuentra hasta por menos de 1$. Lo que nos demuestra que el precio de un nodo debe ser me por debajo de 1$, esto para que la red sensorial entera sea justificada y completamente útil. El costo de un radio Bluetooth, el cual, ya es conocido llega a ser un dispositivo de bajo costo, es incluso 10 veces más costoso que el precio ofrecido para sensores.

Nótese que un sensor o nodo, además tiene también algunas ventajas de operación y unidades adicionales, así como por ejemplo las unidades de sensamiento y procesamiento descritos en la sección correspondiente al hardware; incluso, podemos encontrar fácilmente en el mercado que los sensores se encuentran equipados con sistemas de localización, movimiento, GPS, generación de energía, entre otros, dependiendo de la aplicación a donde vayan dirigidos los distintos nodos.

CAPITULO 6

PRUEBAS Y RESULTADOS

6.1 ADQUISICIÓN DE DATOS DE LA RED.

Una vez que se ha constituido toda la infraestructura de la Red Sensorial, sus equipos de adquisición de la información y la parte de la misma dedicada al control de la variable Temperatura, con la configuración de la interfaz en la cual se visualiza el desempeño de la Red y se realiza desde una estación, en este caso un PC, el control; es entonces que podemos realizar una serie de pruebas sobre el ambiente simulado, como es el caso un área destinada a Sistemas de Telecomunicaciones avanzados, para verificar que el funcionamiento de la implementación es el correcto. De tal forma que se somete al espacio físico a ciertas alteraciones sobre la variable Temperatura, es decir, se manipula de manera aleatoria, subiendo o bajando la misma en los lugares cercanos a los dispositivos de adquisición, los sensores de características inalámbricas.

Dichas alteraciones se ejecutan para verificar que la Red conste de parámetros de funcionamiento típicos para ser aceptados en áreas industriales; parámetros entre los que podemos enumerar:

- Velocidad de Respuesta
- Precisión
- Tolerancia
- Alcance

- Ambiente de Instalación
- Consumo de Energía
- Entre otras…

De cada una de las características enumeradas, su cumplimiento se lo verifica con las pruebas y alteraciones a las que se expuso a la Red, a continuación se presenta de forma detallada las mencionadas pruebas y su influencia sobre los dispositivos así como el funcionamiento de la Red Sensorial; verificando de esta manera el nivel de aceptación con respecto de las características del listado anterior.

En primer lugar, y no coincidencialmente sino por razones de operación, el tiempo de respuesta de los dispositivos, frente a cambios bruscos en la variable a la que están sujetos, es primordial para determinar cuan útiles nos pueden ser en caso de situaciones de emergencia dentro de un área especifica de operación. La verificación de este ítem se obtiene mediante la alteración de manera abrupta sobre la temperatura, dicha alteración puede ser llevada a cabo a través de instrumentación externa, ya sea un calefactor o un simple secador de cabello, y usando mediciones alternas con termómetros ambientales; de tal manera que cambiando drásticamente el valor de la temperatura en un sector especifico de la Red, la Velocidad de Respuesta puede ser comprobada a dos niveles, es decir, es dependiente de la velocidad de lectura a la que esta programada la interfaz de Monitoreo y Control implementada, que para el caso depende de un temporizador que determina la frecuencia de adquisición.

En segundo lugar, depende también de la velocidad de transmisión a la que se encuentran configurados los puertos ya sea de recepción como de transmisión, es decir el parámetro que se determina al momento de realizar la compilación y programación de cada uno de los sensores individuales de la Red, ajustada a un valor típico, por características de los

equipos, de 57600 Kbps, lo que influye directamente en esta velocidad de respuesta sobre todo en lo referente a la medición del parámetro, puesto que la velocidad a la que las señales de activación o desactivación de los efectores finales no es de gran relevancia por formar parte de un sistema de control de tipo ON/OFF, que como lo demuestra su teoría de aplicación, no depende de un tiempo muerto ni de histéresis aplicadas, simplemente funciona en paralelo con las señales recibidas y su constante cambio en el tiempo.

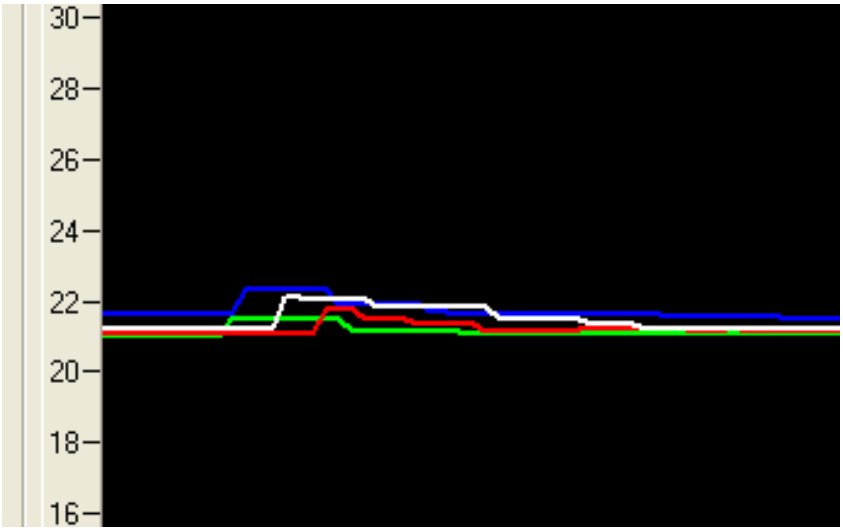


Figura 6. 1: Comprobación de Tiempo de Respuesta

Como se puede apreciar en la Figura anterior, la alteración sobre la temperatura en cualesquier punto de la Red, debido a la programación de la interfaz, la información es adquirida y desplegada en pantalla, en este caso por medio de la gráfica, y su tiempo de respuesta es idéntico al cambio que se observa en un termómetro Ambiental normal, de tal forma que su capacidad para recibir y denotar el cambio en el parámetro es considerablemente suficiente para el ambiente en el que se desarrolla, es decir, dentro del ambiente industrial de Telecomunicaciones reducido, su velocidad de respuesta frente a un cambio en la temperatura es más que bueno para realizar el control en caso de sobrepasar el valor permitido de operación.

Para la parte concerniente a la precisión, de igual manera tenemos dos frentes, una parte dependiente del hardware que constituye a los dispositivos, es decir los tableros de sensamiento, aclarando de por medio que esta dependencia es de la precisión del Termistor ubicado en el MTS310CB, acerca de lo cual se explica en la sección en la que se describe el hardware de la Red. El otro aspecto en el cual se puede medir la precisión, es sobre los datos desplegados en la pantalla de la interfaz de Monitoreo y Control, y en este caso lo que determinado por la tasa de errores inmersa en la comunicación serial, esto, teniendo en cuenta que se ha utilizado la estación base MIB510; como es conocido, de acuerdo a la cantidad de información que se envíe.

Obviamente el tamaño de la información y su velocidad están directamente relacionados con el tamaño de la Red, es decir, respecto al Número de Nodos que se tenga en su constitución. Una trama serial de 17 bytes implica una carga de datos que debe ser transmitida a la velocidad especificada de 57600 Kbps, y esto por cada uno de los nodos de la Red, lo que nos destina a la posibilidad en aumento proporcional con dicho número de nodos y baja del nivel de precisión, justamente por la trama de bytes grande.

Otra manera de permitir la opción de medir el nivel de precisión de la Red en su operación y la forma en que nos es mostrada la información del sistema; es a través de la comparación de los valores obtenidos en la pantalla de usuario del Programa MoteView y la información que nos presenta la interfaz de control y monitoreo del presente proyecto. Claro está, que para este procedimiento es necesario poder ajustar al o a los nodos a una temperatura fija, esto debido a que no podemos operar en las dos ventanas al mismo tiempo, por el conflicto de uso de puerto para la lectura de datos. Es por tal motivo que esta prueba se ejecutó por medio de la Programación de uno de los MOTE’s y el alcance de ese

valor constante de temperatura o valor específico de lectura, un dato fijo pero no exacto, es decir una lectura con cifras decimales, que permita ver la diferencia entre la información de cada uno de los sistemas de Monitoreo; a través de lo cual se obtuvo el siguiente resultado:

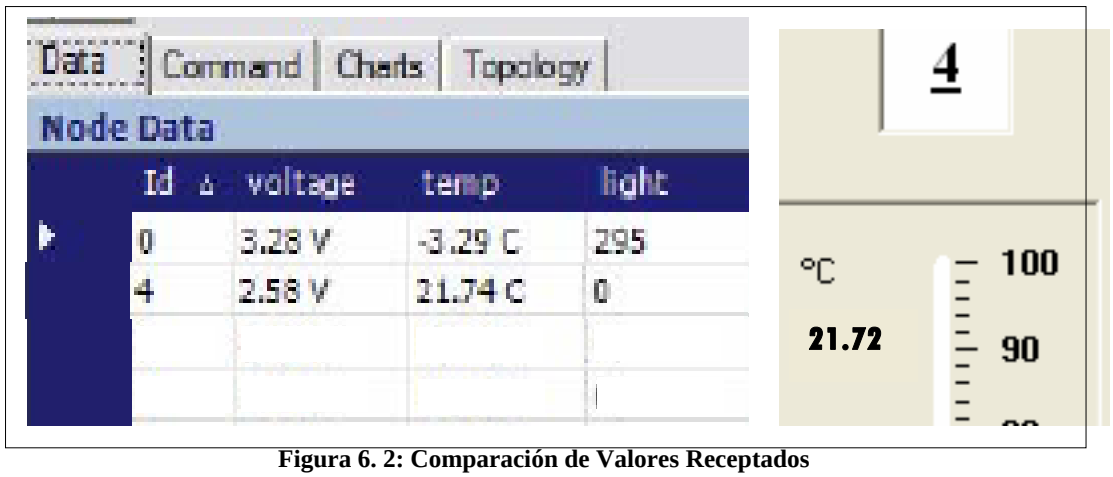


Figura 6. 2: Comparación de Valores Receptados

Como se puede apreciar, la comparación entre las dos mediciones, da como resultado valores muy cercanos, con fallas apenas en las cifras decimales, y teniendo en cuenta la tasa de errores de la comunicación y la forma poco accesible de mantener fija la temperatura, se podría afirmar la corrección y total precisión de la información mostrada por la interfaz desarrollada para el proyecto.

Definitivamente, al hablar de tolerancia, estamos expuestos a las especificaciones técnicas de los equipos, es decir que vamos a tener que estar completamente apegados a lo que el sensor de temperatura físico, en este caso, parte del MTS, pueda tolerar para su correcto funcionamiento, de tal forma, que conociendo que el rango máximo de medición especificado por el fabricante es de 50° centígrados, las pruebas orientadas en esta sección van mas bien hacia la verificación de cuan susceptible al cambio drástico de la variable puede ser un MOTE, para lo cual se realizaron una serie de cambios instantáneos de

temperatura en los sensores, con la interfaz funcionando y se pudo notar que frente a cambios violentos, puede suceder uno de dos casos; o la trama recibida en el Terminal de Visual Basic es diferente de tamaño, y por una validación es desechada evitando mostrar datos erróneos; o existe una alteración inusual en el dato receptado, mostrando en lapsos insignificantes de tiempo, caídas o alzas abruptas de los valores receptados y desplegados en el monitor de la estación base, frente a lo cual podemos afirmar que los sensores son bastante tolerantes a cambios y condiciones adversas del ambiente, pero debido a la forma de recepción de la información en la Interfaz.

La tolerancia a fallas es otro factor importantísimo que analizar en la estructuración de las redes sensoriales, fallas que pueden ser enlistadas por distintos casos, de tal manera que un sensor podría fallar o ser bloqueado debido a un error de energía, por tener daños físicos o interferencia ambiental. Es tanto así que lo que se debe tener en cuenta es que no se vean involucrados en el desarrollo de toda la red estos errores individuales, afectando por completo a la red entera. Por lo que se denomina como tolerancia a fallas a la habilidad de sustentar las funcionalidades de la Red Sensorial sin tener que interrumpirlas debido a fallas individuales de nodos del sistema.

Esta característica, es marcada mediante Rk(t) y existe toda una teoría acerca de su tolerancia a fallas basados en la distribución de Poisson para capturar la probabilidad de no tener fallas dentro de un intervalo de tiempo (0; t), a través de los cual, dicho modelamiento ha dado como resultado la siguiente expresión:

$$R_k(t) = \exp(-\lambda_k t)$$

Ecuación 6. 1: Tolerancia de Fallas

; Donde λ_k y t son la tasa de fallas y el período de tiempo

respectivamente de un sensor k

Como resultado del nivel de tolerancia a las fallas, es la aplicación de la red sensorial, y por su puesto también la serie de esquemas desarrollados en el diseño de la misma.

Además, podemos también expresar la forma de medir la tolerancia de los equipos, a la forma en que se ven afectadas las mediciones y comunicación o desempeño de la red, debido a la existencia de ciertos posibles limitantes o elementos de riesgo en el mencionado ambiente de la implementación. Esta prueba se realizó con la infiltración de ciertos elementos comunes en un área de comunicaciones avanzadas con capacidad de generar señales de transmisión y/o recepción, equipos como computadores, equipos Gíreles, equipos con tecnología Bluetooth, radios, generadores de frecuencias, entre otros.

Esta situación podría afectar directamente a las comunicaciones entre los nodos y la estación base. Sin embargo, teniendo en cuenta la tecnología o normativa sobre la cual operan los WSN, las frecuencias se encuentran ampliamente limitadas a no ser afectadas con este tipo de señales, de tal forma que la interferencia no es un factor de riesgo a menos que se maneje la misma tecnología en equipamiento alterno, para lo cual, las soluciones se amplían al re – direccionamiento de las señales transmitidas por los nodos.

Del Alcance que pueden tener los equipos, podemos decir que cada uno de ellos, de a cuerdo al modelo, o al tipo que haya sido adquirido de la casa, en el presente caso

Crossbow, tenemos las capacidades de alcance también, dependientes claro esta, de la capacidad que nos permita el modelo de la antena, su propagación y la tecnología de transmisión.

Para el proyecto desarrollado, los tableros de transmisión – recepción adquiridos, los MPR2400, nos permiten un alcance de hasta 30 metros de acción a la redonda, es decir un área suficientemente buena como para lograr la implementación de la Red, destinada a sectores en los cuales se necesiten ciertos parámetros, todos los que permitan los sensores en cuestión.

De tal forma que, para la realización de estas pruebas, lo que se hizo fue el alejamiento físico de los nodos de la estación base, la PC, manteniendo una variación de la temperatura para su verificación en el monitor a través de cualquiera de las interfaces de monitoreo, una vez alcanzado el límite de distancia física de los mismos, la señal de no transmisión o el repentino estado estático de las lecturas determinaría la distancia máxima de acción, lo que se logró verificar también como parte complementaria de la cantidad de energía, es decir que para el caso, el nivel de voltaje de las señales verificado en la interfaz MoteView determina también la distancia máxima, puesto que con una señal de energía más baja, dicha distancia se acorta.

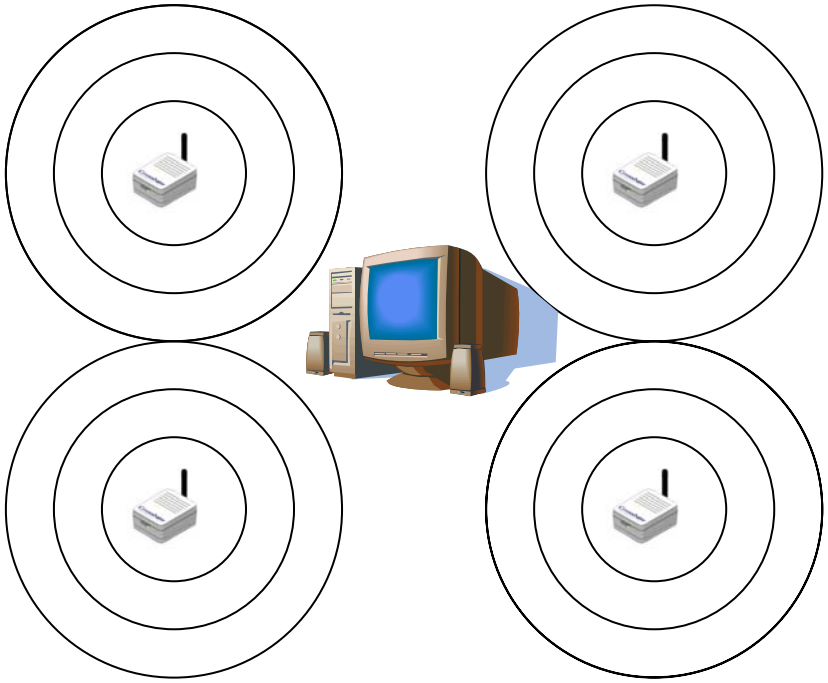


Figura 6. 3: Área de Acción de los MOTES

Otra de las características importantes que deben ser descritas, para el caso del presente proyecto, es el área de desenvolvimiento de la Red Sensorial Inalámbrica, y los factores que intervienen en él. Los sensores se encuentran densamente distribuidos muy cerca o directamente dentro del fenómeno que va a ser analizado. Además en otros casos también, los nodos trabajan en lugares en donde no reciben atención por ser áreas remotas, por lo que los sensores deberían ser desplegados en áreas donde su tremenda utilidad no sea desperdiciada, es decir en sectores donde puedan ser explotados al máximo como en:

- Intersecciones con mucho tráfico,
- En el interior de maquinaria pesada,
- En el fondo del océano,
- Dentro de huracanes,
- En la superficie del mar durante un tornado,
- En campos contaminados biológica o químicamente,

- En campos de batalla cerca de líneas enemigas,
- En el hogar o en grandes construcciones,
- En grandes industrias,
- Para control animal,
- Para control vehicular, y
- En ríos que muevan grandes corrientes.

Este listado muestra solo unos pequeños ejemplos de lugares adversos en los cuales los sensores serían de gran utilidad. Todo eso debido a que su constitución los hacen aptos para trabajar bajo altas presiones (debajo del mar), o bajo fríos extremos como para ser ubicados para monitoreo en el motor de alas de aviones o en ambientes extremadamente inhóspitos, como por ejemplo en regiones de la Antártica, o simplemente en zonas con extremadamente altos índices de ruido.

El consumo de energía es un factor muy relevante, debido que a lo largo de la realización de las distintas pruebas y controles de operación de los equipos, la duración de las baterías adjuntas a cada uno de los equipos, responde de manera que su agotamiento evita el buen desempeño de los nodos. Los WSN, siendo parte de los dispositivos de micro-electrónica, pueden solamente ser equipados con limitadas fuentes de energía (<0.5 Ah, 1.2 V). En algunos escenarios de aplicación, el tiempo de vida de los nodos tiene una muy alta dependencia del tiempo de vida de las baterías, justamente. En una red multihop Ad-Hoc, cada nodo cumple una doble funcionalidad, tanto como generador de datos, como de enrutador de información.

El malfuncionamiento de algunos de los nodos podría causar muy significantes cambios topológicos y cambios estructurales que adviertan una nueva topología. Por estas razones en cualquier red sensorial, es de vital importancia la excelente administración y conservación de energía.

Básicamente, el consumo de energía puede ser clasificado en tres grupos o dominios:

1. Sensamiento,
2. Comunicación; y,
3. Procesamiento de Datos.

El consumo en el primer caso, en sensamiento, varía con la naturaleza de la aplicación, así por ejemplo, un sensamiento esporádico consume muchísimo menos energía que un monitoreo continuo de una variable.

En el segundo caso, el consumo cuando hay comunicación de datos y procesamiento, en el tercer caso, son temas discutidos, porque en comunicaciones es cuando se tiene el mayor consumo de energía, esto involucra tanto a la transmisión como a la recepción. Esto puede observarse en pequeños rangos de comunicación con poca radiación de poder (_0 dbm).

6.2 ANÁLISIS DE RESULTADOS

De los resultados obtenidos durante serie de pruebas realizadas sobre la implementación se ha logrado determinar algunas de las características como las mencionadas en la sección anterior, de su nivel en la operación del sistema completo, se ha

podido determinar que los datos mostrados son fieles a la realidad y tienen la verificación adecuada a través del hardware utilizado.

Una vez probados, verificados y en uso constante, la vida útil de cada uno de los dispositivos va formando una malla segura y confiable para mantener el completo control sobre una variable, que en el presente proyecto es la temperatura, y lo que permite es la opción sobre la generación de un sistema completo de acción para mantener en operación un área destinada a trabajo a niveles industriales, verificando con la serie de pruebas realizadas que puede o no causar variaciones, afectaciones o errores del conjunto, es decir de la Red en sí.

Las distintas alteraciones en el tiempo y la generación de un historial gráfico, en el caso de la opción de visualización presente en la interfaz desarrollada, permiten un análisis genérico de lo que se ha ido sucediendo, de tal forma que por medio de la mencionada opción se admite la permanencia de bases de datos, claro está, existen relaciones aplicadas, como es el caos inmediato del software de la casa, con el que fueron adjuntos los equipos, MoteView, que además de los análisis temporales no permiten también la generación de tablas muestreadas en otros programas para poder analizar ciertos parámetros de acción inmediata.

CAPITULO 7

CONCLUSIONES Y RECOMENDACIONES

7.1 CONCLUSIONES

La planeación, desarrollo y culminación a través de pruebas y los distintos análisis realizados sobre el presente proyecto han causado la generación de una serie de elementos de juicio que permitirán al lector del documento tener en cuenta ciertos parámetros de referencia, esto al tratar con este tipo de tecnología, las Redes Sensoriales Inalámbricas.

De lo mencionado, podemos afirmar que entre los principales motivos de fallas y errores de este tipo de equipamiento, está la falta de familiarización con sus principales principios de funcionamiento de allí que, una exhaustiva y comprensiva lectura de los manuales de operación y desarrollo de los Sensores.

Por lo tanto, a continuación se enlista una serie de conclusiones obtenidas de la manipulación de los equipos, de su operación y trabajo, obtenidos mediante la experimentación y la implementación del presente proyecto de tesis:

- Los sensores tienen un alto grado de sensibilidad a daños y fallas de los equipos es parte esencial al operar con los mismos, esto debido a que su conformación electrónica puede ser alterada o mal usada por su estructura modular.

- La exposición de los equipos sin las protecciones anexas, como lo son las carcasas y distintos armazones, a condiciones adversas, pueden producir fallas operativas como el no encendido de los mismos, o lecturas erróneas.

- De la comunicación de los nodos con la estación base y entre sí, podemos concluir, por la presencia de obstrucciones en el área de implementación, que los nodos, son autoconfigurables, es decir que en caso de interrupción, ellos buscan un nodo cercano a través del cual puedan llevar la información a la base.

- La programación de los módulos de comunicación sensorial es de suma atención, errores de forma y no de fondo pueden ser compilados sin tener consecuencias en los programas internos, sin embargo al momento de recibir la información, dichos errores pueden derivar en lecturas no confiables ni necesarias.

- El hecho de que los sensamientos de Luz y Temperatura compartan el mismo canal o dispositivo es un característica que se debe tener presente, esto debido a que, los programas pueden ser correctamente compilados y grabados sobre los equipos, sin embargo su funcionamiento produciría, por cuestiones de dependencia de las dos variables, errores de lectura, es decir, si medimos una de las dos, el otro dato deberá estar restringido para operación.

- La capacidad energética es un factor importante para poder determinar los montos exactos de operación referidos a distancia, esto debido a que los equipos, por la cantidad de energía remanente en las baterías pueden o no alcanzar sus rangos

máximos de comunicación radial, baja potencia de alimentación, deriva en una comunicación pobre o nula.

7.2 RECOMENDACIONES

- Planear una etapa de familiarización con el equipamiento, meditando en su estructura la lectura, análisis y comprensión de cada una de las hojas técnicas y manuales de operación tanto del Software como del Hardware con los cuales se esté trabajando.

- Tener en cuenta cada una de las indicaciones de riego del equipo recomendados por el proveedor, esto para evitar la posible afectación de los Sensores y los distintos módulos en su funcionamiento.

- Ya una vez inmersos en el trabajo con los Nodos Sensores, verificar el correcto armado, estado y configuración de los mismos, relaciones de energía, constitución modular, correspondencia de operación, entre otras.

- Debido a los alcances del programa se puede explotar su potencial desarrollando aspectos en el software de aplicación tales como la representación física real a escala de la topología a través de un sistema de localización de cada nodo.

- Se puede recomendar a instituciones y en especial a la universidad en realizar mas estudios sobre este tipo de tecnologías las cuales están en evolución y seguramente tendrán gran acogida a futuro, mediante el diseño y construcción de este tipo de equipos o ya sea mediante el estudio de redes explotando los recursos que pueden ofrecer los sensores inalámbricos.

ANEXO 1

CÓDIGO EN VISUAL BASIC DE LA INTERFAZ

```
Option Explicit
Dim Ret As Integer      'Entero auxiliar.
Dim Temp As String      'Cadena auxiliar.
Dim hLogFile As Integer 'Controlador de archivo de registro abierto.
Dim StartTime As Date   'Almacena la hora de inicio del cronómetro del puerto
Dim HexA As String 'Guarda el valor hexadecimal de la trama serial
Dim temphaux(7) As Double 'Arreglo que almacena valores de temperatura en grados centígrados de cada nodo
Dim tiempo As Long 'variable de almacenamiento del tiempo de conexión
Dim descx(7) As Double      'variable  de desconexión de sensor
Dim contdesc(7) As Integer  'contador de tiempo de desconexión de sensor
```

```
Private Sub Form_Load()
Dim CommPort As String     'Nombre de designio del puerto
Dim Handshaking As String  'Opción de manejo del COM
Dim Settings As String     'Cadena que guarda opciones del puerto
Dim i As Integer           'Contador auxiliar

   On Error Resume Next
```

'Establecimiento de las características de visualización de la pantalla principal de la interfaz como colores, longitud de texto mostrado, tamaño entre otros.

```
   txtTerm.SelLength = Len(txtTerm)
   txtTerm.SelText = ""
   txtTerm.ForeColor = blue
```

'Inicialización de parámetros y variables globales del programa, limpieza de cajas de texto, se ocultan las variables no necesarias o que deberán aparecer durante el transcurso de la Operación de la interfaz y del sistema entero, tales como los gráficos de los nodos, la información sobre las mediciones y su estado.

```
    Temptiem.ClearData      'Se limpia el Área de Gráfico
    Text2(0).Text = ""      'se limpia la caja de texto para el nodo 0 (Estación Base)
    HexA = 0          'Encere la variable que guarda la Trama Hexadecimal
    tiempo = 0              'Inicialización en cero del tiempo de conexión
    Graficos.Visible = False  'Ocultamos Frame de Gráficos
```

‘Ocultamos cada una de las cajas de Texto que muestran información de los nodos de la red tales como Identificador de Nodo, Longitud de la Trama Receptada y Dato de Temperatura

```
    For i = 0 To 7
       Text4(i).Visible = False
       Text3(i).Visible = False
       Text2(i).Visible = False
    Next i
```

```
'Establecimiento del título del Prgograma
App.Title = "Programa de Monitoreo y Control de Temperatura en WSN"
```

```
'Establece la luz indicadora de estado
imgNotConnected.ZOrder
```

```
' Centra el formulario
frmTerminal.Move (Screen.Width - Width) / 2, (Screen.Height - Height) / 2
```

```
'Se carga la configuración del registro
Settings = GetSetting(App.Title, "Properties", "Settings", "")
If Settings <> "" Then
   MSComm1.Settings = Settings
   If Err Then
      MsgBox Error$, 48
      Exit Sub
   End If
End If
```

```
CommPort = GetSetting(App.Title, "Properties", "CommPort", "")
If CommPort <> "" Then MSComm1.CommPort = CommPort

Handshaking = GetSetting(App.Title, "Properties", "Handshaking", "")
If Handshaking <> "" Then
   MSComm1.Handshaking = Handshaking
   If Err Then
      MsgBox Error$, 48
      Exit Sub
   End If
End If

Echo = GetSetting(App.Title, "Properties", "Echo", "") ' Echo
On Error GoTo 0
End Sub
```

```
Private Sub Form_Resize()
  'Sitúa la luz indicadora de estado
  Frame1.Left = ScaleWidth - Frame1.Width * 1.5
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
   Dim Counter As Long                 'Contador Auxiliar

   If MSComm1.PortOpen Then
     Counter = Timer + 10        ' Espera 10 segundos para transmitir los datos.
     Do While MSComm1.OutBufferCount
       Ret = DoEvents()
       If Timer > Counter Then
         Select Case MsgBox("Imposible enviar los datos", 34)
           ' Cancelar.
           Case 3
             Cancel = True
             Exit Sub
           ' Reintentar.
           Case 4
             Counter = Timer + 10
           ' Ignorar.
           Case 5
             Exit Do
         End Select
       End If
     Loop

     MSComm1.PortOpen = 0
   End If
End Sub
```

```
' Llama a la rutina mnuOpen_Click para alternar entre conectar y desconectar.
Private Sub imgConnected_Click()
   Call mnuOpen_Click
End Sub
```

```
' Llama a la rutina mnuOpen_Click para alternar entre conectar y desconectar.
Private Sub imgNotConnected_Click()
   Call mnuOpen_Click
End Sub
```

```
' Llama a la rutina mnuHangup_Click para alternar entre conectar y desconectar.
Private Sub mnucolgar_Click()
   Call mnuHangup_Click
End Sub
```

```
' Alterna la propiedad DTREnabled del puerto de Comunicación Serial
Private Sub mnuDTREnable_Click()
   MSComm1.DTREnable = Not MSComm1.DTREnable
   mnuDTREnable.Checked = MSComm1.DTREnable
End Sub
```

```
'Utiliza Form_Unload, ya que tiene código para comprobar si hay datos no enviados
y si hay un archivo de registro abierto.
Private Sub mnuFileExit_Click()
   Form_Unload Ret
   End
```

```
End Sub
```

‘ Habilita y muestra la pantalla de Visualización de Gráfica Temperatura en función del Tiempo

```
Private Sub mnugraf_Click()
   Graficos.Visible = True
   Topologia.Visible = False
End Sub
```

```
' Alterna la propiedad DTREnable para colgar la línea.
Private Sub mnuHangup_Click()
   On Error Resume Next
                        tiempo = 0          ‘Encera la variable tiempo para inicializar en caso de volver a conectar
   MSComm1.Output = "ATH"      ' Envía la cadena de colgar
   Ret = MSComm1.DTREnable     ' Guarda el valor actual.
   MSComm1.DTREnable = True    ' Activa DTR.
   MSComm1.DTREnable = False   ' Desactiva DTR.
   MSComm1.DTREnable = Ret     ' Restablece el valor anterior.
   mnuHangup.Enabled = False‘Deshabilita el botón de colgar la conexión
   tbrToolBar.Buttons("HangUpPhone").Enabled = False
   mnuOpen.Enabled = Trae     ‘Habilita el Botón de Conexión
   mnucolgar.Enabled = False   ‘Deshabilita el menú para cerrar la conexión

   ' Si el puerto continúa abierto, lo cierra
   If MSComm1.PortOpen Then MSComm1.PortOpen = False

   ' Notifica el error al usuario
   If Err Then MsgBox Error$, 48
   tbrToolBar.Buttons("Conectar").Enabled = True
   mnuHangup.Enabled = False
   tbrToolBar.Buttons("HangUpPhone").Enabled = False
   sbrStatus.Panels("Settings").Text = "Valores:"
   mnuOpen.Enabled = True
   mnucolgar.Enabled = False

   ' Apaga la luz indicadora y quita la marca al menú
   mnuOpen.Checked = False
   imgNotConnected.ZOrder

   ' Detiene el cronómetro del puerto
   StopTiming
   sbrStatus.Panels("Status").Text = "Estado:"
   On Error GoTo 0
End Sub
```

```
' Muestra el valor de la propiedad CDHolding.
Private Sub mnuHCD_Click()
   If MSComm1.CDHolding Then
      Temp = "True"
   Else
      Temp = "False"
   End If
```

```
    MsgBox "CDHolding = " + Temp
End Sub
```

```
' Muestra el valor de la propiedad CTSHolding.
Private Sub mnuHCTS_Click()
    If MSComm1.CTSHolding Then
        Temp = "True"
    Else
        Temp = "False"
    End If
    MsgBox "CTSHolding = " + Temp
End Sub
```

```
' Muestra el valor de la propiedad DSRHolding.
Private Sub mnuHDSR_Click()
    If MSComm1.DSRHolding Then
        Temp = "True"
    Else
        Temp = "False"
    End If
    MsgBox "DSRHolding = " + Temp
End Sub
```

```
' Este procedimiento establece la propiedad InputLen, que determina el número de bytes de datos leídos cada vez que se usa Input para recuperar datos del búfer de entrada. Al establecer 0 en InputLen se especifica que debe leerse todo el contenido del búfer.
Private Sub mnuInputLen_Click()
    On Error Resume Next
    Temp = InputBox$("Escriba nuevo InputLen:", "InputLen", Str$(MSComm1.InputLen))
    If Len(Temp) Then
        MSComm1.InputLen = Val(Temp)
        If Err Then MsgBox Error$, 48
    End If
End Sub
```

```
‘Muestra la Pantalla Principal de Visualización de la interfaz
Private Sub mnulstn_Click()
    Graficos.Visible = False
    Topologia.Visible = False
End Sub
```

```
' Muestra el formulario de propiedades de CommPort
Private Sub mnuProperties_Click()
  frmProps.Show vbModal

End Sub
```

```
' Alterna el estado del puerto (abierto o cerrado).
Private Sub mnuOpen_Click()
    On Error Resume Next
    Dim OpenFlag

    Temptiem.ClearData ‘Limpiamos la Pantalla de Gráficos al conectar
    tiempo = 0              ‘Inicializamos la variable del tiempo
```

```
    mnuOpen.Enabled = False    ‘deshabilitamos el botón de conectar
                                                    MSComm1.PortOpen =
                                                Not MSComm1.PortOpen ‘abrimos
                                                el puerto de Comunicaciones
    If Err Then MsgBox Error$, 48

    OpenFlag = MSComm1.PortOpen

    mnuOpen.Checked = OpenFlag

    If MSComm1.PortOpen Then
                        ' Habilita el botón de colgar y el elemento de menú
                    asociado
      mnuHangup.Enabled = True
      tbrToolBar.Buttons("HangUpPhone").Enabled = True
      tbrToolBar.Buttons("Conectar").Enabled = False
      mnuOpen.Enabled = False
      mnucolgar.Enabled = True


      imgConnected.ZOrder
      sbrStatus.Panels("Settings").Text = "Valores:" & MSComm1.Settings
      StartTiming
    Else
                                'Deshabilita el botón de colgar y el elemento de
                            menú asociado
      mnuHangup.Enabled = False
      tbrToolBar.Buttons(“HangUpPhone”).Enabled = False
      tbrToolBar.Buttons(“Conectar”).Enabled =    rae
      mnuOpen.Enabled =    rae
      mnucolgar.Enabled = False

      imgNotConnected.Zorder
      sbrStatus.Panels(“Settings”).Text = “Valores:”
      StopTiming
    End If

End Sub
```

'Este procedimiento establece la propiedad ParityReplace, que contiene el carácter que reemplazará a los caracteres incorrectos recibidos a causa de un error de paridad.

```
Private Sub mnuParRep_Click()
    On Error Resume Next

    Temp = InputBox$("Escriba el carácter que desea reemplazar", "ParityReplace",
frmTerminal.MSComm1.ParityReplace)
    frmTerminal.MSComm1.ParityReplace = Left$(Temp, 1)
    If Err Then MsgBox Error$, 48
End Sub
```

```
' Este procedimiento establece la propiedad RThreshlold, que determina el número de bytes que pueden llegar al búfer de recepción antes de disparar el evento OnComm y de que se establezca comEvReceive en la propiedad CommEvent.

Private Sub mnuRThreshold_Click()
   On Error Resume Next

   Temp = InputBox$("Introduzca el nuevo RThreshold:", "RThreshold", Str$(MSComm1.RThreshold))
   If Len(Temp) Then
      MSComm1.RThreshold = Val(Temp)
      If Err Then MsgBox Error$, 48
   End If

End Sub
```

```
‘Muestra la Pantalla correspondiente a la Topología de la Red y su Ditribución
Private Sub mnutopl_Click()
   Graficos.Visible = False
   Topologia.Visible = True
End Sub
```

```
'El evento OnComm se usa para interceptar eventos y errores de comunicaciones.

Private Static Sub MSComm1_OnComm()
   Dim EVMsg$                ‘Variable que guarda Mensajes de Eventos
   Dim ERMsg$          ‘Variable que guarda Mensajes de Error
   Dim flag As Integer ‘Bandera auxiliar
   Dim valor As String ‘Variable que guarda la temperatura
   Dim L1 As Integer   ‘Variable que guarda la longitud de la trama
   Dim salida As String          ‘Variable que almacena la trama de datos
   Dim car As String    ‘Variable que almacena carácter por carácter la trama
   Dim car1 As String  ‘Variable que almacena la trama hexadecimal completa
   Dim i As Long                 ‘Variable contador auxiliar
                        Dim temph As String    ‘Variable que almacena el valor de temperatura en hexadecimal
                              Dim nodeid As Integer       ‘Variable que almacena el número de nodo de cada sensor
                           Dim vart As String ‘Variable que almacena caracteres hexadecimales de temperatura

   ' Bifurca según la propiedad CommEvent.
   Select Case MSComm1.CommEvent
      ' Mensajes de evento.

      Case comEvReceive
         Dim Buffer As Variant
         If MSComm1.CommEvent = comEvReceive Then
'Se recibió un dato por el puerto serial
            If MSComm1.InBufferCount > 0 Then
'Se verifica si hay un dato válido en el buffer
               Buffer = MSComm1.Input
```

```
                valor = Buffer
        'Se lee el dato que hay en el buffer de entrada
                    End If
            End If
            Debug.Print "Recibir - " & StrConv(Buffer, vbUnicode)
            ShowData txtTerm, (StrConv(Buffer, vbUnicode))
            salida = txtTerm.Text    'Guarda la Trama recibida por el puerto
            L1 = Len(salida)                'Se obtiene la longitud de la Trama
                                        If L1 = 17 Then    'Se verifica como
                                trama válida con extensión de 17 caracteres
            For i = 1 To L1
                                            car = Mid(salida, i, 1)    'Se obtiene cada
                                carácter de la trama para su identificación
              HexA = Hex(Asc(car))        'Se obtiene el ascci de cada carácter de la
trama
              car1 = car1 & HexA  'Se almacena la cadena completa en ascci
              Text1.Text = car1
              If i = 9 Then
                                                nodeid = Asc(car)    'el novemo
                                carácter de la cadena representa el número identificador
                                de nodo
              End If
              If nodeid < 8 Then
                                                If i = 13 Then  'el carácter trece de la
                                cadena es parte de la generación del dato de temperatura
                 If ConvertHexToDecimal(HexA) < 10 Then
                                                        vart = "0" & HexA
                                            'si el decimal de la variable tiene un solo
                                        dígito hay que anteponer un cero
                 Else
                    vart = HexA
                 End If
               End If
               If i = 14 Then
                 temph = HexA & vart
```

En esta sección del código del programa se logra la transformación de los datos de temperatura de cada uno de los nodos a través de la regla de tres mostrada, dicha relación fue obtenida de manera experimental usando una tabla de valores típicos obtenidos durante el muestreo, estos datos se almacenan en el arreglo de la variable global *temphaux*(7) que pertenece de acuerdo al índice, a cada nodo de la Red

```
                 temphaux(nodeid) = ConvertHexToDecimal(temph)
                 temphaux(nodeid) = ((temphaux(nodeid) * 19.78) / 448)
              End If
              End If
            Next i
            End If
            If L1 = 16 Then
              For i = 1 To L1
              car = Mid(salida, i, 1)
```

```
            HexA = Hex(Asc(car))
            car1 = car1 & HexA
            Text1.Text = car1
            If i = 9 Then
               nodeid = Asc(car)
            End If
            If i = 13 Then
               temph = HexA
               temphaux(nodeid) = ConvertHexToDecimal(temph)
               temphaux(nodeid) = ((temphaux(nodeid) * 19.78) / 448)
            End If
         Next i
         End If
If nodeid <> 0 And nodeid < 8 Then   'Si el valor del nodo está en el rango de 1 a 7 se muestra la información
Text4(nodeid).Visible = Trae         'Caja de Texto que muestra el Número de Nodo
  Text4(nodeid).Text = nodeid
  Text3(nodeid).Visible = Trae       'Caja de Texto que muestra el valor de Temperatura del Nodo
  Text3(nodeid).Text = temphaux(nodeid)
  Text2(nodeid).Visible = Trae       'Caja de Texto auxiliar que muestra el valor de la longitud de cada trama
  Text2(nodeid).Text = L1
  Nodo(nodeid).Visible = True        'Pestaña de selección que permite visualizar o no las gráficas de cada nodo
  Shape1(nodeid).Visible = True      'Identificador de colores del Área de Gráficas
  Tempslide(nodeid).Visible = True   'Activex que permite la visualización de los termómetros de la Pantalla Principal
  Tempslide(nodeid).Value = temphaux(nodeid)
  Inftxt(nodeid).Visible = True      'Caja de texto que permite mostrar el valor de temperatura en la pantalla de Topología
  Inftxt(nodeid).Text = "Nodo º" & nodeid & "      " & temphaux(nodeid)
         End If
      Case comEvSend
      Case comEvCTS
         EVMsg$ = "Detectado cambio en CTS"
      Case comEvDSR
         EVMsg$ = "Detectado cambio en DSR"
      Case comEvCD
         EVMsg$ = "Detectado cambio en CD"
      Case comEvRing
         EVMsg$ = "El teléfono está sonando"
      Case comEvEOF
         EVMsg$ = "Detectado el final del archivo"

      ' Mensajes de error.
      Case comBreak
         ERMsg$ = "Parada recibida"
      Case comCDTO
```

```
            ERMsg$ = "Sobrepasado el tiempo de espera de detección de portadora"
         Case comCTSTO
            ERMsg$ = "Soprepasado el tiempo de espera de CTS"
         Case comDCB
            ERMsg$ = "Error recibiendo DCB"
         Case comDSRTO
            ERMsg$ = "Sobrepasado el tiempo de espera de DSR"
         Case comFrame
            ERMsg$ = "Error de marco"
         Case comOverrun
            ERMsg$ = "Error de sobrecarga"
         Case comRxOver
            ERMsg$ = "Desbordamiento en el búfer de recepción"
         Case comRxParity
            ERMsg$ = "Error de paridad"
         Case comTxFull
            ERMsg$ = "Búfer de transmisión lleno"
         Case Else
            ERMsg$ = "Error o evento desconocido"
      End Select

      If Len(EVMsg$) Then
         ' Muestra los mensajes de evento en la barra de estado.
         sbrStatus.Panels("Status").Text = "Estado:" & EVMsg$

         ' Activa el cronómetro para que el mensaje de la barra
         ' de estado se borre después de dos segundos.
         Timer2.Enabled = True

      ElseIf Len(ERMsg$) Then
         ' Muestra los mensajes de evento en la barra de estado.
         sbrStatus.Panels("Status").Text = "Estado:" & ERMsg$

         ' Muestra los mensajes de error en un cuadro de alerta.
         Beep
            Ret = MsgBox(ERMsg$, 1, "Haga clic en Cancelar para salir, clic en
Aceptar para ignorar.")

            ' Si el usuario hace clic en Cancelar (2)...
            If Ret = 2 Then
               MSComm1.PortOpen = False    ' Cierra el puerto y sale.
            End If

            ' Activa el cronómetro para que el mensaje de la barra
            ' de estado se borre después de dos segundos.
            Timer2.Enabled = True
         End If
      End Sub
```

' Este procedimiento establece la propiedad SThreshold, que determina el número máximo de caracteres que deben estar esperando en el búfer de salida para que se establezca comEvSend en la propiedad CommEvent y se dispare el evento OnComm.

```
Private Sub mnuSThreshold_Click()
   On Error Resume Next

   Temp = InputBox$("Escriba el nuevo valor de SThreshold", "SThreshold", Str$(MSComm1.SThreshold))
   If Len(Temp) Then
      MSComm1.SThreshold = Val(Temp)
      If Err Then MsgBox Error$, 48
   End If
End Sub
```

' Este procedimiento agrega datos a la propiedad Text del control Term. También filtra los caracteres de control, como RETROCESO, retorno de carro y avances de línea, y escribe datos en un archivo de registro. Los caracteres RETROCESO eliminan el carácter situado a su izquierda, ya sea en la propiedad Text o en la cadena pasada. Se agregan caracteres de avance de línea a todos los retornos de carro. El tamaño de la propiedad Text del control Term también se controla para que nunca exceda de MAXTERMSIZE caracteres.

```
Private Static Sub ShowData(Term As Control, Data As String)
   On Error GoTo Handler
   Const MAXTERMSIZE = 16000
   Dim TermSize As Long, i

   ' Se asegura que el texto existente no se haga demasiado largo.
   TermSize = Len(Term.Text)
   If TermSize > MAXTERMSIZE Then
      Term.Text = Mid$(Term.Text, 4097)
      TermSize = Len(Term.Text)
   End If

   ' Apunta al final de los datos de Term.
   Term.SelStart = TermSize

   ' Filtra y procesa los caracteres RETROCESO.
   Do
      i = InStr(Data, Chr$(8))
      If i Then
         If i = 1 Then
            Term.SelStart = TermSize - 1
            Term.SelLength = 1
            Data = Mid$(Data, i + 1)
         Else
            Data = Left$(Data, i - 2) & Mid$(Data, i + 1)
         End If
      End If
   Loop While i
```

```
    ' Elimina los avances de línea.
    Do
      i = InStr(Data, Chr$(10))
      If i Then
        Data = Left$(Data, i - 1) & Mid$(Data, i + 1)
      End If
    Loop While i

    ' Se asegura de que todos los retornos de carro tengan un
    ' avance de línea.
    i = 1
    Do
      i = InStr(i, Data, Chr$(13))
      If i Then
        Data = Left$(Data, i) & Chr$(10) & Mid$(Data, i + 1)
        i = i + 1
      End If
    Loop While i

    ' Agrega los datos filtrados a la propiedad SelText.
    Term.SelText = Data

    ' Registra los datos en un archivo si así se solicita.
    If hLogFile Then
      i = 2
      Do
        Err = 0
        Put hLogFile, , Data
        If Err Then
          i = MsgBox(Error$, 21)
        End If
      Loop While i <> 2
    End If
    Term.SelStart = Len(Term.Text)
  Exit Sub

  Handler:
    MsgBox Error$
    Resume Next
  End Sub
```

```
  Private Sub Timer2_Timer()
    sbrStatus.Panels("Status").Text = "Estado:"
    Timer2.Enabled = False
  End Sub
```

' Las pulsaciones interceptadas aquí se envían al control MSComm, donde se devuelven a través del evento OnComm (comEvReceive), y se muestran con el procedimiento ShowData.

```
  Private Sub txtTerm_KeyPress(KeyAscii As Integer)
```

```
    ' Si el puerto está abierto...
    If MSComm1.PortOpen Then
       ' Envía la pulsación al puerto.
       MSComm1.Output = Chr$(KeyAscii)

     'Si el eco no está activado, no hay necesidad de que el control de texto muestre la
tecla. Normalmente, el módem devolverá el carácter.
       If Not Echo Then
          ' Sitúa la posición al final del terminal
          txtTerm.SelStart = Len(txtTerm)
          KeyAscii = 0
       End If
    End If
End Sub
```

```
Private Sub tbrToolBar_ButtonClick(ByVal Button As MSComCtlLib.Button)
    Select Case Button.Key
       Case "HangUpPhone"
          Call mnuHangup_Click
       Case "Conectar"
          Call mnuOpen_Click
       Case "Propiedades"
          Call mnuProperties_Click
       Case "Listado"
          Call mnulstn_Click
       Case "Graficas"
          Call mnugraf_Click
       Case "Topología"
          Call mnutopl_Click
    End Select
End Sub
```

```
Private Sub Timer1_Timer()
    Dim i As Integer
    Dim acum(7) As Double

            tiempo = tiempo + 1          'Incrementa el valor del tiempo cada segundo
  en uno
    For i = 1 To 7
       If Nodo(i).Value = 1 Then
         Text5.Visible = True            'caja de texto que muestra la longitud de la
                                  trama
          Text6.Visible = True
                Text5.Text = temphaux(i)        'Caja de Texto que muestra el valor de
la temperatura de cada nodo
        Text6.Text = tiempo             'Caja de Texto que muestra el tiempo
transcurrido desde la conexión
          Temptiem.Plots(i).ChartXvsY tiempo, temphaux(i)
          'Gráfica de la variable temperatura en el tiempo
       End If
```

```
            If descx(i) = temphaux(i) Then
                        contdesc(i) = contdesc(i) + 1
                        If contdesc(i) > 120 Then
                           Text3(i).ToolTipText = "Sin Señal..."
                           Text3(i).BackColor = &H8080FF
                           Nodo(i).Value = 0
                           Nodo(i).Enabled = False
                        End If
                        Else
                           contdesc(i) = 0
                           Nodo(i).Enabled = True
                           Text3(i).BackColor = &H80000005
                           Text3(i).ToolTipText = "En Línea..."
            End If
            descx(i) = temphaux(i)
         Next i

         ' Muestra la hora de conexión
         sbrStatus.Panels("ConnectTime").Text = Format(Now - StartTime, "hh:nn:ss") & "
"
         txtTerm.Text = ""
      End Sub
```

```
      'Llama a esta función para iniciar el cronómetro ConnectTime
      Private Sub StartTiming()
         StartTime = Now
         Timer1.Enabled = True
      End Sub
```

```
      'Llama a esta función para detener el cronometraje
      Private Sub StopTiming()
         Timer1.Enabled = False
         sbrStatus.Panels("ConnectTime").Text = ""
      End Sub
```

ANEXO 2

CÓDIGO DE PROGRAMACION DE LOS SENSORES Y LA ESTACION BASE

- **PROGRAMACIÓN DE LA RED PARA SENSORES XMTS310**

- **XMTS310M.NC**

```
#include "appFeatures.h"
//includes XCommand;
includes sensorboard;
//definicion y llamada al modulo xmts310M
module XMTS310M {
   provides {
      interface StdControl;
   }
   uses {
// setea funciones de networking tipo malla para trabajo en RF
      interface MhopSend as Send; //interface de envio de tramas
      interface RouteControl; //interface de control y ruta de las tramas

      interface XCommand; //interfaz de comandos
      interface XEEControl; //interfaz de control de la red tipo malla
      // interface ReceiveMsg as Bcast; //interfaz de recepcion de tramas como broadcast
// Batteria
   interface ADC as ADCBATT;  //definicion de la interface para baterias
   interface StdControl as BattControl; // control de baterias
//Temp
      interface StdControl as TempControl;    //definicion interfaz de control de
temperatura
      interface ADC as Temperature; //interfaz de recepcion Analoga de temperatura
//Light
      interface StdControl as PhotoControl;   //definicion de la interfaz de control de luz
      interface ADC as Light; //adquiere datos del ADC para la luz
// Mic
      interface StdControl as MicControl; //definicion function microfono
      interface Mic; //interface para microfono
      interface ADC as MicADC; // obtiene la informacion del ADC para el microfono
// Sounder
      interface StdControl as Sounder; //function de control para sonido
// Accel
      interface StdControl as AccelControl; //function de control de la aceleracion
```

```
        interface ADC as AccelX;
        interface ADC as AccelY;
// Mag
        interface StdControl as MagControl;
        interface ADC as MagX;
        interface ADC as MagY;

        //interface ADCControl;
        interface Timer;
        interface Leds;

#if (FEATURE_UART_SEND | TOS_LOCAL_ADDRESS==0) //
#if FEATURE_UART_SEND
        interface SendMsg as SendUART; //envia por el uart el paquete en casi de tener el
permiso
#endif
        command result_t PowerMgrEnable(); //llama a function de activacion de
potenciapara transimision de mensajes
        command result_t PowerMgrDisable();//
#endif
  command void health_packet(bool enable, uint16_t intv); //2 bytes de dentro de la
cabecera informacion del paquete
    }
}
implementation {
    enum { START, BUSY, SOUND_DONE}; //inicio del sensor, prendido, ocupado, y
sonido de inicio una vez listo

/* should debe ser movido a xmesh/bin una vez realizado a cambio de   ---sfmao
    enum {
    MODE_UPSTREAM,
    MODE_UPSTREAM_ACK,
    MODE_DOWNSTREAM,
    MODE_DOWNSTREAM_ACK,
    MODE_ANY2ANY,
    MODE_ONE_HOP_BROADCAST
  };
  enum {
    BASE_STATION_ADDRESS       = 0, //reconoce la estacion base como 0
  };
*/
    #define MSG_LEN  29
    TOS_Msg    gMsgBuffer; //longitud del mensaje como 29 envia al buffer
    TOS_Msg    msg_buf_radio; //revisa el la frecuencia con la que se envia el mensaje por
el buffer
    TOS_MsgPtr msg_radio; //frecuencia de transmission de mensajes
    bool       sleeping;          // application command state
    norace XDataMsg   readings;
    char main_state;
    norace bool sound_state, sending_packet,sensinginsession;
```

```
   norace uint8_t        miccnt;
   norace uint32_t  val;

/*****************************************************************************
 * Task to xmit radio message
 *
 *    msg_radio->addr = TOS_BCAST_ADDR;
 *    msg_radio->type = 0x31;
 *    msg_radio->length = MSG_LEN;
 *    msg_radio->group = TOS_AM_GROUP;

*****************************************************************************/
   task void send_radio_msg() {
       uint8_t   i;
       uint16_t  len;
       XDataMsg *data;

       call Leds.yellowOn();
       // Fill the given data buffer.
       data = (XDataMsg*)call Send.getBuffer(msg_radio, &len);

       for (i = 0; i <= sizeof(XDataMsg)-1; i++)
          ((uint8_t*)data)[i] = ((uint8_t*)&readings)[i];

       data->board_id  = SENSOR_BOARD_ID;
       data->packet_id = 1;
       //data->node_id   = TOS_LOCAL_ADDRESS;
       data->parent    = call RouteControl.getParent();
       data->packet_id = data->packet_id | 0x80;
#if FEATURE_UART_SEND
       if (TOS_LOCAL_ADDRESS != 0) {
          call PowerMgrDisable();
          TOSH_uwait(1000);
          if (call SendUART.send(TOS_UART_ADDR, sizeof(XDataMsg),
                                 msg_radio) != SUCCESS)
          {
         atomic sending_packet = FALSE;
         call Leds.yellowOff();
         call PowerMgrEnable();
          }
       }
       else
#endif
       {
          // Send the RF packet!
          if (call
Send.send(BASE_STATION_ADDRESS,MODE_UPSTREAM,msg_radio,
sizeof(XDataMsg)) != SUCCESS) {
```

```
          atomic sending_packet = FALSE;
          call Leds.yellowOff();
           }
        }
       return;
}
    static void initialize() {
    atomic {
         sleeping = FALSE;
         main_state = START;
         sound_state = TRUE;
         sending_packet = FALSE;
#ifdef APP_RATE
          timer_rate = XSENSOR_SAMPLE_RATE;
#else
#ifdef USE_LOW_POWER
          timer_rate = XSENSOR_SAMPLE_RATE  + ((TOS_LOCAL_ADDRESS%255)
<< 7);
#else
         timer_rate = XSENSOR_SAMPLE_RATE + ((TOS_LOCAL_ADDRESS%255) <<
2);
#endif
#endif
         miccnt=0;
         val=0;
         sensinginsession=FALSE;
     }
  }
   static void start()
    {
     call BattControl.start();
     call Mic.gainAdjust(64);  // Set the gain of the microphone. (refer to Mic)
     call MicControl.start();
     call TempControl.start();
     call PhotoControl.start();
#ifdef MTS310
     call AccelControl.start();
     call MagControl.start();
#endif
    }
    task void battstop()
    {
        call BattControl.stop();
    }
    task void tempstop()
    {
        call TempControl.stop();
    }
    task void photostop()
    {
```

```
        call PhotoControl.stop();
    }
    task void Micstop()
    {
        call MicControl.stop();
    }
    task void Accelstop()
    {
        call AccelControl.stop();
    }

/*****************************************************************************
 * Initialize the component. Initialize ADCControl, Leds
 *
 *****************************************************************************/
    command result_t StdControl.init() {

        atomic msg_radio = &msg_buf_radio;

    //  MAKE_BAT_MONITOR_OUTPUT();  // enable voltage ref power pin as output
    //  MAKE_ADC_INPUT();           // enable ADC7 as input
    call BattControl.init();
// usart1 is also connected to external serial flash
// set usart1 lines to correct state
//  TOSH_MAKE_FLASH_SELECT_OUTPUT();
      TOSH_MAKE_FLASH_OUT_OUTPUT();           //tx output
      TOSH_MAKE_FLASH_CLK_OUTPUT();           //usart clk
//  TOSH_SET_FLASH_SELECT_PIN();
// Llamada de funciones para control de las variables del sensor
      call Leds.init();
      call TempControl.init();
      call PhotoControl.init();
      call MicControl.init();
      call Mic.muxSel(1);  // Set the mux so that raw microhpone output is selected
      call Mic.gainAdjust(64);  // Set the gain of the microphone. (refer to Mic)
#if FEATURE_SOUNDER
      call Sounder.init();
#endif

#ifdef MTS310
      call AccelControl.init();
      call MagControl.init();
#endif

#if (TOS_LOCAL_ADDRESS==0)
call PowerMgrDisable();
#endif
```

```
    initialize();
    return SUCCESS;
}
/*****************************************************************************
 * Start the component. Start the clock.
 *
*****************************************************************************/
  command result_t StdControl.start()
  {
        call StdControl.stop();
        call health_packet(TRUE,TOS_HEALTH_UPDATE);
    call Timer.start(TIMER_REPEAT, timer_rate);
    return SUCCESS;
  }
/*****************************************************************************
 * Stop the component.
 *

*****************************************************************************/
  command result_t StdControl.stop() {
    call BattControl.stop();
    call TempControl.stop();
    call PhotoControl.stop();
    call MicControl.stop();
#ifdef MTS310
    call AccelControl.stop();
    call MagControl.stop();
#endif
    return SUCCESS;
  }
/*****************************************************************************
 * Measure Temp, Light, Mic, toggle sounder
 *
*****************************************************************************/
  event result_t Timer.fired() {
    if (sending_packet)
      return SUCCESS;           //don't overrun buffers
    if(main_state==BUSY)
      return SUCCESS;
    start();
    atomic main_state = BUSY;
    if (!sensinginsession){
      call ADCBATT.getData();
      atomic sensinginsession = TRUE;
```

```
      }
    return SUCCESS;
  }
/*****************************************************************************
 * Battery Ref  or thermistor data ready
*****************************************************************************
**/
  async event result_t ADCBATT.dataReady(uint16_t data) {
    if (!sensinginsession) return FAIL;
    readings.vref = 0;
        //data;
    atomic sensinginsession = FALSE;
    post battstop();
    //call TempControl.start();
    call Temperature.getData();
    return SUCCESS;
  }



/*****************************************************************************
 * Temperature ADC data ready
 * Read and get next channel.
*****************************************************************************
*/
  async event result_t Temperature.dataReady(uint16_t data) {
    readings.thermistor = data;
    //call PhotoControl.start();
    call Light.getData();
    return SUCCESS;
  }
/*****************************************************************************
 * Photocell ADC data ready
 * Read and get next channel.
*****************************************************************************
*/
  async event result_t Light.dataReady(uint16_t data) {
    readings.light =0;
        //data;
    post photostop();
    post tempstop();
    call MicADC.getData();
    return SUCCESS;
  }

/*****************************************************************************
```

```
 * MicroPhone ADC data ready
 * Read and toggle sounder.
 * send uart packet
**************************************************************************
*/
  async event result_t MicADC.dataReady(uint16_t data) {
    if(miccnt<50)
    {
          atomic miccnt=miccnt+1;
          TOSH_uwait(1000);
          if(val<data)
          {
                atomic val=0;
                //data;
          }
          call MicADC.getData();
          return SUCCESS;
    }
    else
    {
          atomic miccnt=0;
    }
    readings.mic = val;//data;
    post Micstop();
    val=0;
#ifdef MTS310
    call AccelX.getData();
#else
    // This is the final sensor reading for the MTS300...
    atomic {
        if (!sending_packet) {
          sending_packet = TRUE;
          post send_radio_msg();
        }
    }
#if FEATURE_SOUNDER
    if (sound_state) call Sounder.start();
    else call Sounder.stop();
    atomic {
        sound_state = SOUND_STATE_CHANGE;
    }
#endif
#endif
    return SUCCESS;
 }
/**************************************************************************
***
 *  ADC data ready
 * Read and toggle sounder.
 * send uart packet
```

```
**********************************************************************
**/
  async event result_t AccelX.dataReady(uint16_t data) {
     readings.accelX = 0;
         //data;

     call AccelY.getData();
     return SUCCESS;
  }
/**********************************************************************
***
 *  ADC data ready
 * Read and toggle sounder.
 * send uart packet

**********************************************************************
**/
  async event result_t AccelY.dataReady(uint16_t data) {
     readings.accelY = 0;
         //data;
     post Accelstop();
     call MagX.getData();
     return SUCCESS;
  }

 /**
  * In response to the <code>MagX.dataReady</code> event, it stores the
  * sample and issues command to sample the magnetometer's Y axis.
  * (Magnetometer B pin)
  * @return returns <code>SUCCESS</code>
  */
  async event result_t MagX.dataReady(uint16_t data){
     readings.magX = 0;
         //data;

     call  MagY.getData(); //get data for MagnetometerB
     return SUCCESS;
  }
/**
  * In response to the <code>MagY.dataReady</code> event, it stores the
  * sample and issues a task to filter and process the stored magnetometer
  * data  * It also has a schedule which starts sampling the Temperture and
  * Accelormeter depending on the stepdown counter.
  * @return returns <code>SUCCESS</code>
  */
  async event result_t MagY.dataReady(uint16_t data){
     readings.magY = 0;
         //data;
     atomic {
         if (!sending_packet) {
```

```
            sending_packet = TRUE;
            post send_radio_msg();
        }
    }
#if FEATURE_SOUNDER
    if (sound_state) call Sounder.start();
    else call Sounder.stop();
    atomic {
        sound_state = SOUND_STATE_CHANGE;
    }
#endif
    return SUCCESS;
  }
/**
  * Handles all broadcast command messages sent over network.
  * NOTE: Bcast messages will not be received if seq_no is not properly
  *       set in first two bytes of data payload.  Also, payload is
  *       the remaining data after the required seq_no.
  event result_t XCommand.received(XCommandOp *opcode) {

    switch (opcode->cmd) {
        case XCOMMAND_SET_RATE:
            // Change the data collection rate.
            timer_rate = opcode->param.newrate;
            call Timer.stop();
            call Timer.start(TIMER_REPEAT, timer_rate);
            break;
        case XCOMMAND_SLEEP:
            // Stop collecting data, and go to sleep.
            sleeping = TRUE;
            call Timer.stop();
            call StdControl.stop();
            call Leds.set(0);
           break;
        case XCOMMAND_WAKEUP:
            // Wake up from sleep state.
            if (sleeping) {
           initialize();
           call Timer.start(TIMER_REPEAT, timer_rate);
           call StdControl.start();
           sleeping = FALSE;
            }
            break;

        case XCOMMAND_RESET:
            // Reset the mote now.
            break;
        case XCOMMAND_ACTUATE: {
            uint16_t state = opcode->param.actuate.state;
            if (opcode->param.actuate.device != XCMD_DEVICE_SOUNDER) break;
```

```
            // Play the sounder for one period.
            sound_state = state;
            if (sound_state) call Sounder.start();
            else call Sounder.stop();
            atomic {
           sound_state = SOUND_STATE_CHANGE;
            }
            break;
        }
        default:
            break;
    }
    return SUCCESS;
  }
#if FEATURE_UART_SEND
 /**
  * Handle completion of sent UART packet.
  *
  event result_t SendUART.sendDone(TOS_MsgPtr msg, result_t success)
  {
    //      if (msg->addr == TOS_UART_ADDR) {
    atomic msg_radio = msg;
    msg_radio->addr = TOS_BCAST_ADDR;
    if (call Send.send(BASE_STATION_ADDRESS,MODE_UPSTREAM,msg_radio,
sizeof(XDataMsg)) != SUCCESS) {
        atomic sending_packet = FALSE;
        call Leds.yellowOff();
    }
    if (TOS_LOCAL_ADDRESS != 0) // never turn on power mgr for base
        call PowerMgrEnable();
    //}
    return SUCCESS;
  }
#endif
/**
  * Handle completion of sent RF packet.
  *
  event result_t Send.sendDone(TOS_MsgPtr msg, result_t success)
  {
    atomic {
        msg_radio = msg;
        main_state = START;
        sending_packet = FALSE;
        call StdControl.stop();
    }
    call Leds.yellowOff();
#if FEATURE_UART_SEND
    if (TOS_LOCAL_ADDRESS != 0) // never turn on power mgr for base
        call PowerMgrEnable();
#endif
```

```
    return SUCCESS;
  }
  event result_t XEEControl.restoreDone(result_t result)
  {
        if(result) {
                        call Timer.stop();
                        call Timer.start(TIMER_REPEAT, timer_rate);
        }
       return SUCCESS;
  }
}
```

- **PROGRAMACIÓN DE LA RED PARA LA ESTACIÓN BASE**

- **XMTS310.NC**

```
#include "appFeatures.h"

includes sensorboardApp;

configuration XMTS310 {
// this module does not provide any interface
}
implementation
{
   components Main,
            TimerC,
                GenericCommPromiscuous as Comm,
                MULTIHOPROUTER,XMTS310M, QueuedSend,
                Voltage, MicC, PhotoTemp, Accel, Mag, Sounder,
                XCommandC,
#if (FEATURE_UART_SEND | TOS_LOCAL_ADDRESS==0)
       HPLPowerManagementM,
#endif
       LEDS_COMPONENT
       Bcast;
   Main.StdControl -> XMTS310M;
   Main.StdControl -> QueuedSend.StdControl;
   Main.StdControl -> MULTIHOPROUTER.StdControl;
   Main.StdControl -> Comm;
   Main.StdControl -> TimerC;
   LEDS_WIRING(XMTS310M)
#if (FEATURE_UART_SEND | TOS_LOCAL_ADDRESS==0)
   // Wiring for UART msg.
   XMTS310M.PowerMgrDisable -> HPLPowerManagementM.Disable;
   XMTS310M.PowerMgrEnable -> HPLPowerManagementM.Enable;
#if FEATURE_UART_SEND
   XMTS310M.SendUART -> QueuedSend.SendMsg[AM_XDEBUG_MSG];
```

```
#endif
#endif
   XMTS310M.Timer -> TimerC.Timer[unique("Timer")];
   // Wiring for Battery Ref
//llamada a funciones de control de variables sensoriales
   XMTS310M.BattControl -> Voltage;
   XMTS310M.ADCBATT -> Voltage;
   XMTS310M.TempControl -> PhotoTemp.TempStdControl;
   XMTS310M.Temperature -> PhotoTemp.ExternalTempADC;
   XMTS310M.PhotoControl -> PhotoTemp.PhotoStdControl;
   XMTS310M.Light -> PhotoTemp.ExternalPhotoADC;
   XMTS310M.Sounder -> Sounder;
   XMTS310M.MicControl -> MicC;
   XMTS310M.MicADC -> MicC;
   XMTS310M.Mic -> MicC;
   XMTS310M.AccelControl -> Accel;
   XMTS310M.AccelX -> Accel.AccelX;
   XMTS310M.AccelY -> Accel.AccelY;
   XMTS310M.MagControl-> Mag;
   XMTS310M.MagX -> Mag.MagX;
   XMTS310M.MagY -> Mag.MagY;
   XMTS310M.XCommand -> XCommandC;
   XMTS310M.XEEControl -> XCommandC;

   // uniendo parametros RF para trabajo de la red malla Wiring for RF mesh networking.
   XMTS310M.RouteControl -> MULTIHOPROUTER;
   XMTS310M.Send -> MULTIHOPROUTER.MhopSend[AM_XMULTIHOP_MSG];
   MULTIHOPROUTER.ReceiveMsg[AM_XMULTIHOP_MSG] -
>Comm.ReceiveMsg[AM_XMULTIHOP_MSG];
   XMTS310M.health_packet -> MULTIHOPROUTER;
}
```

- **XMESHBASEM.NC**

```
#include "appFeatures.h"
module XMeshBaseM {
   provides {
  interface StdControl;
   }
   uses {
  interface RouteControl;
  interface XCommand;
  interface Leds;
   }
}
implementation {


  command result_t StdControl.init() {
```

```
    call Leds.init();
    //this is to make sure that
    //no matter how the application is loaded (jtag,etc)
    //the local address of the base station is 0
    atomic{ TOS_LOCAL_ADDRESS = 0; }
     return SUCCESS;
  }
  command result_t StdControl.start(){
     return SUCCESS;
  }
  command result_t StdControl.stop() {
     return SUCCESS;
  }
  event result_t XCommand.received(XCommandOp *opcode) {
     switch (opcode->cmd) {
    case XCOMMAND_SET_RATE:
       break;
    case XCOMMAND_SLEEP:
          break;
    case XCOMMAND_WAKEUP:
       break;
    case XCOMMAND_RESET:
       break;
    default:
       break;
     }
     return SUCCESS;
  }
}
```

- **XMESHBASE.NC**

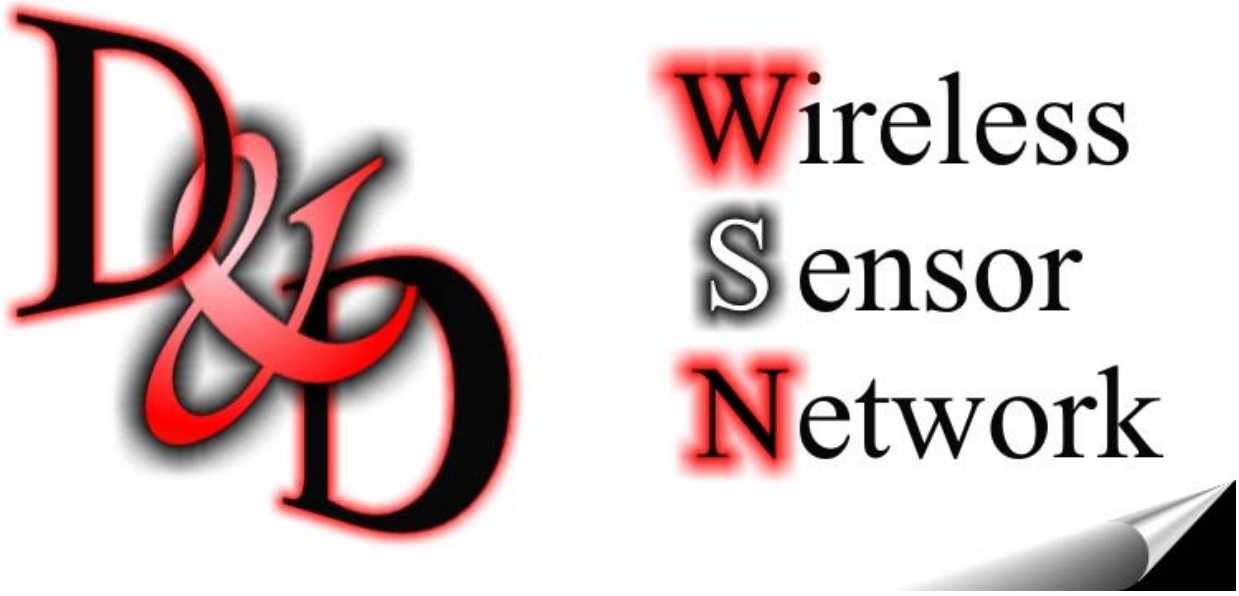
```
#include "appFeatures.h"

includes sensorboardApp;
/**
 * XMeshBase is the gateway firmware for all XMesh suite applications.
 */
configuration XMeshBase {
// this module does not provide any interface
}
implementation
{
  Components Main,
  MULTIHOPROUTER,XMeshBaseM,
  JOIN_COMPONENT
  LEDS_COMPONENT
  HEARTBEAT_COMPONENT
  XCommandC;
```

```
    Main.StdControl -> XMeshBaseM;
    JOIN_WIRING()
    HEARTBEAT_WIRING()
    LEDS_WIRING(XMeshBaseM)
    XMeshBaseM.XCommand -> XCommandC;
    // Wiring for RF mesh networking.
    XMeshBaseM.RouteControl -> MULTIHOPROUTER;
}
```

ANEXO 3

MANUAL DE USUARIO



INFORMACIÓN TÉCNICA

D&D Asociates INC.

El presente documento es una explicación de cada una de las características y formas de operación de la Red Sensorial Inalámbrica y de la Interfaz de Control y Monitoreo de la Variable temperatura, generada para el proyecto:

"Estudio, Diseño y Aplicación de las Redes Sensoriales Inalámbricas para el Monitoreo y Control de Temperatura en un Área de Comunicaciones Simulada"

Proyecto realizado para la Escuela Politécnica del Ejército

Campus Sangolquí

Martes 1 de Mayo del 2007

INTRODUCCIÓN

Gracias por trabajar con la Interfaz de Control y Monitoreo de Temperatura D&D

ANTES DE USAR LA INTERFAZ

1. Lea detenidamente este instructivo

   Contiene información sobre la operación y mantenimiento de la Interfaz. Cuando haya terminado de leerlo, asegúrese de guardarlo para futuras soluciones y consultas.

2. Modelo y Número de Serie

   Por favor, en cualquier tipo de correspondencia respecto a la interfaz en cuestión, indique su número de serie otorgado a continuación.

   MODELO:

   _____

   NUMERO DE SERIE:

PRECAUCIONES DE SEGURIDAD

Para el correcto funcionamiento de la Red Sensorial y su sistema de Monitoreo y Control, siga las siguientes recomendaciones:

1. Asegure un lugar estable para cada uno de los nodos Sensoriales del Sistema

2. No utilice el equipo en caso de estar dañado o que no funcione correctamente, apague los nodos y llame a servicio al cliente.

3. Siga correctamente las instrucciones de instalación, tanto de la Red como de la Interfaz de manejo de los Equipos.

4. Asegure las Instalaciones eléctricas tanto de los efectores finales (ventiladores), como de la PC que va a funcionar como estación Base para evitar la acción adversa de caídas de Tensión.

CARACTERISTICAS DE LAS PARTES

A continuación se muestra un listado de las partes que constituyen el KIT para armar la Red Sensorial Inalámbrica, tanto los elementos del Hardware como los del Software:

1. Tablero Receptor Base y de Grabación MIB510 (x1)
2. Módulos de Transmisión/Recepción MPR2400(x5)
3. Tableros Sensores MTS310CB (x5)
4. Baterías Industriales de 3000 horas de operación (x10)
5. Efectores Finales (Ventiladores x 5)
6. Tablero de Control de Salidas de Efectores Finales (x1)
7. Software de Monitoreo y Control D&D

TABLERO RECEPTOR BASE Y DE GRABACIÓN MIB510

Para la instalación de este equipo siga lo siguientes pasos:

- Conecte el cable serial tanto a la PC como al MIB510
- Conecte el Adaptador del equipo para energizarlo
- Para la Programación ubique el selector en ON
- Para Transmisión y Recepción de Datos ubique el selector en OFF
- Para la grabación de un MOTE utilice el conector de 50 pines de la placa
- Para hacer de Estación Base conecte el MOTE numerado como Base o Nodo 0

Módulos Tx/Rx MPR2400 y Sensores MTS310CB

Para la instalación de este equipo siga lo siguientes pasos:

- Adjunte a cada uno de los Transmisores las baterías para su operación.

- Adjunte a cada uno de los tableros de transmisión un sensor a través del conector especial de 50 pines.

- Para encendido de los Nodos de la Red, ubique el selector adjunto en la posición de encendido para su operación.

- Programar los tableros accediendo al programa incluido en el CD de instalación, para la estación Base, con el archivo "*Xmeshbase.nc*" mientras que para los nodos externos con el archivo "*xmts310cb.nc*"

- Ubicar los sensores en lugares estratégicos de selección para poder recibir los datos de temperatura de los mismos.

Interfaz de Monitoreo y Control de Temperatura D&D

Para la instalación del programa de Control de la Red Sensorial a ser implementada, Ejecute el archivo de Instalación en le CD para copiar la carpeta del Software en la dirección:

"***C:\Program Files\Crossbow\D&D***"

Asegúrese que las imágenes necesarias para la ejecución del programa no sean alteradas o accedidas por gente ajena a la operación de los equipos y del Software mismo.

OPERACIÒN DEL SOFTWARE

Bienvenido a la parte operativa de la Red Sensorial, en esta sección usted será guiado a través del uso de las bondades que ofrece la Interfaz de Control y Monitoreo D&D.

En primer lugar lo que se va a desarrollar es acerca del uso de las pantallas que muestra la interfaz, llevando a cabo un detalle acerca de sus principales funciones y como operarlas.

Pantalla Principal de Visualización

Es la primera pantalla de visualización que se muestra en la interfaz de monitoreo y control, es aquella en la que tenemos la posibilidad de observar detalladamente numérica y gráficamente cada una de las mediciones de los sensores conectados a la Red.

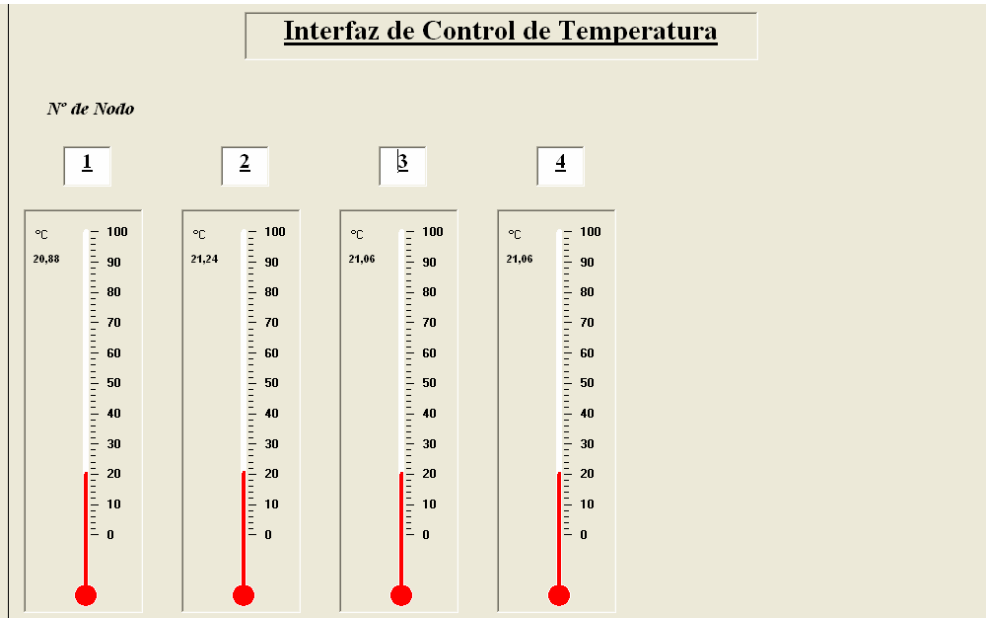


Figura 4: Pantalla Principal de Visualización

Como se muestra en la Figura, la medición de la Temperatura es desplegada en valor numérico y en forma visual, a manera de un termómetro gráfico, en el cual podemos apreciar por nivel el valor mostrado, el especio de la pantalla nos es suficiente para poder tener al mismo tiempo un máximo de 7 sensores en el sistema.

Pantalla de Historial Gráfico de Datos en el Tiempo

Permite acceder a una gráfica en la que se muestren los datos de temperatura medidos por cada sensor a lo largo del tiempo de conexión, con la posibilidad de habilitar o no la grafica individual de los mismos.

En general, la pantalla mostrada para esta opción se describe por si sola, muestra el área de gráfico y a un lado, de acuerdo a su activación, a las pestañas de selección de cada uno de los nodos posibles con el identificador del color que lo determina.

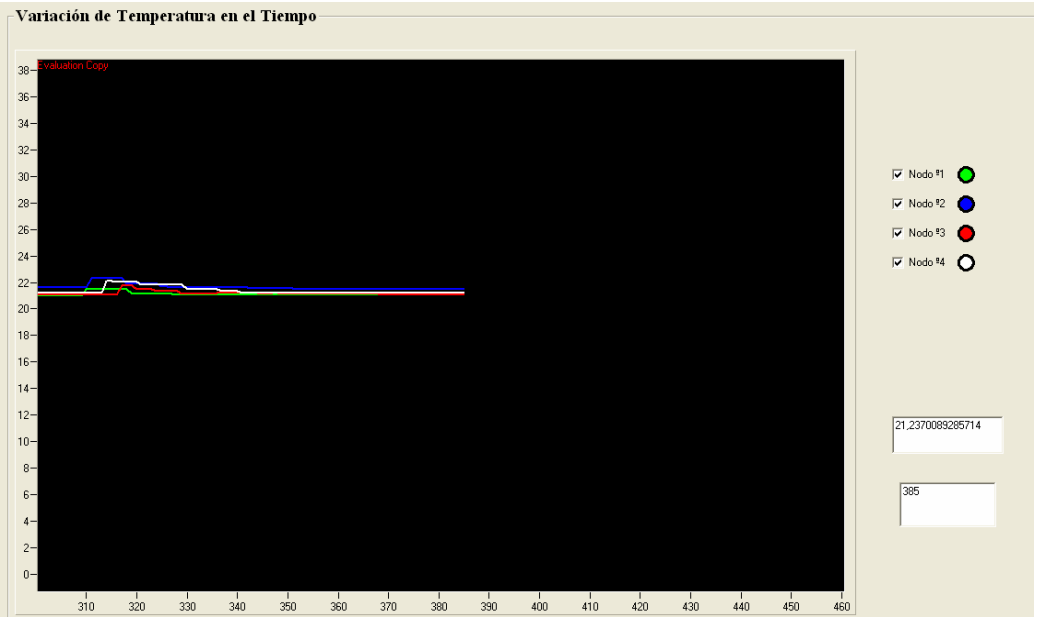


Figura 5: Pantalla de Gráfica en el Tiempo

Pantalla de Visualización de Topología de la Red

Muestra es la forma en la que los sensores se encuentran distribuidos a lo largo del área de trabajo, que para el caso, representa el área industrial de comunicaciones, en donde se va a controlar la temperatura parámetro importante para poder asegurar el correcto funcionamiento y desempeño de los equipos inmersos en su generación.

Por lo tanto, esta parte de la visualización, dentro de la interfaz nos permite ver la forma física en que los sensores han sido distribuidos en dicha área, de igual forma que en las otras opciones de visualización, en este caso, el aparecimiento de un nuevo sensor en la red se verifica a través de imágenes de extensión GIF que representan a los nodos.

También tenemos la posibilidad de acceder al valor numérico de la medida de temperatura de cada uno de los sensores activos junto con su identificador en el lado izquierdo de la interfaz.

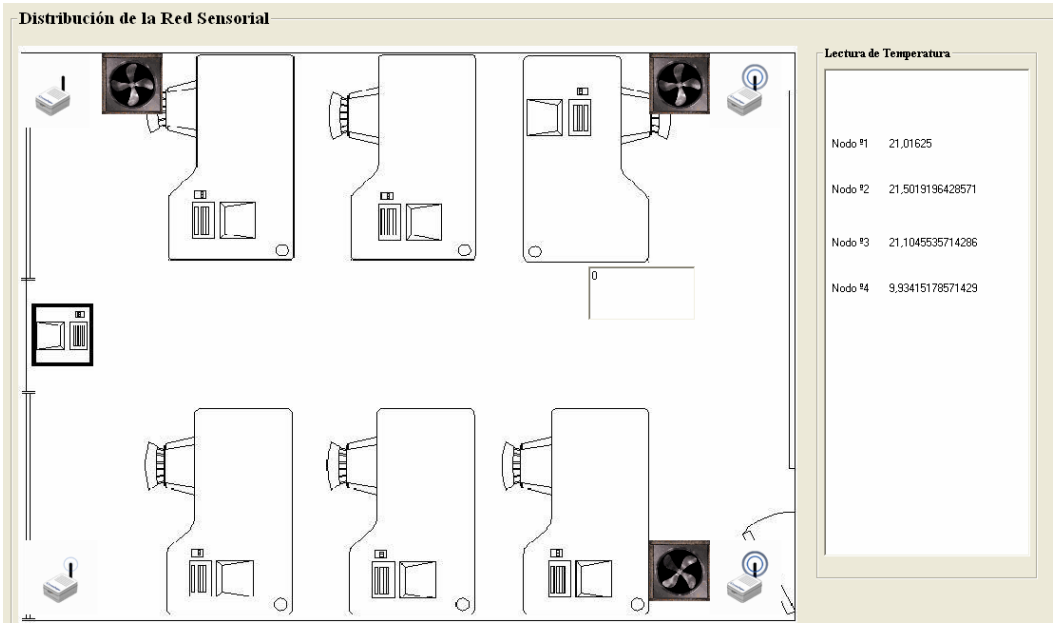


Figura 6: Pantalla de Visualización de la Topología de la Red

MANEJO DE LA INTERFAZ

Para facilitar el uso de la interfaz de monitoreo y control es necesario que nos familiaricemos con la parte operativa, es decir con cada uno de los controles disponibles en su presentación, de tal forma, que esta sección del manual estará orientada a la explicación de las bondades del programa:

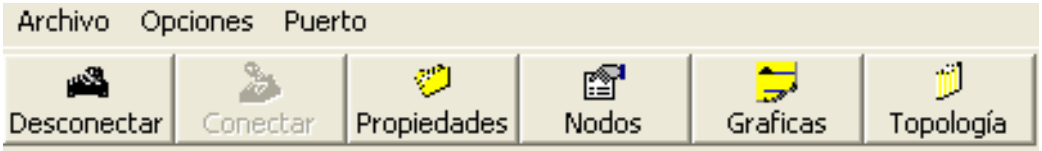


Figura 7: Barra de Herramientas de la Interfaz

Desconectar; Botón de Acceso directo a la opción de desconexión del puerto de comunicaciones escogido en los ajustes o ventana de propiedades, es decir, internamente,

en el programa lo que se realiza a través de esta opción es cerrar el mismo para evitar el flujo de información.

Conectar; Botón de acción contraria al caso anterior, es decir, el acceso directo que nos permite abrir el puerto de comunicaciones escogido en las propiedades, hay que tener en cuenta que los dos botones, tanto el de conexión como de desconexión, complementariamente se activan y desactivan cuando uno de los dos es presionado, esto para evitar la apertura del puerto una y otra vez, o el caso contrario, lo que podría generar algunos errores.

Propiedades; el acceso directo a la ventana de Propiedades del puerto de Comunicaciones, es decir, esta opción nos permite el acceso a la ventana en la cual podemos realizar los cambios necesarios en el puerto de comunicaciones, datos tales como:

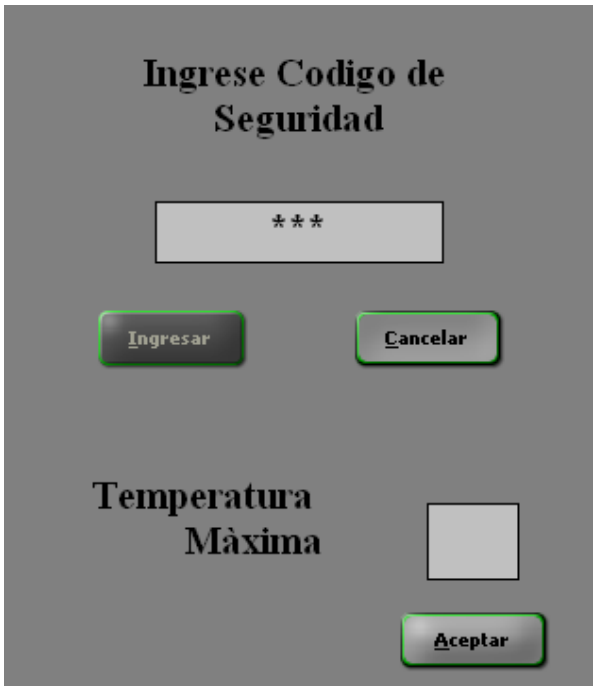
Escoger el Puerto (COM);
Velocidad de Transmisión de Datos;
Bits de la trama;
Paridad;
Entre otras…

Nodos; opción que nos muestra la pantalla Principal de Visualización, explicada anteriormente.

Gráficas; lo que nos permite acceder a la pantalla de visualización de las gráficas del dato Temperatura en función del tiempo, igualmente explicada en la sección anterior.

Topología; acceso a la pantalla de visualización de la distribución de la Red en el área especificada.

TempMAX; que permite a un usuario con clave de acceso, ajustar el valor máximo de temperatura permitido para el área de comunicaciones, este valor debera estar entre los 10 y 30 grados centígrados.



ALERTAS DE LA INTERFAZ

En la estructura de programación de la Interfaz, es posible encontrar ciertas señales a través de las cuales nos podemos dar alerta de ciertos eventos, el más común y claro en cualquiera de las pantallas es l desconexión de alguno de los nodos de la red, es decir, en cualquiera de las opciones de visualización, podemos tener señales a través de las cuales darnos cuenta que un nodo de la Red Sensorial ha salido de operación y necesita revisión o mantenimiento, esta señales son las siguientes:

En la Pantalla Pricipal, cuando un Nodo deja de transmitir Datos a la *Estación Base*, entonces cambia la tonalidad del termómetro de la interfaz a color rojo y un texto de ayuda nos indica que esta fuera.

En la Pantalla de Gráficas del Temperatura vs. Tiempo, simplemente cuando deja de transmitir, la grafica decae y no se grafica más en el tiempo de conexión.

Finalmente en la pantalla de topología, las animaciones de los Sensores se detienen y dejan de transmitir la información.

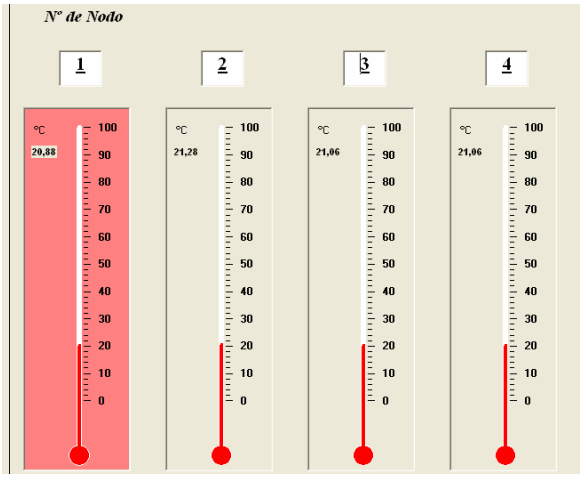


Figura 8: Alerta Gráfica de que el nodo ha salido de operación

REFERENCIAS BIBLIOGRAFICAS

[1] LEWIS, F. L., "*Wireless Sensor Net*", Chapter 4; Associate Director for Research Head, Advanced Controls, Sensors, and MEMS Group; Automation and Robotics Research Institute; University of Texas at Arlington.

[2] MOHAMMAD, Ilyas y Imad, Mahgoub, *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*, Clearence Center, 2005

[3] Xbow Technologies, *TinyOs Starting Guid*, 2004

[4] JENS, Mache; ALLICK, Chris, y Charnas, *Sensor Network Lab Exercises Using TinyOS and MicaZ Motes,*Mathematical and Computer Science; Lewis & Clark College; Portland OR, USA; jmache@lclark.edu

[5] CHIEN-LIANG Fok, *TinyOs Tutorial* (PDF);; CSE467s Spring 2007

[6] LEWIS, F. L, Wireless Sensor Networks, *Smart Environments: Technologies, Protocols, and Applications,* D.J. Cook and S.K. Das, John Wiley, New York, 2004. Web: http://arri.uta.edu/acs/networks/WirelessSensorNetChap04.pdf

[7] Grupo de Sistemas de Información y Comunicación, *GSIC*. en el Instituto de Robótica. Web: *http://147.156.223.19/home.php*

[8] BHARATHIDASAN, Archana, y SAI PONDURU, Vijay Anand, *Sensor Networks: An Overview,*
Web: http://wwwcsif.cs.ucdavis.edu/~bharathi/sensor/survey.pdf

[9] WEISS, Mark, *TINYOS. An Operative System for Wireless Sensor Networks*, University of Nebraska, Lincoln,
Web: http://csce.unl.edu/~witty/f2004/csce489/WSN-20031201.pdf , Diciembre del 2003

[10] Ganesan, Deepak, y Cerpa, Alberto, *Networking Issues in Wireless Sensor Network,* Web: http://www.isi.edu/~weiye/pub/jpdc.pdf

INDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS