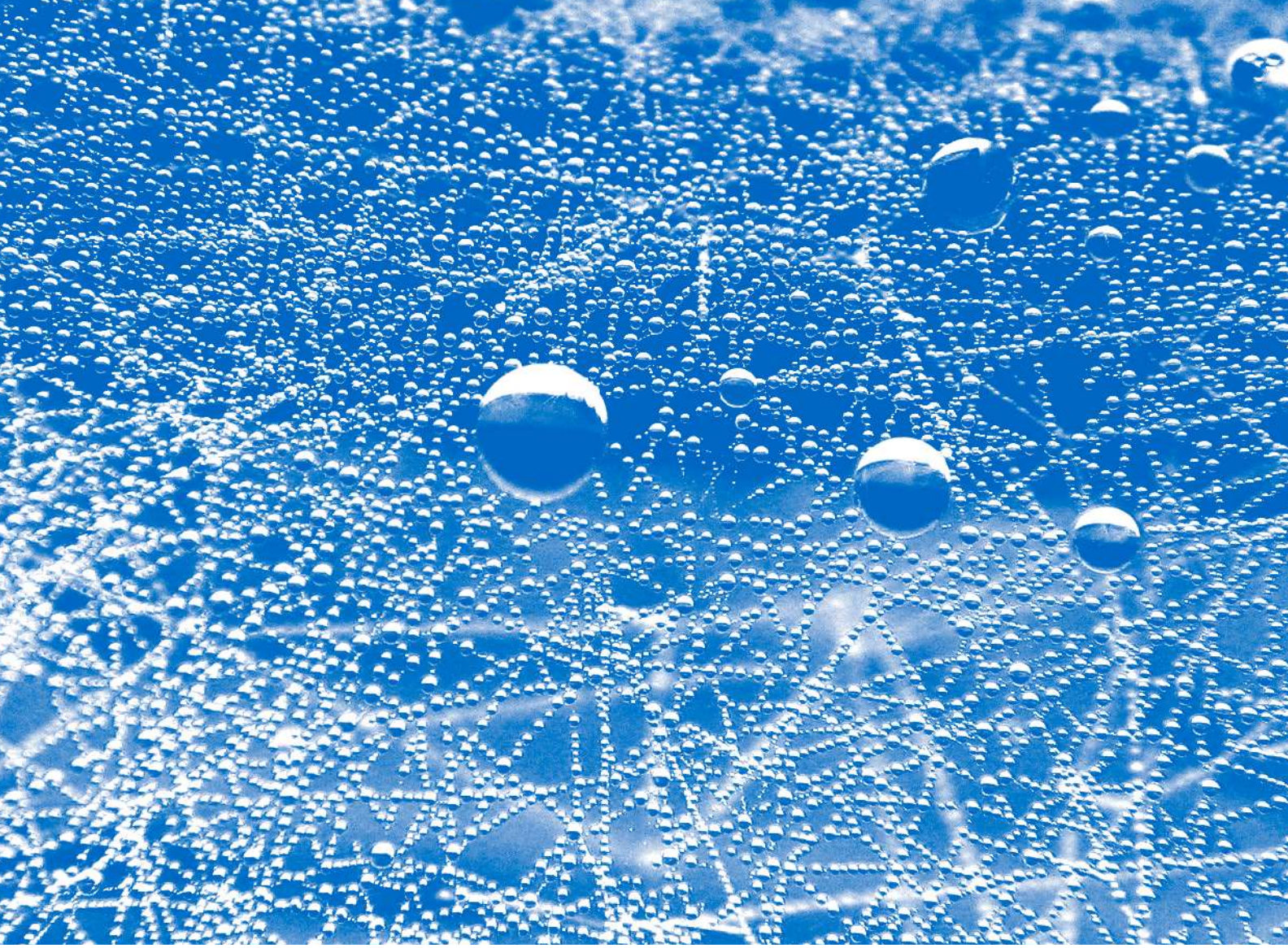

Redes inalámbricas de sensores: teoría y aplicación práctica

Roberto Fernández Martínez, Joaquín Ordieres Meré,
Francisco Javier Martínez de Pisón Ascacíbar,
Ana González Marcos, Fernando Alba Elías,
Ruben Lostado Lorza, Alpha Verónica Pernía Espinoza,
Integrantes del Grupo de Investigación EDMANS

UNIVERSIDAD DE LA RIOJA

REDES INALÁMBRICAS DE SENSORES: TEORÍA Y APLICACIÓN PRÁCTICA

MATERIAL DIDÁCTICO
Ingenierías
nº 26

Otros títulos de la colección

4. **Problemas de electrónica analógica**
   Antonio Zorzano Martínez
   1999, 118 pags. ISBN 84-88713-96-7

5. **Programar es fácil**
   Julio Blanco Fernández
   1999, 250 pags. ISBN 84-88713-97-5

6. **Problemas resueltos de topografía práctica**
   Jacinto Santamaría Peña
   1999, 84 pags. ISBN 84-88713-98-3

7. **Resistencia de materiales. Nivel básico**
   Eduardo Martínez de Pisón Ascacíbar
   1999, 316 pags. ISBN 84-95301-14-8

8. **Prácticas de C.A.D. Microstation 2D (2ª ed.)**
   José Lafargue Izquierdo
   1999, 224 pags. ISBN 84-95301-15-6

9. **Programación de proyectos**
   Joaquín Ordieres Meré
   1999, 96 pags. ISBN 84-95301-16-4

10. **Termodinámica fundamental (2ª ed.)**
    J. M. Sala Lizarraga, Luis M. López
    2000, 448 pags. ISBN 84-95301-25-3

11. **Termodinámica aplicada (2ª ed.)**
    J. M. Sala Lizarraga, L. M. López y Victor de la Peña
    2000, 584 pags. ISBN 84-95301-26-1

12. **Problemas Termodinámica fundamental (2ª ed.)**
    J. M. Sala Lizarraga, Luis M. López y Felipe Jiménez
    2000, 490 pags. ISBN 84-95301-27-X

13. **Problemas Termodinámica aplicada (2ª ed.)**
    J. M. Sala Lizarraga, Luis M. López y M.M: Ruiz de Adana
    2000, 432 pags. ISBN 84-95301-28-8

14. **Problemas de calor y frío industrial**
    L. M. López, J. M. Sala y J. M. Blanco Ilzarbe
    2000, 418 pags. ISBN 84-95301-29-6

15. **Apuntes de cartografía y proyecciones cartográficas**
    Jacinto Santamaría Peña
    2000, 74pags. ISBN 84-95301-30 X

16. **Apuntes de fotogrametría**
    Jacinto Santamaría Peña y Teófilo Sanz Méndez
    2000, 68pags. ISBN 84-95301-30-X

17. **Perspectiva: fundamentos y aplicaciones. Axonométrico. Caballera. Cónico**
    Ricardo Bartolomé Ramírez
    2000, 260 pags. ISBN 84-95301-33-4

18. **Problemas de resistencia de materiales. Nivel básico. Ingeniería agrícola**
    Eduardo Martínez de Pisón Ascacibar
    2001, 446 pags. ISBN 84-95301-44-X

19. **Sonometría y contaminación acústica.**
    Javier de Cos, J. Ordieres, M. Castejón, F. J. Martínez de Pisón
    2001, 384 pags. ISBN 84-95301-47-4

20. **Cuadernos de prácticas de informática industrial. Modulo 1: enunciados de prácticas en ensamblador**
    F. J. Martínez de Pisón, J. Ordieres, M. Castejón, F. J. de Cos, M. Gil.
    2001, 110 pags. ISBN 84-95301-58-X

21. **La oficina técnica y los proyectos industriales**
    F. J. Martínez de Pisón, J. Ordieres, M. Castejón, F. J. de Cos, E. P. Vergara, F. Alba.
    2 v. ISBN 84-95475-32-4

22. **Manual de prácticas de topografía y cartografía**
    Jacinto Santamaría Peña.
    Edición electrónica
    115 pags. ISBN 84-689-4103-4

23. **Problemas de electrotecnia**
    Edición electrónica.
    José Fernando Azofra Catroviejo
    113 pags. ISBN 84-689-7232-0

24. **Tecnicas y algoritmos básicos de visión artificial**
    Ana González Marcos y otros (Integrantes del Grupo de Investigación EDMANS)
    Edición electrónica
    2006, 96 pags. ISBN 84-689-9345-X

25. **Prácticas de CAD 3D SolidEdge V18: I. Entornos de pieza, conjunto y plano**
    José Lafargue Izquierdo
    2008, 331 pags. ISBN 978-84-95301-29-6

Roberto Fernández Martínez
Joaquín Ordieres Meré
Francisco Javier Martínez de Pisón Ascacíbar
Ana González Marcos
Fernando Alba Elías
Ruben Lostado Lorza
Alpha Verónica Pernía Espinoza

Integrantes del Grupo de Investigación EDMANS

REDES INALÁMBRICAS DE SENSORES: TEORÍA Y APLICACIÓN PRÁCTICA

UNIVERSIDAD DE LA RIOJA
SERVICIO DE PUBLICACIONES
2009

Redes inalámbricas de sensores: teoría y aplicación práctica / Roberto Fernádez Martínez ... [et al.] (integrantes del Grupo de Investigación EDMANS). – [Logroño] : Universidad de La Rioja, Servicio de Publicaciones, 2009
95 p. : il. ; 30 cm. – (Materia didáctico. Ingenierías ; 26)
ISBN 978-84-692-3007-7
1. Redes de ordenadores. 2. Telecomunicaciones. I. Fernández Martínez, Roberto. II. Universidad de La Rioja. Grupo de Investigación EDMANS
621.39

Edita: Universidad de La Rioja. Servicio de Publicaciones
Diseño de portada: Universidad de La Rioja. Servicio de Comunicación

ISBN 978-84-692-3007-7

Impreso en España - Printed in Spain.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE IMÁGENES

1 INTRODUCCIÓN

Actualmente, existe una problemática al dar soporte a un proceso cuando deben realizarse tareas de operación, mantenimiento y reparación. Los ejecutivos de plantas progresivas, los directores de mantenimiento y los planificadores del trabajo han querido siempre tener información acerca de los recursos disponibles cuando ellos realmente los necesitan. Desafortunadamente, todos los datos están esparcidos por diferentes sistemas de información, uno para cada tipo: histórico, datos fiables, análisis vibratorios, lectura de infrarrojos, análisis de aceites, monitorización en el control de aparatos y varios más.

En muchos de estos sistemas hay una persona encargada que debe desplazarse varias veces, tanto para obtener la información necesaria para hacer un diagnóstico y realizar la tarea, como para recoger las órdenes de trabajo, realizar los informes y terminar la tarea en sí. Incluso a veces, la información no es recogida por el sistema debido al costo de transmisión, a las dificultades de acceso o a que ésta debe ser escrita a mano.

En las empresas actuales es difícil, si no imposible, ver los diferentes tipos de información en el mismo ordenador, compilarlos y sincronizarlos dentro de un informe que sirva como base inteligente de recursos a la hora de tomar decisiones en la dirección. Incluso cuando los sistemas son accesibles desde el mismo terminal, éste normalmente requiere de diferentes programas que usan distintos lenguajes.

Lo ideal, y lo que las organizaciones quieren, es crear una plataforma que cubra todas sus necesidades a la hora de realizar este tipo de tareas, integrando tanto la información que se recibe en tiempo real, como la información estática necesaria, ofreciendo interfaces que permitan acceder de forma remota a todos los servicios de la plataforma.

A pesar de todo ello, la cantidad de datos leídos cada día es mayor ya que los sensores y las redes de sensores tienen un crecimiento imparable. La habilidad para gestionar esta gran cantidad de datos, como se procesan y usan, va a ser crucial para obtener las soluciones. Los tres principales pasos a seguir para la explotación de los datos son éstos:

- Los datos recolectados deben ser convertidos en información, a través de una recolección sistemática, una integración y una organización. Este proceso basado en recogida de datos de sensores es una de las áreas donde la investigación está más abierta y existen muchas definiciones para convertir los datos en información útil.

- A partir de la información debemos obtener un conocimiento. El proceso de convertir esta información a través de un análisis predictivo es definido como descubrimiento del conocimiento. A diferencia de lo que podría ser un estudio estadístico básico, se considera descubrimiento del conocimiento a la habilidad de generar conocimientos no triviales, no intuitivos y llenos de significado a partir de la información.

- Obtener una decisión a partir de un conocimiento. La información obtenida y los conocimientos adquiridos tienen que ser estudiados en conjunto, junto con un análisis en tiempo real que nos permita tomar decisiones. Estos análisis deben ayudar al usuario final a tomar decisiones sobre requerimientos estratégicos, operacionales o tácticos.

Estos tres pasos deben ser estrechamente relacionados con el contexto del dominio o de la aplicación donde la información, el conocimiento y las decisiones son buscadas.

Este libro se centra en el primer punto: cómo captar, recoger, gestionar, enviar y almacenar la información para que posteriormente puedan desarrollarse los pasos siguientes. Para ello se usará un sistema de medida en el que se realizan funciones de medición de varias magnitudes físicas, procesando esta información para poder conocer el funcionamiento del sistema que se pretende gestionar, según los datos obtenidos en el proceso de adquisición de datos y medición.

Hoy en día, los sensores pueden encontrarse en gran número de sistemas y dispositivos electrónicos. La mayor parte de estos sensores adolecen de la capacidad de procesar y analizar los datos que detectan, limitándose a funcionar como un transductor que realiza la medición de una o más variables del entorno y envía dicha información a un procesador central.

En cualquier sistema, no sólo en los industriales, se realizan mediciones de magnitudes que más tarde serán el apoyo para actuar dentro de éste. Los sensores son los dispositivos del sistema de medida que

interactúan con el sistema físico que se pretende estudiar, los cuales permiten la toma de medidas de las distintas magnitudes físicas que se van a analizar.
Los sensores no son nuevos dentro de un proceso industrial o dentro de una monitorización de un sistema. Pero tienen ciertas limitaciones:

- El reducido número de componentes debido a determinadas características como el precio o el tamaño.
- Dificultad de añadir un nuevo componente a un sistema e integrarlo fácilmente.

Estos problemas se van a solucionar gracias a una nueva tecnología basada en pequeños dispositivos capaces de monitorizar el entorno mediante sensores, que se disponen de una manera ad-hoc en una determinada área, dotados de inteligencia propia, capaces de organizarse a sí mismos y de interconectarse de forma inalámbrica con otros semejantes. Esta tecnología ha sido denominada como redes de sensores inalámbricas (WSN, Wireless Sensor Networks) y esta integrada por unos dispositivos con características de sistemas empotrados. Dicha tecnología en los últimos años está adquiriendo cada vez mayor importancia, por la aparición de nuevos aparatos micro-electro-mecánicos (MEMS) que están revolucionando las posibilidades de captación de variables. Estos aparatos, MEMS pueden ayudarnos, gracias a que nos proporcionan una baja demanda de energía y unos procesadores muy potentes.

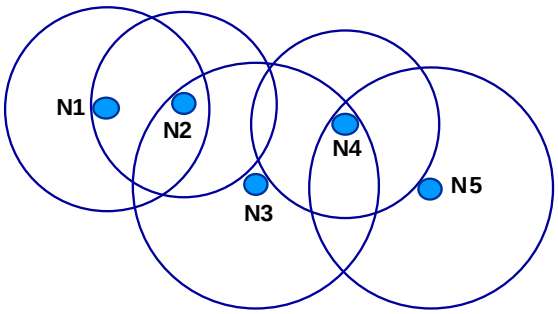

Figura 1 Ejemplo de red ad-hoc

En los últimos años, varios laboratorios de investigación, y especialmente multinacionales como Intel, han apostado fuertemente por esta tecnología. En diversos informes se augura que este tipo de redes conllevará una revolución tecnológica similar a la que tuvo la aparición de Internet. De hecho, DARPA (Defense Advanced Research Projects Agency), institución dependiente del Departamento de Defensa estadounidense, también se ha involucrado en el desarrollo de este tipo de redes. Ya se habla de redes de vigilancia global del planeta, capaces de registrar los hábitos de la gente, realizar un seguimiento de personas y mercancías concretas, monitorizar el tráfico, etc. Aunque para ello habrá que esperar todavía unos años, sí que han surgido múltiples iniciativas y proyectos de investigación de enorme interés y aplicabilidad práctica.

2 REDES INALÁMBRICAS DE SENSORES

2.1 INTRODUCCIÓN

La idea de una red de sensores surge gracias a las posibilidades que nos da la tecnología de crear una red de dispositivos de captura constante, que nos permita registrar y almacenar una determinada información, transmitir datos de un dispositivo a otro, y después retransmitir toda la información para almacenarla en una localización central. Teniendo siempre en cuenta que todo ello funcionará con un gasto de energía muy reducido.

Una red de este tipo es un flexible y poderoso instrumento para poder monitorizar complejos sistemas, donde situar los sensores puede ser imposible de cualquier otra manera. El objetivo de la recolección de datos por los sensores en la monitorización, es la obtención de los datos teniendo como única limitación las características de los sensores.

Podemos crear una infraestructura sólida y barata que permita que cada sensor proporcione unas medidas y una información detallada de una zona localizada, lo cual puede ser difícil de obtener con la instrumentación tradicional. Los dispositivos son unidades autónomas que constan de un microprocesador, una fuente de energía, un radiotransceptor y un elemento sensor.

A medida que este tipo de redes sean más utilizadas y gracias también, a los avances tecnológicos constantes en las técnicas con semiconductores, los precios de los dispositivos bajarán a la vez que las prestaciones subirán. Las aplicaciones en las que podrán utilizarse se multiplicarán, así como las áreas de investigación científica.

2.2 REDES INALÁMBRICAS DE SENSORES

Una red de sensores (del inglés sensor network) es una red de diminutos dispositivos, equipados con sensores, que colaboran en una tarea común.Las redes de sensores están formadas por un grupo de sensores con ciertas capacidades sensitivas y de comunicación inalámbrica que permiten formar redes ad-hoc sin infraestructura física preestablecida ni administración central.

La expresión ad-hoc hace referencia a una red en la que no hay un nodo central, sino que todos los dispositivos están en igualdad de condiciones. Ad-hoc es el modo más sencillo para crear una red, un tipo de red formada por un grupo de nodos móviles que forman una red temporal sin la ayuda de ninguna infraestructura externa. Para que esto se pueda llevar a la práctica es necesario que los nodos se puedan ayudar mutuamente para conseguir un objetivo común: que cualquier paquete llegue a su destino aunque el destinatario no sea accesible directamente desde el origen. El protocolo de encaminamiento es el responsable de descubrir las rutas entre los nodos para hacer posible la comunicación.

Las redes de sensores son un concepto relativamente nuevo en adquisición y tratamiento de datos con múltiples aplicaciones en distintos campos, tales como entornos industriales, domótica, entornos militares, detección ambiental.

Esta clase de redes se caracterizan por su facilidad de despliegue y por ser autoconfigurables, pudiendo convertirse en todo momento en emisor, receptor, ofrecer servicios de encaminamiento entre nodos sin visión directa, así como registrar datos referentes a los sensores locales de cada nodo. Otra de sus características es su gestión eficiente de la energía, que les permite obtener una alta tasa de autonomía que las hacen plenamente operativas.

La miniaturización de los dispositivos, cada día mayor, dio a luz la idea de desarrollar dispositivos extremadamente pequeños y baratos que se comunican de forma inalámbrica y se organizan autónomamente. La idea de estas redes es repartir aleatoriamente estos nodos en un territorio amplio, del cual los nodos captan cierta información hasta que sus recursos energéticos se agoten. Los adjetivos 'pequeño', 'barato' y 'autónomo' dieron a conocer la idea como polvo inteligente (de ingles smart dust).

Las redes de sensores es un tema muy activo de investigación en varias universidades, aunque ya empiezan a existir aplicaciones comerciales basadas en este tipo de redes. Una de las redes de sensores más grande desarrollada hasta la fecha consistió en 800 nodos y fue puesta en servicio el 27 agosto 2001, por una duración breve, en la Universidad de Berkeley para demostrar la potencia de esa técnica en una presentación.

Incluso podemos comprobar como los costos de instalación de una de estas redes son bastante menores a un sistema de adquisición tradicional.

Tabla 1 Comparativa de precios

	Tradicional	Redes de sensores inalámbricas
Coste del sensor	2000 $	350 $
Coste de la adquisición de datos	65 $	0 $
Coste del cableado	75 $	15 $
Coste por sensor	**2140 $**	**365 $**

2.3 ARQUITECTURA DEL SISTEMA

Las redes de sensores están formadas por un conjunto de pequeños dispositivos denominados nodos sensores, con capacidad limitada de cómputo y comunicación, cuyo tiempo de vida depende de una batería adjunta al dispositivo. El tiempo de vida de la red de sensores dependerá por tanto del tiempo de vida de la batería de sus nodos. Estos dispositivos se encuentran dispersos de manera ad-hoc en una determinada área a monitorizar. En redes de comunicación, dicha expresión hace referencia a una red en la que no hay un nodo central, sino que todos los nodos están en igualdad de condiciones.
Típicamente, el modelo seguido por las aplicaciones es el siguiente: realizar una serie de mediciones sobre el medio, transformar dicha información en digital en el propio nodo y transmitirla fuera de la red de sensores vía un elemento gateway a una estación base, donde la información pueda ser almacenada y tratada temporalmente para acabar finalmente en un servidor con mayor capacidad que permita componer un histórico o realizar análisis de datos.
En una red de sensores inalámbricos, por lo tanto, podemos encontrarnos:

- nodos inalámbricos
- puertas de enlace
- estaciones base

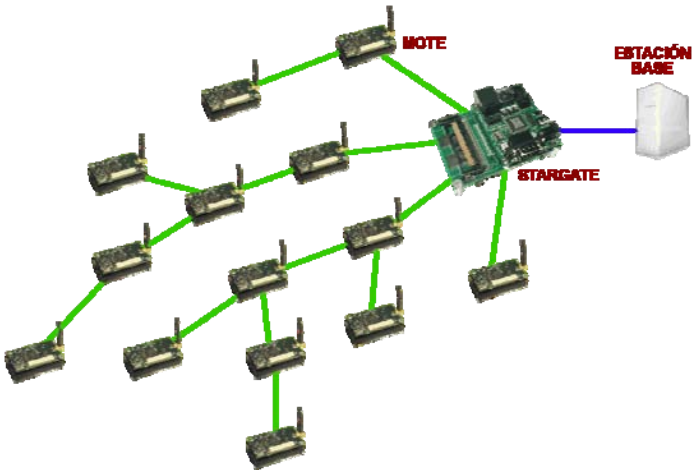

Figura 2 Red inalámbrica de sensores

2.3.1 NODO INALÁMBRICO

Los nodos inalámbricos se llaman motas, del inglés 'mote', por su ligereza y reducido tamaño. Son dispositivos electrónicos capaces de captar información proveniente del entorno en el que se encuentran, procesarla y transmitirla inalámbricamente hacia otro destinatario.
Diseñar un mota no se reduce a miniaturizar un ordenador personal. Hay que tener en cuenta que queremos un espacio reducido, un consumo muy bajo de energía y un coste de los dispositivos reducido. Y en contraposición a esto una potencia de ejecución de programas elevadas y una transmisión de datos eficaz y con amplia longitud de emisión.

Estos motas son diseñados y programados para formar parte de una red con un objetivo particular, lo que quiere decir que un mota aislado tiene muy poca utilidad.
El hardware de cada uno de estos dispositivos tiene varias partes bien diferenciadas.

Figura 3 Arquitectura de un nodo inalámbrico

2.3.1.1 Procesador

Es el componente que interpreta y procesa los datos para transmitirlos a otra estación. También gestiona el almacenamiento de datos en la memoria.
Puesto que de un nodo sensor se espera una comunicación y una recogida de datos mediante sensores, debe existir una unidad de procesado, que se encargue de gestionar todas estas operaciones.
Hay muchos tipos diferentes de productos disponibles en el mercado para ser integrados en un nodo, como microcontroladores, microprocesadores y FPGA.

FPGA: Actualmente éstas presentan varias desventajas, la mayor de ellas es el consumo. A pesar de que en el mercado podemos encontrar FPGAs de bajo consumo, este consumo no es lo suficientemente bajo como debería ser para este tipo de nodos. Esto no significa que en un futuro cercano, éstas sean una buena opción si se consigue que reduzcan el consumo.

Microprocesadores: Han sido sustituidos por los microcontroladores, ya que éstos integran dentro de un mismo dispositivo, un microprocesador y memoria.

Microcontroladores: Como se ha dicho, incluyen un microprocesador y memoria, pero además tienen una interface para ADCs, UART, SPI, temporizadores y contadores. Hay muchos tipos de microcontroladores que van desde los 4 bits hasta 64 bits, con una variación del numero de temporizadores, con diferentes consumos de energía,…

Actualmente los más utilizados son los siguientes procesadores de bajo consumo:

- ARM7: Se denomina ARM a una familia de microprocesadores RISC diseñados por la empresa Acorn Computers y desarrollados por Advanced RISC Machines Ltd., una empresa derivada de la anterior. http://www.arm.com/

- Atmel AVR: Los AVR son una familia de microcontroladores RISC de Atmel. La arquitectura de los AVR fue concebida por dos estudiantes en el Norwegian Institute of Technology, y posteriormente refinada y desarrollada en Atmel Norway, la empresa subsidiaria de Atmel, fundada por los dos arquitectos del chip.

  El AVR es una CPU de arquitectura Harvard. Tiene 32 registros de 8 bits. Algunas instrucciones sólo operan en un subconjunto de estos registros. http://www.atmel.com/products/avr/

- Intel Xscale: El Intel XScale es un núcleo de microprocesador, la implementación de Intel de la quinta generación de la arquitectura ARM. La tecnología fue vendida a Marvell Technology Group en junio de 2006.

  Está basado en el ISA v5TE sin las instrucciones de coma flotante. El XScale usa un entero de 7 niveles y 8 niveles de memoria Superpipeline de arquitectura RISC. Es el sucesor de la línea de

microprocesadores y microcontroladores Intel StrongARM, que Intel adquirió de la división de Semiconductores Digitales de DEC como efecto colateral de un pleito entre las dos compañías. http://www.marvell.com/

- Intel 8051: El Intel 8051 es un microcontrolador (μC) desarrollado por Intel en 1980 para uso en productos embebidos. Es un microcontrolador muy popular.

  Los núcleos 8051 se usan en más de 100 microcontroladores de más de 20 fabricantes independientes como Atmel, Dallas Semiconductor, Philips, Winbond, entre otros.

  La denominación oficial de Intel para familia de μCs 8051 es MCS 51.

  Este microcontrolador está basado en una Arquitectura Harvard (es decir, existen espacios de direcciones separados para código y datos). Aunque originariamente fue diseñado para aplicaciones simples, se permite direccionar 64 KB de ROM externa y 64 KB de RAM por medio de líneas separadas chip select para programa y datos. http://www.intel.com/design/mcs51/

- PIC: Los 'PIC' son una familia de microcontroladores tipo RISC fabricados por Microchip Technology Inc. y derivados del PIC1650, originalmente desarrollado por la división de microelectrónica de General Instruments.

  El nombre actual no es un acrónimo. En realidad, el nombre completo es PICmicro, aunque generalmente se utiliza como Peripheral Interface Controller (Controlador de Interfaz Periférico).

  El PIC original se diseñó para ser usado con la nueva UCP de 16 bits CP16000. Siendo en general una buena UCP, ésta tenía malas prestaciones de E/S, y el PIC de 8 bits se desarrolló en 1975 para mejorar el rendimiento del sistema quitando peso de E/S a la UCP. El PIC utilizaba microcódigo simple almacenado en ROM para realizar estas tareas; y aunque el término no se usaba por aquel entonces, se trata de un diseño RISC que ejecuta una instrucción cada 4 ciclos del oscilador. http://www.microchip.com/

- TI MSP430: El MSP430 es una familia de microcontroladores producidos por Texas Instruments. Construido con una CPU de 16 bits, el MSP430 está diseñado para aplicaciones empotradas de bajo costo y bajo consumo de energía. La arquitectura tiene reminiscencias del DEC PDP-11. Desafortunadamente, el MSP430 carece de una característica muy importante del PDP11, la cual era la memoria para indexar memoria. Esta característica permitía que las rutinas de interrupción se escribieran sin utilizar registros, por lo que no requería apilamientos. El MSP430 es muy útil para aplicaciones inalámbricas o para aplicaciones de bajo consumo. http://www.ti.com/msp430

2.3.1.2 Alimentación

Normalmente la fuente de alimentación son baterías difícilmente sustituibles o transformadores con salida adecuada para el nodo si se dispone de toma de corriente. Para las situaciones en donde no se dispone de red eléctrica y las posibilidad de sustituir las baterías es muy complicada, se están estudiando diferentes técnicas para alimentar el sensor, como puede ser mediante placas solares.
Ante la limitación de la vida útil del dispositivo hay que realizar una gestión eficiente del consumo energético.
El consumo de energía viene dado por lo que consumen los sensores, la comunicación y el procesado. La mayor cantidad de energía es consumida en la transmisión de información, siendo menor en el procesado y uso de los sensores. Por ejemplo el coste de transmisión de 1 Kb. a una distancia de 100 metros es aproximadamente el mismo que ejecutar 3 millones de instrucciones por un procesador de 100 millones de instrucciones por segundo.
Las baterías son la principal fuente de energía de estos motes, pudiendo ser recargables o no recargables. Están clasificados según el material electroquímico usado para el electrodo como pueden ser NiCd (níquel-cadmio), NiZn (níquel -zinc), Nimh (níquel metal hidruro), y Litio-Ion.
Actualmente se están estudiando sistemas basados en energía renovables para solucionar el problema de la energía en estos nodos, basados en energía solar, termo generación, energía basada en vibraciones,

Tabla 2 Densidad de energía en diferentes fuentes

Fuente de energía	Densidad de energía
Baterías (Zinc Aire)	1050-1560 mWh/cm3
Baterías (recargable Litio)	300 mWh/cm3 (3-4 V)
Solar (exterior)	150 mWh/cm3 (expuesto al sol) 0.15 mWh/cm3 (día nublado)
Solar (interior)	0.006 mWh/cm3 (nivel escritorio oficina) 0.57 mWh/cm3 (< 60 W lámpara de escritorio)
Vibración	0.01-0.1 mWh/cm3
Ruido acústico	3*10-6 mWh/cm2 a 75 Db 9.6*10-4 mWh/cm2 a 100 Db
Reacción nuclear	80 mW/cm3 , 106 mWh/cm3

2.3.1.3 Comunicación inalámbrica

El dispositivo de comunicación se trata de un dispositivo vía radio que permite enviar y recibir datos para comunicarse con otros dispositivos dentro de su rango de transmisión.
Los nodos usan la banda ISM que son bandas reservadas internacionalmente para uso no comercial de radiofrecuencia electromagnética en áreas industrial, científica y médica. El uso de estas bandas de frecuencia está abierto a todo el mundo sin necesidad de licencia, respetando las regulaciones que limitan los niveles de potencia transmitida.
Los medios a elegir para realizar una comunicación inalámbrica son varios, radio frecuencia, comunicación óptica mediante láser e infrarrojos. La comunicación por láser es la que menos energía consume pero requiere de una comunicación visual entre emisor y receptor, y además también depende de las condiciones atmosféricas. Los infrarrojos como el láser, no necesitan antena, aunque es bastante limitado en su capacidad de transmisión. La radio frecuencia, RF, es la más adecuada para usar en aplicaciones inalámbricas. Las WSN usan las frecuencias de comunicación que andan entre 433 MHz y 2.4 GHz.
Las funcionalidades de emisión y recepción se combinan en un solo aparato que es llamado transceptor. Los estados de operación son emitir, recibir, dormir e inactividad. En los actuales modelos de transceptor, el modo inactivo consume casi igual que el modo recepción. Por lo que es mejor tener completamente apagado las comunicaciones radio, en el modo Inactivo, cuando no se esta emitiendo ni recibiendo. También es significativa la cantidad de energía consumida cuando cambia de modo durmiente a transmisión de datos.
Los sistemas mas populares dentro de los sistemas de comunicación de radio para nodos de redes inalámbricos son:

- Chipcon CC1000 http://www.chipcon.com/
- Chipcon CC1020 http://www.chipcon.com/
- Chipcon CC2420 http://www.chipcon.com/
- Xemics XE1205 http://www.semtech.com/
- 802.15.4 Chipsets and SoC http://www.jennic.com/

Una comparativa de las principales características de estos dispositivos es:

Tabla 3 Características de dispositivos inalámbricos I

	CC1000	CC1021	CC2420	TR1000	XE1205
Manufacturer	Chipcon	Chipcon	Chipcon	RFM	Semtech
Operating Frequency [MHz]	300 - 1000	402 - 470 / 804 - 940	2400	916	433 / 868 / 915
Bit Rate [kbps]	76.8	153.6	250	115.2	1.2 - 152.3
Power Supply Voltage [V]	2.1 - 3.6 (typ. 3.0)	2.3 - 3.6 (typ. 3.0)	2.1 - 3.6 (int. 1.8)	2.2 - 3.7 (typ. 3.0)	2.4 - 3.6

Tabla 4 Características de dispositivos inalámbricos II

	CC1000	CC1021	CC2420	TR1000	XE1205
Sleep Mode [uA]	0.2 - 1 (osc. core off)	1.8 (core off)	1	0.7	0.2
Standby Mode [uA]			426 (Voltage and osc. runnning)		850

	CC1000	CC1021	CC2420	TR1000	XE1205
RX [mA]	9.3 (433MHz) / 11.8 (868MHz)	19.9	19.7	3.8 (115.2kbps)	14
TX Min [mA]	8.6 (-20dBm)	14.5 (-20dBm)	8.5 (-25dBm)		33 (+5dBm)
TX Max [mA]	25.4 (+5dBm)	25.1 (+5dBm)	17.4 (0dBm)	12 (+1.5dBm)	62 (+15dBm)

Todos los dispositivos en las WSN usan la tecnología Zigbee, más ampliamente explicada en el capítulo correspondiente.

2.3.1.4 Sensores

Los sensores son dispositivos hardware que producen una respuesta medible ante un cambio en un estado físico, como puede ser temperatura o presión. Los sensores detectan o miden cambios físicos en el área que están monitorizando. La señal analógica continua detectada es digitalizada por un convertidor analógico digital y enviada a un controlador para ser procesada.
Las características y requerimientos que un sensor debe tener son un pequeño tamaño, un consumo bajo de energía, operar en densidades volumétricas altas, ser autónomo y funcionar desatendidamente y tener capacidad para adaptarse al ambiente.
Los sensores pueden estar clasificados en tres categorías:

- Sensores pasivos omnidireccionales: Los sensores pasivos captan los datos sin necesidad de manipular el entorno. Son autoalimentados y solo usan la energía para amplificar la señal analógica captada. No hay ninguna noción de 'dirección' involucrada en estas mediciones.
- Sensores pasivos unidireccionales: Son sensores pasivos que tienen bien definida la dirección desde donde deben captar la información. Un ejemplo típico es una cámara.
- Sensores activos: Este tipo de sensores sondean el ambiente, por ejemplo un radar o un sonar o algún tipo de sensor sísmico que generan ondas expansivas a través de pequeñas explosiones.

2.3.1.5 Memoria

Desde un punto de gasto de energía, las clases más relevantes de memoria son la memoria integrada en el chip de un microcontrolador y la memoria flash, la memoria RAM fuera del chip es raramente usada. Las memorias flash son usadas gracias a su bajo coste y su gran capacidad de almacenamiento.
La memoria flash es una forma desarrollada de la memoria EEPROM que permite que múltiples posiciones de memoria sean escritas o borradas en una misma operación de programación mediante impulsos eléctricos, frente a las anteriores que sólo permite escribir o borrar una única celda cada vez. Por ello, flash permite funcionar a velocidades muy superiores cuando los sistemas emplean lectura y escritura en diferentes puntos de esta memoria al mismo tiempo.
Las memorias flash son de carácter no volátil, esto es, la información que almacena no se pierde en cuanto se desconecta de la corriente, una característica muy valorada para la multitud de usos en los que se emplea este tipo de memoria.
Los requerimientos de memoria dependen mucho de la capacidad que necesite nuestra aplicación. Hay dos categorías de memorias según el propósito del almacenamiento:

- Memoria usada para almacenar los datos recogidos por la aplicación.
- Memoria usada para almacenar el programa del dispositivo.

2.3.2 PUERTA DE ENLACE

Elementos para la interconexión entre la red de sensores y una red de datos (TCP/IP). Es un nodo especial sin elemento sensor, cuyo objetivo es actuar como puente entre dos redes de diferente tipo.
En este tipo de aplicaciones donde se usan redes de sensores, éstas no pueden operar completamente aisladas y deben contar con alguna forma de monitoreo y acceso a la información adquirida por los nodos de la red de sensores. De aquí surge la necesidad de conectar las redes de sensores a infraestructuras de redes existentes tales como Internet, redes de área local (LAN) e intranets privadas. Los dispositivos que

realizan la función de interconectar dos redes de diferente naturaleza se les llama dispositivo puerta de enlace; pero el término más conocido en el ambiente de las redes es gateway.

2.3.3 ESTACIÓN BASE

Recolector de datos basado en un ordenador común o sistema empotrado. En una estructura normal todos los datos van a parar a un equipo servidor dentro de una base de datos, desde donde los usuarios pueden acceder remotamente y poder observar y estudiar los datos.

3 TOPOLOGÍAS DE RED

3.1 INTRODUCCIÓN

Hay varias arquitecturas que pueden ser usadas para implementar una aplicación de WSN como pueden ser estrella, malla o una híbrida entre ellas dos. Cada topología presenta desafíos, ventajas y desventajas. Para entender las diferentes topologías es necesario conocer los diferentes componentes de la WSN.

- Nodos finales: Compuesto por sensores y actuadores donde se capturan los datos sensores. Para las redes basadas en ZigBee son llamados RFD (Reduced Functional Devices).
- Routers: Dan cobertura a redes muy extensas pudiendo salvar obstáculos, problemas de congestión en la emisión de la información y posibles fallos en alguno de los aparatos.
- Puertas de enlace: Recoge los datos de la red, sirve como punto de unión con una red LAN o con Internet.

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

3.2 TOPOLOGÍA EN ESTRELLA

Una topología en estrella es un sistema donde la información enviada sólo da un salto y donde todos los nodos sensores están en comunicación directa con la puerta de enlace, usualmente a una distancia de 30 a 100 metros. Todos los nodos sensores son idénticos, nodos finales, y la puerta de enlace capta la información de todos ellos. La puerta de enlace también es usada para transmitir datos al exterior y permitir la monitorización de la red.
Los nodos finales no intercambian información entre ellos, sino que usan la puerta de enlace para ello, si es necesario.

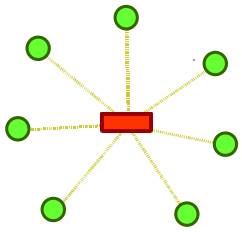

Figura 4 Topología en estrella

La topología en estrella es la que menor gasto de energía desarrolla, pero por el contrario esta limitada por la distancia de transmisión vía radio entre cada nodo y la puerta de enlace. Tampoco tiene un camino de comunicación alternativo en caso de que uno de los nodos tenga obstruido el camino de comunicación, lo que lleva a que en este caso la información de ese nodo sea perdida.

3.3 TOPOLOGÍA EN MALLA

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

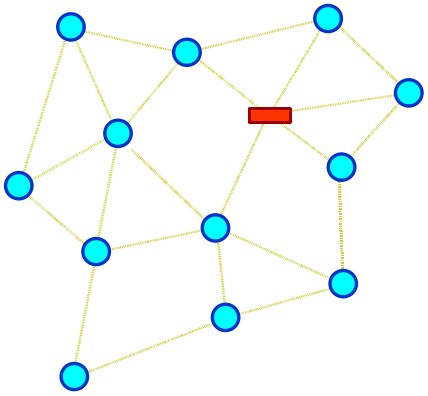

Figura 5 Topología en malla

La propagación de los datos a través de los nodos hacia la puerta de enlace hace posible, por lo menos en teoría, crear una red con una extensión posible ilimitada. Este tipo, también es altamente tolerante a fallos ya que cada nodo tiene diferentes caminos para comunicarse con la puerta de enlace. Si un nodo falla, la red se reconfigurará alrededor del nodo fallido automáticamente.
Dependiendo del número de nodos y de la distancia entre ellos, la red puede experimentar periodos de espera elevados a la hora de mandar la información.

3.4 TOPOLOGÍA HÍBRIDA ESTRELLA-MALLA

Este tipo de red busca combinar las ventajas de los otros dos tipos, la simplicidad y el bajo consumo de una topología en estrella, así como la posibilidad de cubrir una gran extensión y de reorganizarse ante fallos de la topología en malla. Este tipo crea una red en estrella alrededor de routers pertenecientes a una red en malla. Los routers dan la posibilidad de ampliar la red y de corregir fallos en estos nodos y los nodos finales se conectan con los routers cercanos ahorrando energía.

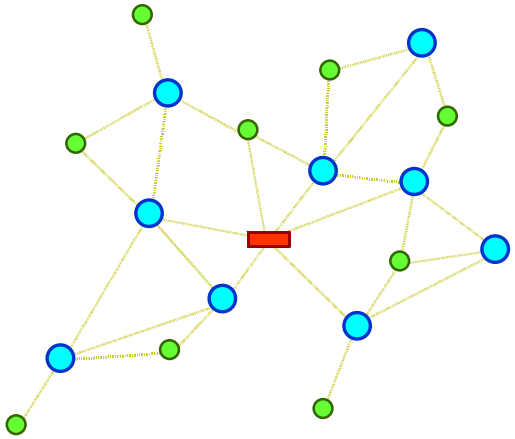

Figura 6 Topología híbrida Malla-Estrella

4 REDES AD-HOC Y TECNOLOGÍA INALÁMBRICA

4.1 REDES AD-HOC

Las redes mas usadas en las implantaciones de redes de sensores inalámbricas son las redes malladas tipo ad-hoc. Son redes sin infraestructura, flexibles en las cuales todas las estaciones ofrecen servicios de encaminamiento para permitir la comunicación de estaciones que no tienen conexión inalámbrica directa.
La principal característica de las redes móviles ad-hoc es que todos los dispositivos que forman parte de la red, además de funcionar como terminales finales, realizan también funciones de retransmisión de paquetes típicamente asociadas a routers.
Esta cualidad nos permite encaminar paquetes a destinos sin cobertura directa a través de otros nodos intermedios que se encuentren en la red. De este modo se nos ofrece la posibilidad de incrementar de una manera extraordinaria la movilidad y el tamaño de una red de datos inalámbrica. La funcionalidad principal de estas redes es la de crear de una manera rápida y eficaz una red temporal en lugares carentes de una infraestructura de red.
Las principales características de una red ad-hoc son:

- **Movilidad**: Este aspecto es la razón de ser de las redes ad-hoc. Los nodos se pueden reposicionar o simplemente ser móviles, siempre que no salgan del alcance radio. Se pueden desplegar rápidamente sin la necesidad de descubrir la zona o formar grupos, es decir, cada nodo es individual y solvente.
- **Multisalto** (Multihopping): Una red multihopping es una red donde el camino de la fuente al destino atraviesa varios nodos intermedios.
- **Autoorganización**: La red de forma autónoma debe determinar sus propios parámetros de configuración: dirección, encaminamiento, clustering, indicador de posición, etc.
- **Conservación de la energía**: Los nodos móviles, tienen una batería limitada y a no ser que dispongan de algún mecanismo de carga (por ejemplo, un panel solar), no tienen capacidad de recarga. Es muy importante diseñar unos protocolos (MAC, encaminamiento) eficientes, con la finalidad de mejorar el rendimiento y prolongar la autonomía de las baterías.
- **Escalabilidad**: En algunos tipos de redes, el número de nodos puede crecer hasta llegar a varios miles. Como no existe un access point concreto, la incorporación y descarte de nodos es un proceso sencillo y transparente.
- **Seguridad**: Las redes inalámbricas son vulnerables a ataques, y las redes ad-hoc lo son especialmente. Pueden padecer tanto ataques activos como pasivos y el atacante puede emular a un nodo legítimo y capturar paquetes de datos y control, destruir tablas de encaminamiento, etc.

4.1.1 PROTOCOLOS

Sin embargo, para que lo anterior sea viable se hace necesaria la inclusión en la red de protocolos de encaminamiento que nos permitan crear las rutas hacia los destinos deseados. Los protocolos tradicionales propios de redes fijas no se adaptan bien a este tipo de entornos tan dinámicos y, por tanto, será necesario el diseño específico de protocolos para proporcionar un comportamiento eficiente a la red.
La principal clasificación es la que diferencia entre protocolos proactivos y reactivos (o bajo demanda): Por una parte, en los protocolos proactivos, periódicamente se envía información de encaminamiento para que en cualquier momento cualquier nodo pueda comunicarse con cualquier otro de la red. Esta característica proporciona una rápida respuesta ante solicitudes de ruta y ofrece un buen comportamiento en situaciones donde la tasa de movilidad es alta. Sin embargo la sobrecarga que se introduce en la red con información de control es alta. Entre estos protocolos podemos destacar el protocolo DSDV (Destination-Sequenced Distance Vector). Por otra parte, los protocolos reactivos sólo crean rutas cuando es necesario. Son protocolos bajo demanda donde la sobrecarga es mucho menor, pero los retrasos de establecimiento de rutas son mayores. Podemos nombrar AODV (Ad hoc On-Demand Distance Vector)

como protocolo reactivo. Existen algunos protocolos híbridos en los que se mantiene una filosofía proactiva en un ámbito local y reactiva a nivel más global, como el protocolo ZRP (Zone Routing Protocol).
Los principales protocolos utilizados son:

- Destination-Sequenced Distance Vector (DSDV). DSDV es esencialmente una modificación del algoritmo de encaminamiento Vector Distancia Bellman-Ford, bien conocido por su utilidad en redes fijas, como por ejemplo en el protocolo RIP. En este algoritmo, los nodos vecinos intercambian periódicamente (proactivo) sus tablas de encaminamiento enteras para estimar la distancia a la que se encuentran los demás nodos no vecinos. Las modificaciones introducidas por DSDV proporcionan básicamente la obtención de rutas sin bucles mediante la introducción de números de secuencia para la determinación de las rutas más nuevas. Aunque DSDV sólo proporciona un camino para cada destino, siempre elige el camino más corto basándose en el número de saltos hacia este destino. DSDV utiliza dos tipos de mensajes de actualización, uno más grande (full-dump) y otro mucho más pequeño (incremental). Los mensajes incrementales pueden utilizarse para actualizaciones intermedias entre envíos periódicos (full-dump) de la tabla entera de encaminamiento. Además se realizan estimaciones de los tiempos de establecimientos de ruta que retrasarán el envío de mensajes incrementales para evitar envíos en cadena de estos mensajes.

- Optimized Link-State Routing Algorithm (OLSR): OLSR incorpora la filosofía utilizada en protocolos tradicionales como OSPF de 'Estado de los Enlaces'. En este algoritmo todos los nodos se intercambian mensajes para formarse una visión consistente de toda la red y así poder decidir el encaminamiento de paquetes. OLSR adolece del mismo problema que DSDV debido a la necesidad de intercambio de un gran número de mensajes periódicos (proactivo). Aquí, el problema podría llegar a ser mayor, ya que además de mensajes 'hello' a los vecinos, envía mensajes de control 'tc' (Topology Control) que se retransmiten a todos los nodos de la red. Sin embargo se ha conseguido una gran optimización en la retransmisión de estos mensajes con la incorporación de la técnica de retransmisión multipunto, a través de la cual, los mensajes sólo son retransmitidos por el mínimo número de nodos necesarios para alcanzar a todos los demás. Estos nodos son conocidos como grupo de retransmisores multipunto (MPR's).

- Dynamic Source Routing (DSR): El protocolo DSR se fundamenta en el encaminamiento desde el origen, es decir, los paquetes de datos incluyen una cabecera de información acerca de los nodos exactos que deben atravesar. No requiere ningún tipo de mensajes periódicos (reactivo), disminuyendo así la sobrecarga con mensajes de control. Además ofrece la posibilidad de obtener, con la solicitud de una ruta, múltiples caminos posibles hacia el destino. Tampoco son un problema, a diferencia de la mayoría de protocolos de encaminamiento en este tipo de redes, los enlaces unidireccionales. Para poder realizar el encaminamiento en el origen, a cada paquete de datos se le inserta una cabecera DSR de opciones que se colocará entre la cabecera de transporte y la IP. Entre dichas opciones se incluirá la ruta que debe seguir el paquete nodo a nodo. Cada nodo mantiene una memoria caché de rutas en la que se van almacenando las rutas obtenidas a través de procesos de descubrimiento de rutas ya sean propias o obtenidas a través de escuchas en la red. En los procesos de descubrimiento de rutas se generan mensajes de solicitud, respuesta y error siendo estos mensajes 'route request', 'reply' y 'error' respectivamente.

- Ad Hoc On-Demand Distance Vector (AODV): En el protocolo AODV los nodos mantienen una tabla de encaminamiento para los destinos conocidos (empleando el algoritmo vector distancia). Inicialmente esta tabla estará formada por sus vecinos. Solamente se le añadirán destinos nuevos cuando sea necesario, es decir, cuando un nodo necesita comunicarse con otro que no está en su tabla, inicia un proceso de descubrimiento de ruta (reactivo) hacia el destino concreto. Para ello se emiten mensajes de descubrimiento de ruta 'rreq' que se van propagando entre todos los nodos de modo similar al DSR. En cambio, aquí los nodos generan una tabla de encaminamiento inversa para que puedan regresar las contestaciones 'rrep' a las solicitudes de ruta al nodo que la originó. Se recomienda el uso de mensajes 'hello' entre vecinos para determinar la conectividad, aunque para reducir el volumen de estos mensajes, sólo debe permitirse su envío a los nodos que estén transmitiendo datos. Debemos destacar además la utilización de las técnicas

de "búsqueda secuencial por anillos" y "reparación local del enlace" así como también que es capaz de proporcionar soporte multicast.

4.2 TECNOLOGÍA INALÁMBRICA

La comunicación vía radio usa el espectro electromagnético para enviar información. Cuando una corriente eléctrica pasa por un cable crea un campo electromagnético que envía ondas en todas direcciones, en forma parecida a la luz, que también es parte del espectro, pero a frecuencias mucho más altas. Con una antena y energía adicional es posible transmitir las señales a gran distancia. La frecuencia de onda puede modificarse de forma que las señales no interfieran entre sí, lo que permite aprovechar una mayor parte del espectro.

Las tecnologías inalámbricas pueden clasificarse en cinco grandes grupos, de acuerdo con la distancia que viaja cada tipo de señal. Primero están las comunicaciones satelitales, como el sistema de posicionamiento global (GPS, por sus siglas en inglés), formado por 24 satélites manejados por las fuerzas armadas de Estados Unidos, las cuales envían constantemente señales a dispositivos en tierra. Sin embargo, estas señales sólo viajan del satélite al aparato receptor

Otra categoría, y con señales de dos vías, están las tecnologías de telefonía celular de cobertura amplia como GSM y CDMA. Entre las versiones avanzadas de 'tercera generación' (3G) destacan HSDPA y LTE, desarrolladas por la industria de los celulares. Un contendiente prometedor es WiMax, tecnología basada en los estándares de Internet con respaldo de la industria informática.

Una tercera categoría incluye señales de menor alcance utilizadas para conectar dispositivos dentro de una habitación o un edificio, como los sistemas Wi-Fi para conectarse a Internet dentro de hoteles o aeropuertos, o Zigbee, protocolo de comunicaciones inalámbricas que sirve para interconectar sensores. Un avance reciente es la tecnología de banda ultra ancha (UWB), que utiliza frecuencias sumamente altas de cobertura muy limitada para transmitir grandes volúmenes de información, por ejemplo, para enviar un video desde un IPod o un dispositivo similar a un televisor.

En cuarto lugar están los protocolos para enlazar dispositivos en una "red de área personal" (PAN, personal area network). Por ejemplo Bluetooth, utilizado para enviar la señal del teléfono celular a un auricular inalámbrico.

El último tipo de comunicaciones son las que se dan cerca de una antena transmisora (NFC, near-field communications). En este caso, el dispositivo receptor debe estar cerca del sistema emisor, por ejemplo, al pasar por un edificio o en el transporte público. Una variante son las etiquetas de identificación por radiofrecuencia (RFID, por sus siglas en inglés), utilizadas por tiendas departamentales y otros usuarios. Cuando pasan por un lector, estas etiquetas envían la información que tienen almacenada. Estos sistemas de radio son tan diferentes entre sí como la luz lo es del sonido; así, los satélites no pueden rastrear etiquetas RFID, lo cual permite descartar riesgos para la intimidad.

Podemos ver como se distribuyen las diferentes tecnologías inalámbricas dependiendo de la velocidad de transmisión y de su utilización.

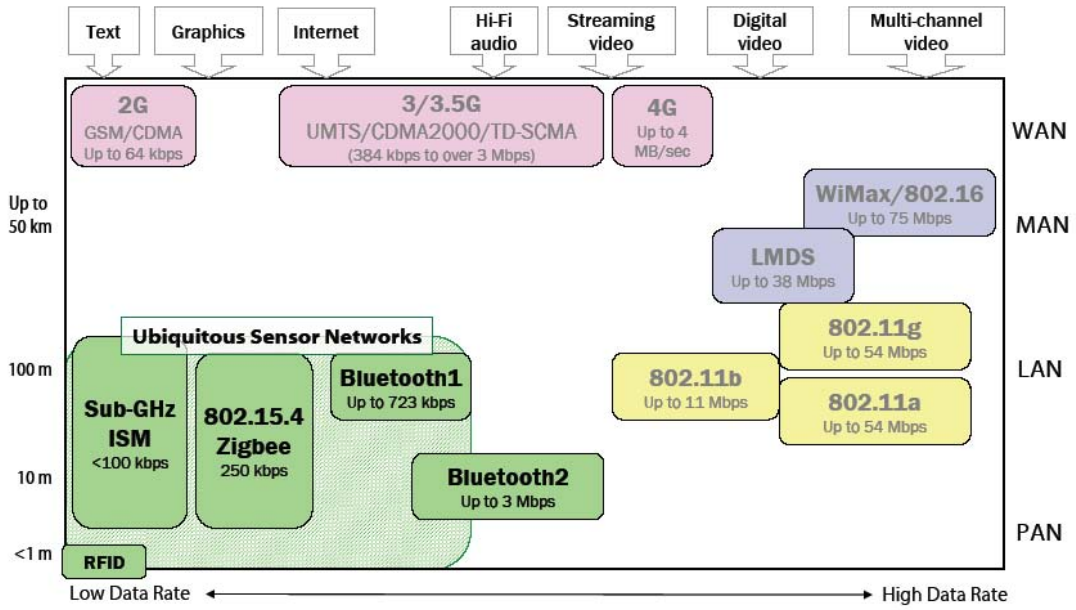

Figura 7 Tecnologías inalámbricas

También podemos estudiar éstas a través del coste que suponen comparado con la cobertura y la velocidad.

Tabla 5 Principales tecnologías inalámbricas de 2 vías. Fuentes William Webb Cambridge. Consultores OCDE, Pyramid Research, Nokia, CSR, Ember y Hitachi.

	Transferencia de datos por segundo	Cobertura	Costo en dólares (año 2008)
WiMax	15Mb	5 Km	8
Celular 3G	14 Mb	10 Km	6
Celular 2G	400 K	35 Km	5
Wi-Fi	54 Mb	50-100 m	4
Bluetooth	700 K	10 m	1
Zigbee	250 K	30 m	4
UWB	400 Mb	5-10 m	5
RFID	1-200 k	0.01-10 m	0.04

Claramente se ve que los protocolos más adecuados para ser usados en WSN son los protocolos Bluetooth y Zigbee. A pesar de que ZigBee es muy similar a Bluetooth podemos encontrar algunas diferencias que hacen más adecuado el protocolo Zigbee para las WSN:

- Una red ZigBee puede constar de un máximo de 65535 nodos distribuidos en subredes de 255 nodos, frente a los 8 máximos de una subred Bluetooth.
- Zigbee tiene un menor consumo eléctrico que el Bluetooth. En términos exactos, ZigBee tiene un consumo de 30mA transmitiendo y de 3µA en reposo, frente a los 40mA transmitiendo y 0.2mA en reposo que tiene el Bluetooth. Este menor consumo se debe a que el sistema ZigBee se queda la mayor parte del tiempo dormido, mientras que en una comunicación Bluetooth esto no se puede dar, y siempre se está transmitiendo y/o recibiendo.
- Zigbee tiene una velocidad de hasta 250 kbps, mientras que Bluetooth es de hasta 3 Mbps. Debido a las velocidades de cada uno, uno es más apropiado que el otro para ciertas cosas. Por ejemplo, mientras que el Bluetooth se usa para aplicaciones como los teléfonos móviles y la informática casera, la velocidad del ZigBee se hace insuficiente para estas tareas, desviándolo a usos tales como la domótica, los productos dependientes de una batería, los sensores médicos, y en artículos de juguetería, en los cuales la transferencia de datos es menor.

4.2.1 ZIGBEE

La relación entre IEEE 802.15.4-2003 y ZigBee es parecida a la existente entre IEEE 802.11 y Wi-Fi Alliance. La especificación 1.0 de ZigBee se aprobó el 14 de diciembre de 2004 y está disponible a miembros del grupo de desarrollo (ZigBee Alliance).
ZigBee utiliza la banda ISM para usos industriales, científicos y médicos; en concreto, 868 MHz en Europa, 915 en Estados Unidos y 2,4 GHz en todo el mundo. Sin embargo, a la hora de diseñar dispositivos, las empresas optan prácticamente siempre por la banda de 2,4 GHz, por ser libre en todo el mundo.
El desarrollo de la tecnología se centra en la sencillez y el bajo coste más que otras redes inalámbricas semejantes de la familia WPAN, como por ejemplo Bluetooth. El nodo ZigBee más completo, requiere en teoría cerca del 10% del hardware de un nodo Bluetooth o Wi-Fi típico; esta cifra baja al 2% para los nodos más sencillos. No obstante, el tamaño del código en sí es bastante mayor y se acerca al 50% del tamaño del de Bluetooth.
En 2006 el precio de mercado de un transceptor compatible con ZigBee se acercaba al dólar y el precio de un conjunto de radio, procesador y memoria rondaba los tres dólares. En comparación, Bluetooth tenía en sus inicios (en 1998, antes de su lanzamiento) un coste previsto de 4-6 dólares en grandes volúmenes; a principios de 2007, el precio de dispositivos de consumo comunes era de unos tres dólares.
La primera versión de la pila suele denominarse ahora ZigBee 2004. La segunda versión de junio de 2006 se denomina ZigBee 2006, y reemplaza la estructura MSG/KVP con una librería de clusters, dejando obsoleta a la anterior versión. ZigBee Alliance esta trabajando con la versión de 2007 de la pila para adecuarse a la última versión de la especificación, en concreto centrándose en optimizar funcionalidades de nivel de red (como agregación de datos). También se incluyen algunos perfiles de aplicación nuevos, como lectura automática, automatización de edificios comerciales y automatización de hogares en base al principio de uso de la librería de clusters.
La tecnología Zigbee es mas robusta que las otras tecnologías como puede verse en la figura.

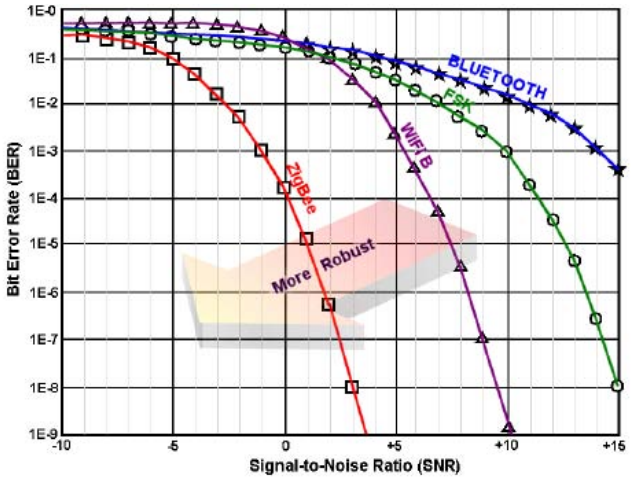

Figura 8 Comparativa de la robustez de las diferentes tecnologías inalámbricas

4.2.2 COMPATIBILIDAD ENTRE WIFI Y ZIGBEE

En multitud de aplicaciones es posible que estos dos sistemas de comunicación inalámbrica estén funcionando simultáneamente muy cerca, y se debe tener seguridad en que no se causaran interferencias entre sí.
En la grafica siguiente podemos ver cuanta energía y en que frecuencias opera la radio en las dos tecnologías inalámbricas. No se emite en un solo canal sino que ocupan varias bandas de frecuencia. La asignación del canal de radio se hace en el centro de la banda de frecuencias (en el centro de la joroba). Podemos observar que el canal WiFi es más ancho que el canal Zigbee, lo que quiere decir que WiFi ocupa más espectro de radio frecuencia que Zigbee.

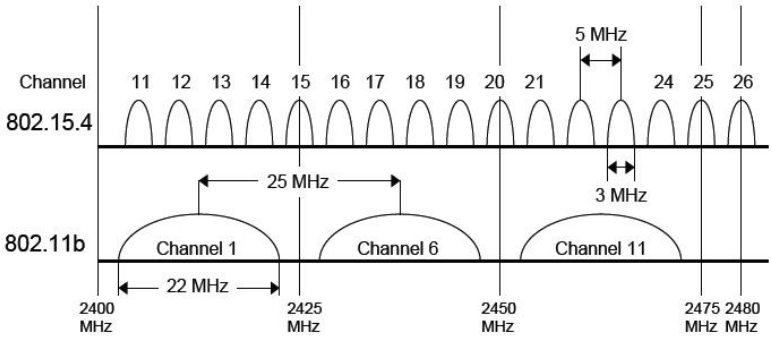

Figura 9 Canales IEEE802.15.4 y IEEE802.11

5 APLICACIONES

Las redes de sensores inalámbricas son colecciones de nodos de tamaño reducido con capacidad computacional y de un precio no muy elevado que pueden medir localmente condiciones medioambientales para luego enviar esta información a un nodo central o estación base. Las WNS pueden soportar un amplio rango de aplicaciones.
A pesar de que la mayor parte de los sensores que podemos encontrar hoy en día son todavía conectados a través de un cable, los inalámbricos ofrecen ventajas significativas sobre los cableados, como reducir el coste del sistema y hacer posible aplicaciones de otra forma imposibles de realizar. Sin embargo el mercado está fragmentado, debido sobre todo a que cada aplicación desarrollada tiene unas características únicas.
Las WNS pueden dividirse en dos tipos de redes, las redes multisalto y las monosalto. Las redes monosalto son las redes que conectan directamente con la estación base y como su propio nombre indica, la información es emitida desde el nodo directamente a la estación base. Y las redes multisalto, son las que la información es retransmitida por varios nodos antes de llegar a la estación base.
El rango potencial de aplicaciones está realmente únicamente limitado por la imaginación ya que la convergencia de las tecnologías de información y comunicaciones inalámbricas, con técnicas de miniaturización, han convertido a las WSN en un área con una capacidad de crecimiento elevada.

En este apartado se realiza un estudio de varias de las aplicaciones más destacadas dentro de este área, aunque sea un campo ilimitado y podamos encontrar aplicaciones en infinidad de casos como: monitorización de estructuras y construcciones, sistemas de defensa, aplicaciones biomédicas, control de fronteras, detección de terremotos, monitorizaciones agrarias, monitorización y actuación industrial, control de almacenes, detección de gases, monitorización del tiempo, etc.

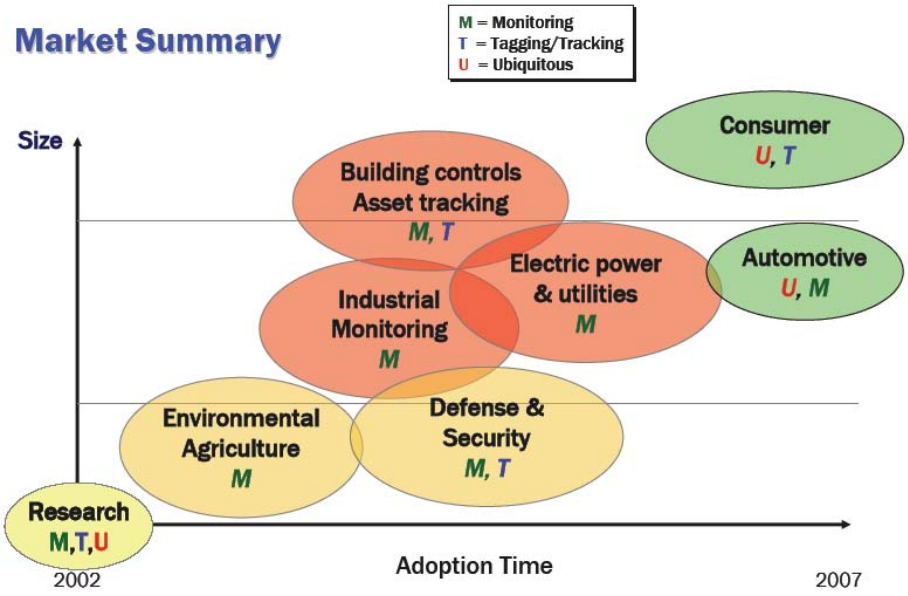

Figura 10 Distribución del mercado

Se puede ver en la gráfica que lo que empezó como investigación en el 2002 poco a poco ha ido extendiéndose dentro del mercado. Cada vez es mayor la cantidad de redes y las aplicaciones en las que se utiliza.

5.1 AUTOMOCIÓN

Con las características de las WSN, los coches podrán pronto estar disponibles para hablar unos con otros y con infraestructuras dentro de carreteras y autopistas. Los sensores pueden aplicarse en las ruedas del vehículo para dar asistencia al conductor y avisar de posibles mensajes de alerta.

Por ejemplo durante un frenazo de emergencia, un mensaje de emergencia desde el coche que frena puede enviarse a todos los coches cercanos para que éstos tomen medidas respecto a este evento. Otra aplicación interesante es la recogida de datos de tráfico en tiempo real. La información de la que puede disponer un coche que venga en dirección contraria puede ser valiosa. Un vehículo puede recibir información de otro vehículo sobre dónde están situados los atascos de tráfico, la velocidad y densidad del tráfico o información de sensores fijos.
Toda esta información puede ser transmitida de vehículo a vehículo para evitar atascos y planificar rutas alternativas.

5.1.1 PROYECTO CAPSTONE DE FORD

Con el 70% de los vehículos de la compañía Ford siendo remodelados es importante estar al día y conocer las nuevas opciones y lanzamientos. Ford esta mirando nuevas maneras de innovar a la hora de recoger medidas de sus vehículos. Las redes de sensores están incrementando su inteligencia y además pueden obtener datos, no sólo limitados a luz, temperatura, humedad y movimiento.
Un equipo de la universidad de Michigan, junto con Ford, está diseñando y desarrollando una manera de identificar cuantas veces un vehículo es inspeccionado por un comprador potencial. Esto podría incluir cuantas veces una puerta es abierta, el capo fue levantado o el maletero fue inspeccionado. Los eventos estudiados podrían extenderse también a cuando una persona ocupó un asiento y cuanto tiempo perduró en él.
El entorno por definición, puede ser muy dinámico, haciendo de la posibilidad de conectarse en una red en malla crítico a largo plazo. [FORD2008]

5.2 CONTROL DOMÓTICO DE UN EDIFICIO

Aplicaciones de este tipo en el control de una oficina o una casa hacen que el tiempo de permanecía dentro de ésta sea mucho más agradable para el ser humano.
El uso de sensores empotrados puede reducir ampliamente los costes de una monitorización de una construcción donde tener un conocimiento de la temperatura y de la luz para poder regular los sistemas de calefacción y aire acondicionado, así como las luces.

5.2.1 SISTEMA INTELIGENTE DE CONTROL DE LUZ

Un control eficiente de la energía dedicada al alumbrado de comercios puede llegar a ahorrar un 40% del consumo de energía. A pesar de este ahorro estos sistemas no están nada extendidos en este tipo de construcciones.
La meta de esta investigación era reducir el consumo de energía así como crear un entorno de luminosidad ante las preferencias del usuario. Se optó por utilizar WSN ante el mayor precio y menor flexibilidad de un sistema cableado centralizado [WEN2006].

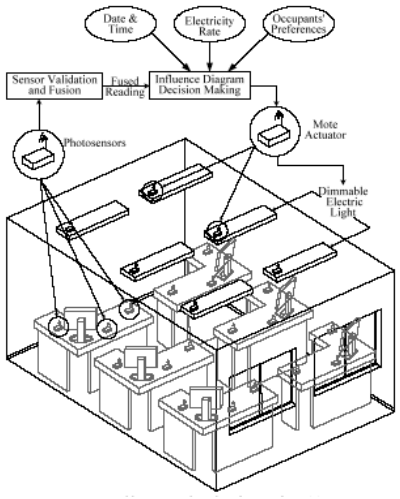

Figura 11 Sistema de iluminación inteligente en oficinas

5.3 MONITORIZACIÓN AMBIENTAL

La monitorización ambiental es una de las primeras aplicaciones donde se utilizaron redes de sensores inalámbricas. Nos permite poder distribuir una gran cantidad de sensores en un espacio natural y obtener una gran cantidad de datos que de otra forma es imposible de obtener con la instrumentación tradicional.

5.3.1 ISLA GREAT DUCK

Los sensores pueden ser usados para monitorizar condiciones y movimientos de animales que vivan en estado salvaje, ya que con esta tecnología no hace falta una presencia de los seres humanos y así se molesta lo mínimo posible en su rutina. Los sensores pueden monitorizar también la calidad del aire, el nivel de polución y para detectar posibles amenazas biológicas o químicas a partir de pequeñas señales.
Un ejemplo de este tipo de aplicación es el llevado a cabo por un equipo de ingenieros de la Universidad de Berkeley en colaboración con el College of the Atlantic en Bar Harbor y el laboratorio de investigación de Intel, en la isla Great Duck en la costa del estado de Maine en el noreste de los Estados Unidos, donde se trata de conservar la fauna mediante la detección de intrusos, ya sean humanos o algún otro tipo de depredador.
De esta manera es posible que un grupo de biólogos pueda observar la actividad de las aves en dicha isla desde sus oficinas mediante un enlace que permita la comunicación de la red de sensores y su red de ordenadores. Los parámetros que son medidos en este experimento son la cantidad de luz existente en los nidos, la temperatura y la humedad dependiendo de la presencia de aves o no [POL2002].

Figura 12 Monitorización del hábitat en Great Duck

5.3.2 MONITORIZACIÓN DE UN GLACIAR

Para entender el cambio climático que involucra un cambio del nivel del mar debido al calentamiento global, es importante entender como los glaciares contribuyen en la liberación de agua fresca al mar. Esto puede causar altos crecimientos del nivel del mar y grandes cambios en la temperatura y por lo tanto en las mareas marinas. El comportamiento de un glaciar y sobre todo de su movimiento puede llevarnos a predecir cambios futuros en su comportamiento.
Durante el verano de 2004 fue estudiado el comportamiento del glaciar Briksdalsbreen en Noruega por medio de redes de sensores inalámbricas. El objetivo fue entender la dinámica del glacial con respecto al cambio climático [MAR2004].

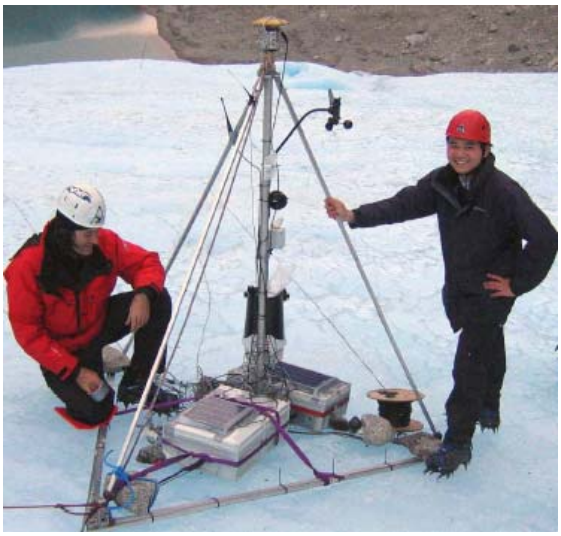

Figura 13 Estación base del sistema de monitorización de un glaciar

5.4 CONTROL DE ALMACENES

Los sensores inalámbricos pueden ser usados para monitorizar y gestionar los materiales dentro de un almacén o en el transporte desde el almacén al destino final.

5.4.1 CONTROL DE TRANSPORTE Y LOGÍSTICA

Las WSN son aplicables también al control de transporte y logística de mercancías, en este caso los nodos controlarán todas las actividades chequeando los posibles errores que pueden ocurrir en el proceso de entregado y distribución de bienes.
Evidentemente la cantidad de sensores dependerá del producto a transportar, pero en el estudio realizado éstos se sitúan en los útiles retornables como pueden ser palets, que una vez hecha la entrega deben ser devueltos al proveedor.

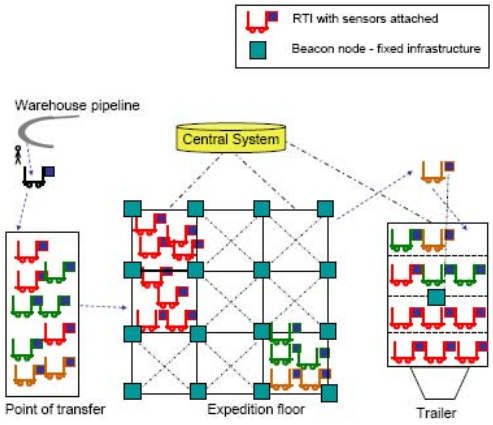

Figura 14 Diagrama del proceso de control de transporte y logística

Cada trailer puede distribuir 60 útiles retornables, cada uno de estos llevara 5 sensores midiendo diferentes magnitudes físicas, dependiendo del contenido que se distribuya [EVER2005].

5.5 CUIDADO DE LA SALUD

Antiguas formas de realizar cuidados médicos pueden verse beneficiadas usando WSN que monitoricen las señales vitales de los pacientes, enviando dicha información hasta las oficinas de los médicos o

simplemente avisando al paciente si alguna de sus señales cae drásticamente. En estos casos también puede usarse como una red personal que sirva para entender los movimientos y el comportamiento de las personas.
El cuidado de personas mayores requiere en la mayor parte de los casos de un seguimiento exhaustivo de sus actividades, lo cual limita su privacidad y al mismo tiempo supone una excesiva carga de trabajo para los cuidadores. Mediante el uso de una red de sensores inalámbricos situados en puntos estratégicos del domicilio del anciano, así como en objetos de uso cotidiano, los cuidadores pueden monitorizar en tiempo real el comportamiento de las personas mayores, evitando la realización de tareas tediosas y centrándose en aspectos más importantes como es la mejora de su calidad de vida.
Adicionalmente, el cuidado médico tanto en hospitales como fuera de los mismos, por ejemplo la rehabilitación de pacientes, también se beneficia del uso de esta tecnología. Un ejemplo de ello es el proyecto CodeBlue, desarrollado en la Universidad de Harvard. En este caso se han implementado distintos tipos de sensores para la monitorización de parámetros vitales: tasa de latidos del corazón, concentración de oxígeno en sangre, datos EKG de electrocardiograma, etc. Toda esta información se recoge por los sensores y se distribuye de forma inalámbrica a una PDA u ordenador portátil para su procesamiento. De este modo, cualquier señal de alerta puede detectarse a distancia en tiempo real.

5.5.1 CUIDADO DE ANCIANOS CON ALZHÉIMER

Un porcentaje cada vez mayor de la creciente cantidad de población anciana está viviendo con problemas de perdida de memoria. Los gastos que necesitan estas personas son elevados ya que tienen que estar vigilados las 24 horas del día.
El Intel Proactive Health Research junto con Intel Research Seattle y la Universidad de Washington están estudiando estos casos. Se trata de controlar el entorno de la persona mediante tecnologías inalámbricas, ya sea controlando los movimientos del paciente como por medio de tecnología de transmisión de imágenes [MOR2003].

5.6 CONTROL DE PROCESOS INDUSTRIALES

En el entorno industrial el uso de sensores es algo común y para poder acceder a zonas donde no podemos usar cableado, las WSN son la herramienta adecuada. Posibles aplicaciones en este entrono son: telemetría en plantas, perdidas de calidad, diagnostico de maquinaria, monitorización,…

5.6.1 CONTROL DE UN TANQUE DE PETRÓLEO EN UNA PLATAFORMA PETROLÍFERA (BP)

El proyecto de BP desarrolló un nuevo sistema de mantenimiento predictivo capaz de monitorizar maquinaria rotativa crítica como bombas y motores dentro de la plataforma Loch Rannoch, captando datos vibratorios para evaluar las condiciones de operación y enviando los datos mediante comunicación inalámbrica.
Los datos vibratorios pueden darnos la información de cómo una máquina está desgastándose y nos ayudan a saber cuando debemos hacerle el mantenimiento correspondiente.

Figura 15 Maquinaria rotativa en una plataforma petrolífera

Las medidas fueron observadas por 150 acelerómetros de la casa Rockwell y la información fue gestionada por motes de la casa Crossbow [ECO2003].

5.7 APOYO MILITAR

El apoyo militar fue el primer propósito por el cual empezó a investigarse esta área. Tener conocimiento en tiempo real del campo de batalla es esencial para el control, las comunicaciones y la toma de decisiones. La red de sensores puede ser rápidamente desarrollada sin tener que realizar una infraestructura como puede suceder en otros casos como un radar, y pasar a recoger información inmediatamente.

5.7.1 VIGILANCIA DE FRONTERAS

Boeing Co. Consiguió un contrato en 2006 del Department of Homeland Security de los Estados Unidos para implementar SBInet (tthe Secure Borders Initiative) a través de las fronteras norte y sur. El programa trataba de desarrollar una infraestructura tecnológica que a través del uso de varios sensores y aparatos de detección lograba a través del satélite Ku-band enviar la información a un centro remoto de operaciones donde era gestionada.

5.8 AGRICULTURA Y GANADERÍA

La agricultura y la ganadería son dos de las áreas donde puede ser importante esta tecnología, ya que una situación al aire libre donde monitorizar condiciones que ayuden a la mejora de producción y calidad en una producción agrícola o el control de cabezas de ganado en continuo movimiento puede ser muy difícil con la tecnología tradicional.

5.8.1 VIÑEDOS CAMALIE

Los viñedos Camalie, en Estados Unidos, tienen uno de los sistemas más avanzados de medida de la humedad del suelo. Usan la tecnología inalámbrica desarrollada por la Universidad de Berkeley en colaboración con Intel y comercializada por Crossbow.
La aplicación consiste en optimizar la irrigación, reduciendo el consumo de agua, la energía utilizada en el bombeado y mejorando la calidad de la uva. Se proporciona una monitorización del sistema de irrigación, mostrando fallos que pueden un impacto sustancial a algo plazo.

Figura 16 Dispositivos implantados en viñedos

Una vez implantado el sistema se comprobó un importante crecimiento en la producción, además de un descenso en el consumo de energía a la hora de usar las instalaciones [CAM2008].

5.8.2 SEGUIMIENTO DE RUTINAS EN GANADO PORCINO

El doctor Philippe Bonnet en la Universidad de Copenhague desarrollo una aplicación pensada para facilitar el trabajo de los veterinarios, controlando varias variables en la rutina diaria de los cerdos en una granja.

Figura 17 Seguimiento de rutina en ganado porcino

5.9 SEGURIDAD Y VIGILANCIA

Un importante campo de aplicaciones es la monitorización de seguridad y vigilancia en edificios, aeropuertos, metros, u otras infraestructuras criticas como redes de energía y telecomunicaciones o como autopistas. Sensores de captación de imágenes y videos pueden ser usados para identificación o seguimiento de posibles objetivos móviles, a pesar de que requieren un alto ancho de banda a la hora de la comunicación de datos.

5.9.1 RED DE VIGILANCIA MEDIANTE VIDEO

Los sistemas de vigilancia mediante cámaras pueden ser combinados con las redes inalámbricas de sensores, teniendo en cuenta que el consumo de energía será más elevado de lo normal. No solo debe gastar energía al hacer que la cámara capte la imagen, sino que la transmisión de todas estas imágenes a través de la red también tiene un consumo muy elevado [PRA2005].

Figura 18 Imote2 con cámara integrada

5.10 CONTROL DEL TRÁFICO

Las redes de sensores son el complemento perfecto a las cámaras de tráfico, ya que pueden informar de la situación del tráfico en ángulos muertos que no cubran las cámaras y también pueden informar a los conductores de una situación, en caso de atasco o accidente, que permita a estos tener la capacidad de reacción para tomar rutas alternativas.

5.10.1 DISTRIBUCIÓN DEL TRÁFICO MEDIANTE REDES AD-HOC EN VEHÍCULOS

Los vehículos son considerados como un área para desarrollar aplicaciones sobre WSN. El objetivo de la investigación relatada es crear una gran red de sensores usando los vehículos como portadores de los sensores, además de sus proveedores de energía. Estos nodos irían transmitiendo información relativa a la situación, velocidad, dirección..., que permitirían poder realizar un control del tráfico en diferentes situaciones.

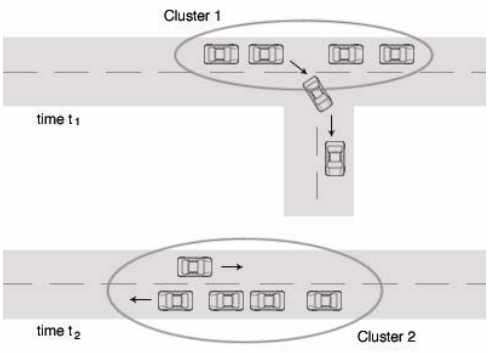

Figura 19 Gestión del tráfico

Además se tendrían varias puertas de enlace fijas que captarían información también, llegando a actuar en cruces semafóricos [THO2004].

5.11 MONITORIZACIÓN DE ESTRUCTURAS

Una gran aplicación de las WSN es el monitoreo de estructuras que sobre todo se ha llevado a cabo en Estados Unidos y Canadá. Aquí se estima que tienen unos 25 trillones de dólares invertidos en estructuras civiles y ante tanta inversión se desea tener un control de las estructuras realizadas. La tecnología utilizada se llama SHM, del ingles Structural Health Monitoring, y se trabaja con la identificación y el monitoreo de comportamientos extraños. Éstos pueden ser una falla en una estructura como puentes, edificios u otras estructuras.

5.11.1 CONTROL DE VIBRACIONES DE UN PUENTE

Gracias a los nuevos sistemas MEMS y a que podemos tener sensores de aceleración que puedan medir inalámbricamente, podemos controlar las vibraciones en construcciones. La universidad de Berkeley en California realizó un estudio sobre un puente para peatones sobre la autopista I-80 en Berkeley.
Esta aplicación da información sobre el estado de vida de la estructura, así como eventos que se produzcan en ella durante la monitorización, como puede ser un terremoto [SHA2007].

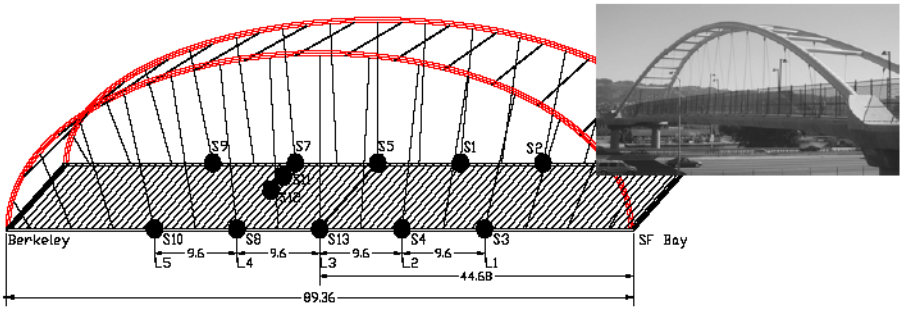

Figura 20 Localización de sensores sobre el puente

5.11.2 MONITORIZACIÓN DE LA SALUD DE UNA INFRAESTRUCTURA CIVIL (PUENTE GOLDEN GATE)

Otro ejemplo de aplicaciones SHM es el llevado a cabo en el puente Golden Gate en San Francisco, donde los nodos tienen la finalidad de monitorizar en varias partes de la estructura las vibraciones producidas, ya sean por el paso de vehículos o por las condiciones atmosféricas.
Se implantaron 64 nodos en un sistema de 46 saltos que median las vibraciones ambientes con una exactitud de 30□ G. Las vibraciones ambientes eran muestreadas a 1kHz con un tiempo de exposición menor de 10□ s [SUK2007].

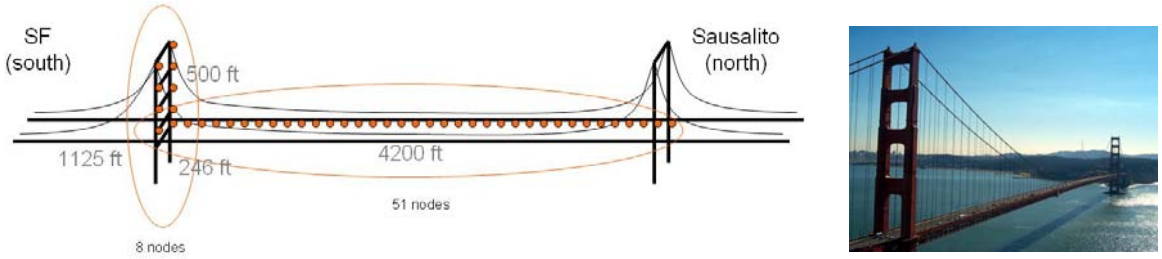

Figura 21 Localización de sensores sobre el Golden Gate

6 MERCADO DE LAS WSN

En el año 1998 Crossbow Technologies [XBOW2008], el mayor fabricante de nodos sensores de la actualidad y el mayor proveedor dentro de los principales grupos de investigación, puso en el mercado el primer modelo de nodo sensor o 'mote', iniciando así una evolución que continua actualmente. Con estos nodos como hardware de referencia, la Universidad de Berkeley tiene la iniciativa más importante y la que más repercusión ha tenido: el desarrollo en el año 2003 de un sistema operativo capaz de gestionar los recursos de las motes y permitir la ejecución de aplicaciones. Este sistema operativo, denominado TinyOS es considerado en la actualidad el estándar de facto. Probablemente el éxito de TinyOS se debe a que ofreció una referencia común a los investigadores, de manera que facilitó oportunidades para que otros pudieran comenzar a trabajar y por tanto dando cohesión a las distintas soluciones.
Podemos ver la evolución del mercado según ON World, In-stat, WTRS y Harbor Research.

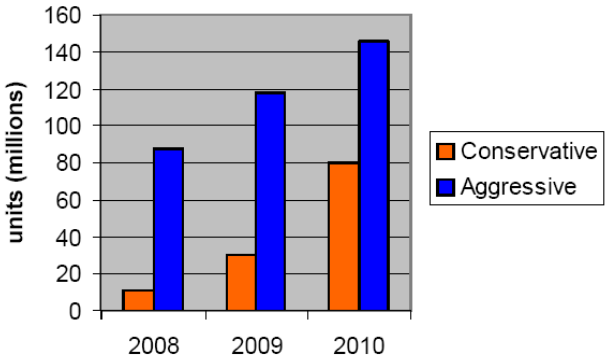

Figura 22 Evolución del mercado

Según un estudio realizado por On World, el mercado de redes de sensores inalámbricas esta experimentando un alto crecimiento en la demanda, porque cada vez más empresas quieren instalar este tipo de red.

El estudio de ON World se llama "Wireless Sensor Networks: Growing Markets, Accelerating Demand" [HAT2005], y está basado en los resultados de una encuesta realizada con 147 empresas. Incluye predicciones sobre producción e ingresos en ocho mercados de redes de sensores inalámbricas, además de análisis y resultados sobre motivación del sector, avances en las aplicaciones actuales de redes de sensores y las últimas tendencias del mercado.
Según la directora de investigación de ON World, durante 2005 se utilizarán al menos un millón de nodos de redes de sensores inalámbricas. Además, las redes de 500 nodos o más son cada vez más frecuentes y existen ya varios casos de miles de nodos en una sola red.
El informe concluye que existe en la actualidad una demanda 'abrumadora' de soluciones ofrecidas por redes de sensores inalámbricas por parte de empresas industriales. De las empresas sondeadas para el estudio, 29 por cien utilizan WSN mientras que más de 40 por cien consideran muy probable la posibilidad de que realicen pruebas piloto con redes de sensores inalámbricas dentro del próximo año y medio.
A pesar del gran futuro de esta tecnología, todavía hay barreras que hacen que muchas empresas no opten por ella.

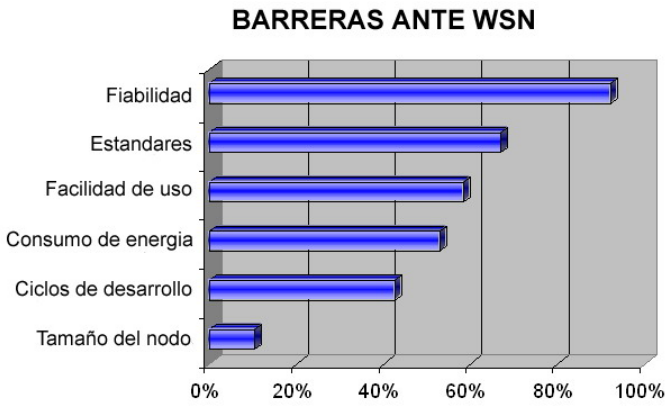

Figura 23 Barreras adopción de WSN. OnWord 2005

6.1 CASAS DE DISPOSITIVOS PARA WSN

Las principales casas que disponen de tecnología para redes de sensores inalámbricas son las siguientes.

6.1.1 CROSSBOW

La tecnología desarrollada por Crossbow ha estado a la vanguardia de la tecnología de sensores inteligentes durante más de una década y ha enviado cientos de miles de sensores inteligentes a más de 4000 clientes a lo largo del mundo. Hoy, Crossbow es el líder en cuanto a tecnología de sensores inalámbricos.

Crossbow http://www.xbow.com/

Especializada en el mundo de los sensores, es una empresa que desarrolla plataformas hardware y software que dan soluciones para las redes de sensores inalámbricas. Entre sus productos de módulos inalámbricos podemos encontrar las plataformas Mica, Mica2, Micaz, Mica2dot, telos, telosb, Iris e Imote2. Dispone también de gran cantidad de módulos gateway y placas sensoras.
En capítulos posteriores se estudian aplicaciones realizadas con estas tecnologías y más concretamente son los módulos inalámbricos de la gama MicaZ e Imote2 y los gateways MIB600 Ethernet Gateway, SPB400 Stargate Gateway, NB100 Stargate Netbridge y IIB2400 Imote2 Interface.

6.1.2 SENTILLA

Otra de las empresas dedicadas a las redes de sensores inalámbricas es Sentilla, también llamada anteriormente MoteIV. Es la encargada de los motes Tmote Sky, diseñados también por la Universidad de Berkeley y preparados para ser usados por TinyOS. Fue Joseph Polastre, antiguo doctorando de un grupo de trabajo de esta universidad quien formó la compañía MoteIV. Ha desarrollado la plataforma Tmote Sky y Tmote Invent.

sentilla http://www.sentilla.com/

6.1.3 SHOCKFISH

Shockfish SA ha desarrollado la plataforma TinyNode pensando en aplicaciones industriales. La misión de esta empresa es hacer de puente entre la investigación académica y el mundo laboral de la industrial en el entorno de redes de sensores inalámbricos.

shockfish / http://www.shockfish.com/

La filosofía de diseño de TinyNode es suministrar una plataforma tanto para proyectos académicos como para aplicaciones industriales. El módulo del núcleo de TinyNode 584 es un nodo sensor de bajo consumo y dispone de conectividad, almacenamiento y alimentación.

6.1.4 BTnode

Los módulos fabricados por BTnode han sido desarrollados por el ETH Zurich, conjuntamente por el Computer Engineering and Networks Laboratory (TIK) y el Research Group for Distributed Systems.

BTnodes http://www.btnode.ethz.ch/

Actualmente, los BTnodes son usados en dos grandes proyectos de investigación, apoyando a la plataforma de desarrollo inicial, estos son NCCS MICS y Smart-Its.

6.1.5 EMBER

Ember es uno de los promotores de la Zigbee Alliance y las soluciones propuestas por esta empresa cumplen la capa física según el estándar IEEE 802.15.4.

ember http://www.ember.com/

La tecnología Ember basada en Zigbee es la adecuada para aplicaciones de redes de sensores escalables que requieran una implementación en malla de bajo consumo como automatizaciones en edificios, entre otras.

6.1.6 SUN

Sun SPOT (Sun Small Programmable Object Technology) es un mote para WSN desarrollado por Sun Microsystems. El aparato esta construido bajo la normativa estándar IEEE 802.15.4. Al contrario de otros nodos inalámbricos, el Sun SPOT esta construido bajo la maquina virtual Java Squawk.

Sun microsystems http://www.sunspotworld.com/

6.1.7 Nano-RK

La casa Nano-RK ha desarrollado la plataforma FireFly como una plataforma de nodos inalámbricos de bajo consumo y bajo coste, con la idea de dar el mejor soporte en aplicaciones en tiempo real.

http://www.nanork.org/

6.2 COMPARATIVA DE NODOS INALÁMBRICOS

En la siguiente tabla podemos comparar las principales características de varios de los nodos inalámbricos de las marcas comentadas anteriormente.

Tabla 6 Comparativa de nodos inalámbricos

Nombre	Casa	Microcontrolador	Transmisor	Memoria Programas y datos	Memoria externa	Programación	Comentarios
EM250	Ember	12MHz XAP2b 16-bit	2.4GHz IEEE 802.15.4 Compliant Transciever	5kB RAM	128kB Flash	C	
Mica	Crossbow	Atmel ATMEGA103 4 MHz 8-bit CPU	RFM TR1000 radio 50 kbit/s	128+4K RAM	512K Flash	nesC Programming	TinyOS Support
Mica2	Crossbow	ATMEGA 128L	Chipcon 868/916 MHz	4K RAM	128K Flash		TinyOS, SOS and MantisOS Support
Mica2Dot	Crossbow	ATMEGA 128		4K RAM	128K Flash		
MicaZ	Crossbow	ATMEGA 128	TI CC2420 802.15.4/ZigBee compliant radio	4K RAM	128K Flash	nesC	TinyOS, SOS, MantisOS and Nano-RK Support
Telos	Crossbow	Motorola HCS08		4K RAM			
TelosB	Crossbow	Texas Instruments MSP430 microcontroller	250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver	10k RAM	48k Flash		Contiki, TinyOS, SOS and MantisOS Support
T-Mote Sky	Sentilla	Texas Instruments MSP430 microcontroller	250 kbit/s 2.4 GHz IEEE 802.15.4 Chipcon Wireless Transceiver	10k RAM	48k Flash		Contiki, TinyOS, SOS and MantisOS Support
IMote	Intel	ARM core 12 MHz	Bluetooth with the range of 30 m	64K SRAM	512K Flash		TinyOS Support
IMote 1.0	Intel	ARM 7TDMI 12-48 MHz	Bluetooth with the range of 30 m	64K SRAM	512K Flash		TinyOS Support
IMote 2.0	Crossbow	Marvel PXA271 ARM 11-400 MHz	TI CC2420 802.15.4/ZigBee compliant radio	32MB SRAM	32MB Flash		Microsoft .NET Micro, Linux, TinyOS Support

Nombre	Casa	Microcontrolador	Transmisor	Memoria Programas y datos	Memoria externa	Programación	Comentarios
Iris	Crossbow	ATmega1281	Atmel AT86RF230 802.15.4/ZigBee compliant radio	8K RAM	128K Flash		nesC TinyOS, MoteWorks Support
SunSPOT	Sun Microsystems	ARM 920T	802.15.4	512K RAM	4 MB Flash		Java Squawk J2ME Virtual Machine
FireFly	Nano-RK	Atmel ATmega 1281	Chipcon CC2420	8K RAM	128K FLASH ROM, 4K EEPROM	C Programming	Nano-RK RTOS Support
BTNode rev.3	BTNode	Atmel ATmega 128L	Bluetooth subsystem: Zeevo ZV4002, supporting AFH/SFH	64+180 Kbyte RAM	128 Kbyte FLASH ROM, 4 Kbyte EEPROM	Standard C Programming	TinyOS Support
eyesIFXv2.1	BTNode	Texas Instruments MSP430F1611	Infineon TDA5250	10 KB RAM	48 KB FLASH		

7 SISTEMAS OPERATIVOS

Las necesidades que tiene un nodo de una WSN son totalmente distintas a las que pueda tener cualquier otro dispositivo como puede ser un PC, por lo tanto estos nodos tienen sus propios sistemas operativos.
Los sistemas operativos para WSN son típicamente menos complejos que los de propósito general, tanto debido a los requisitos especiales de las aplicaciones en las que se usan, como a las restricciones de recursos encontradas en las plataformas hardware utilizadas. Por ejemplo, las aplicaciones de WSN no son interactivas como son las aplicaciones para PC y debido a esto, estos sistemas no necesitan incluir el soporte de interface de usuario. Además, las restricciones de los recursos en términos de memoria hacen imposible de implementar los mecanismos de memoria virtual.
El hardware de la redes inalámbricas de sensores no es muy diferente al de sistemas empotrados tradicionales y por lo tanto es posible utilizar sistemas como eCos o uC/OS. Sin embargo, estos sistemas están diseñados para usar operaciones en tiempo real. A diferencia de los tradicionales sistemas operativos para sistemas empotrados, los sistemas desarrollados para redes de sensores inalámbricas no tienen como objetivo apoyar operaciones en tiempo real.
TinyOS es quizás el primer sistema operativo diseñado específicamente para redes de sensores inalámbricas. A diferencia de la mayoría de los otros sistemas operativos, TinyOS se basa en un modelo de la programación controlado por eventos en vez de multiprocesos. Los programas de TinyOS están compuestos por eventos y tareas guiadas. Cuando un evento externo ocurre, como puede ser la entrada de un paquete de datos o la lectura de un sensor, TinyOS llama al evento apropiado y lo ejecuta. El lanzador de eventos puede posponer tareas durante cierto tiempo. Tanto TinyOS como los programas escritos para él son escritos en un lenguaje de programación especial llamado nesC, que es una extensión del lenguaje de programación C. NesC esta diseñado para determinar las prioridades entre tareas y eventos.
Hay también sistemas operativos que permiten programar en C. Por ejemplo Contiki, MANTIS, BTnut, SOS y Nano-RK. Contiki esta diseñado para soportar la carga de módulos a través de la red y para soportar cargas de ficheros ELF. El kernel de Contiki esta basado en lanzamiento de eventos, como TinyOS, pero puede llegar a soportar multitareas básicas. A diferencia del kernel de Contiki basado en eventos, los kernels MANTIS y Nano-RK están basados en multitareas preventivas. El kernel divide el tiempo entre los procesos activos y decide que proceso debe ser ejecutado para hacer la programación de la aplicación más fácil. Nano-RK tiene un kernel basado en tiempo real que permite controlar la manera de acceder de las tareas, a la CPU, a la red y a las placas sensoras. Como TinyOS y Contiki, SOS es un sistema operativo basado en eventos. La principal característica de SOS es su soporte para módulos cargables. Un sistema complejo se construye a partir de módulos más pequeños, probablemente en tiempo de ejecución. Para soportar el dinamismo inherente en su modulo de interfaz, SOS soporta una gestión de la memoria dinámica. BTnut se basa en multitareas cooperativas y se programa en código C plano, además viene empaquetado con un kit de desarrollo y manuales.

7.1 TINYOS

TinyOS [LEV2005] es un sistema operativo orientado a trabajar con redes de sensores, desarrollado en la Universidad de Berkeley. TinyOS puede ser visto como un conjunto de programas avanzados, el cual cuenta con un amplio uso por parte de comunidades de desarrollo, dada sus características de ser un proyecto de código abierto (Open Source). Este 'conjunto de programas' contiene numerosos algoritmos, que nos permitirán generar enrutamientos, así como también aplicaciones pre-construidas para sensores. Además soporta diferentes plataformas de nodos de sensores, arquitecturas bases para el desarrollo de aplicaciones.
El lenguaje en el que se encuentra programado TinyOS es un meta-lenguaje que deriva de C, cuyo nombre es NesC [GRA2003]. Además existen varias herramientas que ayudan el estudio y desarrollo de aplicaciones para las redes de sensores, que van desde aplicaciones para la obtención y manejo de datos, hasta sistemas completos de simulación.
El diseño de TinyOS está basado en responder a las características y necesidades de las redes de sensores, tales como reducido tamaño de memoria, bajo consumo de energía, operaciones de concurrencia intensiva, diversidad en diseños y usos, y finalmente operaciones robustas para facilitar el desarrollo

confiable de aplicaciones. Además se encuentra optimizado en términos de uso de memoria y eficiencia de energía.
El diseño del Kernel de TinyOS está basado en una estructura de dos niveles de planificación.

- Eventos: Pensados para realizar un proceso pequeño (por ejemplo cuando el contador del timer se interrumpe, o atender las interrupciones de un conversor análogo-digital). Además pueden interrumpir las tareas que se están ejecutando.
- Tareas: Las tareas son pensadas para hacer una cantidad mayor de procesamiento y no son críticas en tiempo (por ejemplo calcular el promedio en un arreglo). Las tareas se ejecutan en su totalidad, pero la solicitud de iniciar una tarea, y el término de ella son funciones separadas.

Con este diseño permitimos que los eventos (que son rápidamente ejecutables), puedan ser realizados inmediatamente, pudiendo interrumpir a las tareas (que tienen mayor complejidad en comparación a los eventos).
El enfoque basado en eventos es la solución ideal para alcanzar un alto rendimiento en aplicaciones de concurrencia intensiva. Adicionalmente, este enfoque usa las capacidades de la CPU de manera eficiente y de esta forma gasta el mínimo de energía.
TinyOS se encuentra programado en NesC, un lenguaje diseñado para reflejar las ideas propias del enfoque de componentes, incorporando además un modelo de programación que soporta concurrencia, manejo de comunicaciones y fácil interacción con el medio (manejo de hardware).

7.2 LINUX

Linux es un sistema operativo tipo Unix que se distribuye bajo la Licencia Pública General de GNU (GNU GPL), es decir que es software libre. Su nombre proviene del kernel de Linux, desarrollado en 1991 por Linus Torvalds.
Hablar de Linux es sólo referirse al Kernel, el núcleo del sistema. El núcleo sólo es una interfaz que permite comunicar el hardware con los programas. Por lo que el Kernel solo no forma el sistema operativo.
Un sistema GNU/Linux empotrado simplemente hace referencia a un sistema empotrado basado en el kernel de Linux. Es importante destacar que no existe un kernel específico para sistemas empotrados, es decir, no necesitamos crear un kernel especial para sistemas empotrados. A menudo se utilizan las versiones oficiales del kernel de Linux para construir un sistema. Por supuesto, es necesario configurar el mismo para dar soporte especial al hardware de un determinado equipo. Fundamentalmente, un kernel utilizado en un sistema empotrado difiere de un kernel usado en una computadora de escritorio o servidor en la configuración del mismo al momento de compilarlo.
La arquitectura de un sistema GNU/Linux está formado por un conjunto de componentes, y el kernel Linux es sólo una parte de este conjunto.
Inmediatamente sobre el hardware se sitúa el kernel. El kernel es el componente central del sistema operativo. Sus funciones son principalmente administrar el hardware de manera coherente y justa mientras se le otorga un nivel de abstracción familiar, a través de las APIs, a las aplicaciones de nivel de usuario.
Entre otras tareas relevantes de un sistema operativo, el kernel Linux maneja dispositivos, administra los accesos de E/S, controla los procesos y administra el uso compartido de memoria.
Dentro del kernel, la interfaz de bajo nivel es específica para cada configuración de hardware, sobre la cual, el kernel ejecuta y provee control directo de los recursos hardware. Típicamente, los servicios de bajo nivel manejan operaciones específicas de la CPU, operaciones de memoria específicas a la arquitectura, y provee interfaces básicas para dispositivos.
La capa de alto nivel provee abstracciones comunes a todos los sistemas Unix, incluyendo procesos, archivos, sockets y señales. Este nivel de abstracción se mantiene constante aunque difiera el hardware.
Entre estos dos niveles de abstracción, el kernel necesita lo que se denomina componentes de interpretación para comprender e interactuar con datos estructurados provenientes de, o hacia ciertos dispositivos.
Los diferentes tipos de sistemas de archivos y los protocolos de red son ejemplos de fuentes de datos estructurados. El kernel necesita interpretarlos e interactuar a fin de proveer acceso a los datos provenientes desde estas fuentes o hacia las mismas.

7.3 MICROSOFT .NET MICRO FRAMEWORK

La .NET Micro Framework fue creada desde el inicio como una solución .NET para dispositivos integrados pequeños de sensores industriales e instrumentación para sistemas empotrados.
Además de estar totalmente integrada con Visual Studio, el kit de desarrollo de software .NET Micro Framework (SDK) viene equipado con un emulador extensible para simular capacidades de hardware. La estructura permite a los desarrolladores de dispositivos conectar diversas soluciones de hardware para prácticamente cualquier dispositivo periférico mediante conexiones de comunicación estándares de la industria y unidades gestionadas personalizadamente.
La .NET Micro Framework dispone de ofertas integradas de Microsoft en un nuevo mercado de dispositivos que se basan en procesadores de 32 bits de bajo coste y están constreñidos en términos de memoria, potencia de la batería y otros recursos. Ofreciendo paradigmas de programación potentes y modernos a este terreno, .NET Micro Framework pretende acelerar la innovación de dispositivos pequeños y conectados.

- Requiere sólo unos pocos cientos de kilobytes de RAM, y tan poco como 512 K de memoria flash.
- Soporta procesadores con y sin MMU.
- Dispone de una interface de control de energía que permite que la aplicación maximice la vida de la batería.

Hasta ahora la gente de Microsoft a publicado el SDK del Microsoft .NET Micro Framework 2.5 el cual nos permitirá desarrollar aplicaciones para pequeños dispositivos utilizando para tal motivo, Visual Studio, C#. Esta parte de Visual Studio es gratis y puede descargarse de la página Web de Microsoft, su descarga es de apenas 9 Mb y en inglés por ahora para Visual Studio 2005 con SP1. Microsoft .NET Micro Framework es un pequeño subconjunto reducido de clases, en donde podemos desarrollar soluciones en C# para pequeños dispositivos, para trabajar con este pequeño Framework necesitamos disponer de Microsoft Visual Studio 2005 Standar o superior, y un sistema operativo Windows XP, Windows Vista o Windows Server 2003. Dispone de un conjunto de librerías .NET completamente integradas enfocadas al uso en sistemas empotrados.
Varias empresas están apostando por esta tecnología, Digi International Inc dio a conocer sus planes para un lanzamiento previo del kit de desarrollo Digi Connect ME para Microsoft .NET Micro Framework. El Digi Connect ME incluye soporte para redes Ethernet, un puerto en serie y señales de propósitos generales entrada/salida (GPIO). Es la primera solución disponible para .NET Micro Framework para dar apoyo a las redes Ethernet.
EmbeddedFusion, que entrega soluciones centrales de hardware y software integrado para desarrolladores de sistemas integrados, anunció el Meridian CPU, que es un módulo central CPU que incorpora procesador Freescale i.MXS, RAM, Flash y .NET Micro Framework. Para asistir a los desarrolladores en el aprendizaje de cómo se aplica .NET Micro Framework en varios escenarios integrados, EmbeddedFusion también creó la plataforma de desarrollo Tahoe, que permite la experimentación y exploración de .NET Micro Framework justo fuera de la caja.
Freescale también introdujo un kit de desarrollo para .NET Micro Framework que permite a los clientes entregar soluciones diferenciadas en el mercado con rendimiento ARM9 a muy baja potencia.
Además, Rhode Consulting, un especialista en tecnologías Microsoft Windows Embedded, anunció la disponibilidad del kit de evaluación de FlexiDis con .NET Micro Framework instalado. La plataforma FlexiDis utiliza los procesadores centrales Atmel ARM7 y ARM9 con velocidad de hasta 180 MHz. La combinación de esas velocidades, hasta 16 MB de memoria flash y SDRAM, y una pantalla QVGA de 2,2 hace de FlexiDis un componente de elección para varios tipos de aplicaciones industriales en las que se requieren HMI integrado o soluciones de visualización.

7.4 OTROS SISTEMAS OPERATIVOS

7.4.1 eCos

eCos (embedded Configurable operating system) es un sistema operativo de código abierto, gratuito y de operación en tiempo real desarrollado para sistemas empotrados y para aplicaciones que necesiten un procesador con múltiples sesiones. Puede ser personalizado para cumplir los requisitos que la aplicación precise, con cientos de opciones, pudiendo llegar a la mejor combinación entre el rendimiento en tiempo real y el hardware necesario.
Este sistema es programable bajo lenguaje C y tiene capas y APIs compatibles para POSIX y µITRON.
eCos fue diseñado para aparatos con un tamaño de memoria sobre cientos de kilobytes o con requerimientos en tiempo real. Puede ser usado en hardware con muy poca RAM soportando Linux empotrado a partir de un mínimo de 2 MB de RAM, sin incluir las necesidades de la aplicación y del servicio.
eCos funciona correctamente en una amplia variedad de plataformas hardware como pueden ser, ARM, CalmRISC, FR-V, Hitachi H8, IA-32, Motorola 68000, Matsushita AM3x, MIPS, NEC V8xx, Nios II, PowerPC, SPARC, and SuperH.
Incluido con la distribución de eCos disponemos de RedBoot, una aplicación de código abierto que usa la capa de abstracción de hardware de eCos que provee soporte de arranque para sistemas empotrados.
eCos fue inicialmente desarrollado por Cygnus Solutions, aunque más tarde fue comprado por Red Hat. A principio de 2002 cesó el desarrollo de eCos y colocó al personal que estaba trabajando en el proyecto formando su propia compañía, eCosCentric, para continuar con su desarrollo y dar soporte comercial para eCos. En enero de 2004, a partir de una solicitud de los desarrolladores de eCos, Red Hat aceptó transferir los derechos de eCos a la Fundación de Software Libre. Esta transferencia fue finalmente ejecutada en octubre de 2005.
eCosPro está formado por una distribución de eCos y RedBoot creada por eCosCentric y está orientado a desarrolladores que quieran integrar eCos y RedBoot dentro de productos comerciales. Está definido como estable, completamente testado, certificado y con soporte, sin embargo, algunas de sus características no han sido liberadas como software libre.

7.4.2 uC/OS

MicroC/OS-II (comúnmente llamado µC/OS-II o uC/OS-II), es un sistema operativo multitarea, en tiempo real, basado en prioridad preventiva, de bajo coste donde el kernel está escrito principalmente en el lenguaje de programación C. Es principalmente entendido para uso en sistemas empotrados.
La designación II es debido a que es la segunda generación de un kernel que originalmente fue publicado en 1992 en la segunda parte de un articulo en la revista Embedded Systems Programming bajo el titulo µC/OS The Real-Time Kernel y escrito por Jean J. Labrosse (ISBN 0-87930-444-8). El autor intentó describir como funciona un sistema operativo portable por dentro y las razones por las que fue desarrollado. Pero rápidamente todo esto tomó un rumbo comercial.
uC/OS-II es soportado por Micrium Inc y se obtiene bajo licencia del producto, aunque el uso de este sistema operativo es gratis para uso educacional o no comercial. Adicionalmente Micrium distribuye otros productos de software como uC/OS-View, uC/CAN, uC/TCP-IP, uC/FS, uC/GUI, uC/MOD-BUS, uC/LCD, uC/USB (Mass Storage Device and Bulk) y un largo grupo de aplicaciones para uC/TCP-IP como software cliente para DHCP, POP3, SNTP, FTP, TFTP, DNS, SMTP, y TTCP. El software en su modalidad de servido incluye HTTP, FTP y TFTP. PPP es también disponible.
Está disponible para la mayor cantidad de procesadores y placas que existen en el mercado y es adecuado para el uso en sistemas empotrados donde la seguridad es crítica como en aviación, sistemas médicos o instalaciones nucleares.

7.4.3 Contiki

Contiki es un pequeño sistema operativo de código abierto, altamente portable y multitarea, desarrollado para uso en pequeños sistemas, desde ordenadores de 8-bit a sistemas empotrados sobre

microcontroladores, incluyendo nodos de redes de sensores. El nombre Contiki viene de la famosa balsa Kon-Tiki de Thor Heyerdahl.
A pesar de incluir multitarea y una pila TCP/IP, Contiki sólo requiere varios kilobytes de código y unos cientos de bytes de RAM. Un sistema totalmente completo con una GUI requiere aproximadamente 30 kilobytes de RAM.
El núcleo básico y la mayor parte de las funciones principales fueron desarrolladas por Adam Dunkels en el grupo de sistemas de redes empotradas en el instituto sueco de ciencias computacionales.
Contiki fue diseñado para sistemas empotrados con poca cantidad de memoria. Una configuración típica de Contiki es 2 kilobytes de RAM y 40 kilobytes de ROM. Contiki consiste en un núcleo orientado a eventos, el cual hace uso de protohilos, sobre el cual los programas son cargados y descargados dinámicamente. También soporta multihilado apropiativo opcional por proceso, comunicación entre procesos mediante paso de mensajes a través de eventos, al igual que un subsistema GUI opcional, el cual puede usar un soporte directo de gráficos para terminales locales, terminales virtuales en red mediante VNC o sobre Telnet
Contiki funciona en una variedad de plataformas, desde microcontroladores empotrados, como el MSP430 y el AVR, a viejas computadoras domésticas. El tamaño del código está en el orden de los kilobytes y el uso de la memoria puede configurarse para que sea de sólo unas decenas de bytes.

7.4.4 MANTIS

El sistema operativo MANTIS (MultimodAl system for NeTworks of In-situ wireless Sensors) suministra un nuevo sistema operativo empotrado de plataforma múltiple para redes de sensores inalámbricos. Ante el incremento de complejidad de las tareas realizadas por las redes de sensores como compresión, agregación y procesado de señales, los multiprocesos en MANTIS sensor OS (MOS) permiten interpaginar complejas tareas con tareas susceptibles al tiempo para así mitigar los problemas en los saltos de buffers.
Para conseguir una eficiencia en el uso de la memoria, MOS es implementado para que utilice una pequeña cantidad de RAM. Usando menos de 500 bytes de memoria, incluyendo el kernel, los controles de tiempo y la pila de comunicación. Para conseguir la eficiencia energética, el controlador de eficiencia energética de MOS hace que el microcontrolador duerma después de ejecutar todas las tareas activas, reduciendo el consumo de energía a un rango de μA.
Una de las características principales de MOS es la flexibilidad en el soporte de múltiples plataformas como PCs, PDAs y diferentes plataformas de microsensores. Otra de las características destacada del diseño de MOS es el soporte de control remoto, permitiendo una reprogramación dinámica y un acceso remoto [BHA2005].

7.4.5 BTnut

Sistema operativo de código abierto creado para correr dentro de sistemas empotrados BTnodes. Fue diseñado principalmente para el procesador Atmel ATmega128 (el cual forma parte de los motes BTnodes) y por lo tanto es el más recomendado para esta clase de motas.
La actual compilación de sus programas (la conversión de código C a código maquina) es realizada usado gcc-avr, el cual es un compilador de C libre para la plataforma de procesadores Atmel. Podemos diferenciar tres partes: las rudimentarias librerías C que son implementadas por la parte avr-libc de gcc-avr; las rutinas de alto nivel construidas para avr-libc por Nut/OS; y los drivers específicos para el funcionamiento del mote BT-node proporcionados por BTnut [BTN2008].

7.4.6 SOS

SOS es un sistema operativo desarrollado en la Universidad de California (UCLA), específicamente en el NESL (Networked & Embedded Systems Laboratory).
SOS es un sistema operativo para redes de sensores que procura remediar algunos de las limitaciones propias de la naturaleza estática de muchos de los sistemas precursores a éste (por ejemplo TinyOS).
SOS implementa un sistema de mensajería que permite múltiples hebras entre la base del sistema operativo y las aplicaciones, las cuales pasan a ser módulos que pueden ser cargadas o descargadas en tiempo de ejecución sin interrumpir la base del sistema operativo.

El principal objetivo de SOS es la reconfigurabilidad. Ésta se define como la habilidad para modificar el software de nodos individuales de una red de sensores, una vez que estos han sido desplegados físicamente e inicializado su funcionamiento. En el caso de encontrar un problema, en caso de no contar con esta solución, habría sido necesario recolectar todos los nodos para poder modificar su software.
La capacidad de dinámicamente agregar o remover módulos, permite la construcción de software mucho más tolerante a fallos. Esto presenta dos grandes ventajas: una es el hecho de poder realizar updates fácilmente, y la otra es la capacidad de anular el funcionamiento de algún módulo defectuoso, de algún nodo que pertenece a la red.
Además de las técnicas tradicionales usadas en el diseño de sistemas empotrados, las características del kernel de SOS son:

- Módulos cargados dinámicamente.
- Programación flexible de prioridades.
- Simple subsistema de memoria dinámica.

7.4.7 Nano-RK

Nano-RK es un sistema operativo completamente preventivo basado en reserva bajo tiempo real (RTOS) desarrollado en la universidad de Carnegie Mellon con soporte para redes multisalto adecuado para el uso en redes de sensores inalámbricas. Nano-RK funciona adecuadamente con plataformas como redes de sensores FireFly y sobre los motes MicaZ.
Incluye un kernel con recursos empotrados de bajo peso con bastantes funcionalidades y soporte de tiempo usando menos de 2 KB de memoria RAM y 18 KB de ROM. Nano-RK soporta multitareas preventivas con prioridad para asegurar que los plazos de las tareas son conocidos, además de soporte de CPU, red y sensores y actuadores.

Las tareas pueden especificar las demandas de recursos y el sistema operativo provee el acceso controlado y garantizado para los ciclos de CPU y los paquetes de red. Todos estos recursos forman la reserva de energía virtual que permite al sistema operativo controlar el nivel de energía del sistema y de las tareas [ESW2005].

8 APLICACIÓN BAJO TECNOLOGÍA MICAZ

La situación actual en los vertederos españoles es un motivo de preocupación para la sociedad. Las investigaciones realizadas muestran la generación de emisiones contaminantes, no solo durante el periodo de vertido sino bastante tiempo después. Intentar controlar estas emisiones o incluso realizar un aprovechamiento energético tratando los caudales de ciertos gases requiere disponer de información.
La tecnología nos permite poder obtener datos de diferentes variables medioambientales gracias al gran avance que ha sufrido el campo de los sensores, como puede ser temperatura, presión, luz, etc. Poder realizar un tratamiento en el momento y condiciones adecuadas dentro del vertedero, puede depender de las magnitudes obtenidas y de completar un estudio apropiado de ellas.
La posibilidad de captar la información del medio no implica que se pueda procesar y estudiar, ya que dichos datos deben medirse en el tiempo previsto y en espacios separados dentro del recinto seleccionado. Instalar un complejo sistema de sensores que transmitan la información hasta una única unidad central capacitada para estudiar los datos, en emplazamientos geográficos y en condiciones ambientales adversas, similares a las que nos podemos encontrar en un vertedero de residuos urbanos, puede ser una tarea cara y complicada.
El objetivo, además de salvar todos los problemas en la instalación, es realizar una recogida de datos de varias magnitudes físicas, almacenarlas en un sistema central y que gracias a un procesado y un almacenamiento de dicha información, una persona cualificada pueda tomar decisiones acerca de lo más conveniente para realizar una correcta gestión del vertedero.

8.1 TECNOLOGÍA MICAZ

Realizado un estudio de las posibles tecnologías disponibles para desarrollar esta WNS, se eligió la tecnología MicaZ. Esta decisión fue tomada ya que, en ese momento, esta familia era la que más accesorios tenía para realizar el trabajo, además de ser una de las que mejores características disponía.
Los nodos MicaZ disponen de un chip de la casa chipcon modelo CC2420 que acata la norma IEEE 802.15.4. Esta norma es un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal con tasas bajas de transmisión de datos, como es nuestro caso. También dispone de un microcontrolador, un Atmega128L, de bajo consumo donde pueden ejecutarse programas desde la memoria flash interna disponible [XBOW2008-2].
Como es una evolución de la tecnología mica, y al disponer de un conector de entrada salida de 51 pines, muchas de las aplicaciones y de las placas sensoras disponibles para esta familia pueden ser utilizadas para los MicaZ.
Estos dispositivos han sido diseñados para trabajar bajo alimentación de baterías, en concreto con dos baterías del tipo AA, sin embargo pueden usarse otras fuentes siempre que la alimentación esté comprendida entre 2.7 y 3.6 VDC. Las baterías suministradas son del tipo alcalinas con una carga de 2000 mAh.

8.1.1 PLACAS SENSORAS EN TECNOLOGÍA MICAZ

Como se dijo antes, la tecnología MicaZ dispone de una de las más amplias gamas de sensores, y aquí se han estudiado varias de ellas.

Tabla 7 Placas sensoras para MicaZ

Placa sensora	Sensores y características
MTS310CB	Luz, temperatura, micrófono, zumbador, acelerómetro de 2 ejes y magnetómetro de dos ejes
MTS400CB	Luz ambiente, humedad relativa, temperatura, acelerómetro de dos ejes y presión barométrica.
MTS420CC	Lo mismo que el MTS400CB pero con un módulo de GPS.
MDA100CB	Luz, temperatura y un área para prototipos.
MDA300CA	Luz, humedad relativa e interface de propósito general para sensores externos.

Con todas estas placas y con los sensores integrados en ellas, se obtienen medidas útiles que pueden ser usadas para nuestro propósito. Hay dos tipos de placas, las placas con sensores integrados y las placas a las que pueden conectarse sensores externos a través de los ADC. Siendo difícil encontrar sensores que se adapten a las características de este tipo de placas. Los sensores externos nunca podrán tener una salida con tensiones mayores de 3 v y menores de 0 v ya que voltajes diferentes a estos podrían dañar la placa [XBOW2008-3].

Figura 24 Placas sensoras MTS/MDA

No solo la tensión obtenida del sensor es un problema para la placa sensora, la tensión que el mote debe suministrar al sensor externo para que este funcione correctamente alimentado, está también limitada, ya que esta tensión deberá obtenerse del sistema de alimentación del mote, en este caso dos baterías del tipo AA.

8.1.2 REDES DE BAJO CONSUMO

En las redes de sensores, tener en cuenta el consumo de energía es muy importante, sobre todo si queremos que los periodos de vida de los dispositivos sean máximos. Puesto que los nodos de nuestra red no estarán conectados a una red eléctrica y su funcionamiento se basa en la utilización de baterías principalmente, se ha intentado reducir y optimizar el consumo de energía dentro del nodo. A pesar de esto, se debe de tener en cuenta cuanto tiempo estará nuestro nodo alimentado correctamente y cuando dejará de funcionar.

Realizar un estudio del consumo de energía en una red de sensores mediante medida directa es inviable, debido al gran numero de nodos y al diferente nivel de consumo en cada uno de ellos. Las investigaciones realizadas sobre este consumo son más bien pocas y la mayoría están basadas en simulaciones de software [HYU2008].

Unos consumos de energía medios en este tipo de modulo inalámbrico son:

Tabla 8 Consumo del procesador en MicaZ

Consumo del procesador	
Modo activo	8 mA
Modo rendimiento completo	12 mA (7.37 Mhz)
Modo durmiente	< 15 µA

Tabla 9 Consumo del sistema de transmisión en MicaZ

Consumo del sistema de transmisión	
Modo recepción	19.7 mA
Modo transmisión -10 dBm	11 mA

Consumo del sistema de transmisión	
Modo transmisión -5 dBm	14 mA
Modo transmisión 0 dBm	17.4 mA
Modo inactivo	20 µA
Modo durmiente	1 µA

Tabla 10 Consumo del acceso a memoria en MicaZ

Consumo acceso a memoria	
Escribir en memoria flash	15 mA
Leer en memoria flash	4 mA
Modo durmiente	2 µA

Tabla 11 Consumo del sistema sensor en MicaZ

Consumo del sistema sensor	
Temperatura	0.4 mA
Humedad	1.5 mA
Luz	0.030 + 0.002 * v mA
Gas	174.618 + v * 0.011

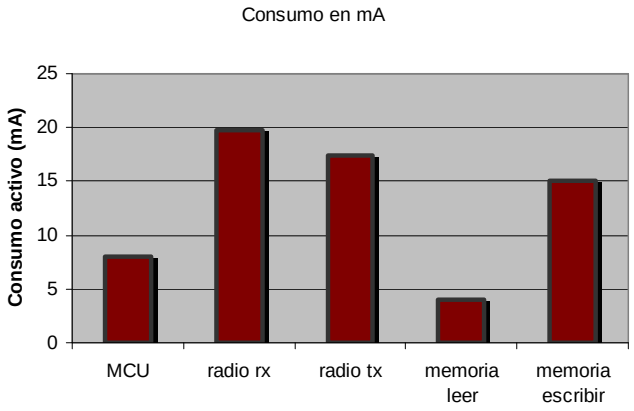

Figura 25 Comparativa de los consumos en tecnología MicaZ

La tecnología MicaZ incorpora un módulo de alimentación que utiliza dos baterías tipo AA. Las baterías suelen expresarse en mAh, lo que quiere decir que una batería de 1000 mAh debería soportar un consumo de 10 mA durante 100 horas, aunque en la teoría es algo menos.
Las tecnologías en las que se basan las baterías pueden ser varias, alcalinas, litio, níkel-metal-hidruro, pero para entender el ciclo de vida del nodo podemos ver la grafica de vida de una pila alcalina, ya que por defecto los nodos micaz vienen con pilas de este tipo.

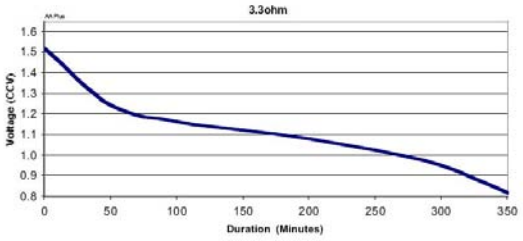

Figura 26 Consumo de una pila Panasonic Industrial Alkaline

El nodo MicaZ esta diseñado para trabajar a una tensión de 2.7 v, lo que nos lleva a trabajar con dos pilas del tipo AA. Pero como vemos en la gráfica, aunque el tiempo depende de la resistencia, el voltaje suministrado rápidamente cae por debajo de lo necesario.

Para solucionar el problema de la alimentación nos decantamos por la energía solar y su sistema de funcionamiento. Actualmente una placa solar puede producir más de 15 mW por centímetro cuadrado y si esta energía es almacenada correctamente bastaría con consumir por la noche lo almacenado por el día.
El dispositivo seleccionado para solucionar el problema es de la casa Atla Labs y se llama Heliomote. Este dispositivo es un sistema de alimentación solar para dispositivos pequeños y de bajo consumo. Su principal propósito es captar la energía del sol y almacenarla en unas pilas NiMH que tiene incorporadas, además de suministrar energía al dispositivo encargado de alimentar.
Este dispositivo está adaptado para poder sobrevivir a ambientes difíciles, mediante un recubrimiento IP-67, a prueba de golpes, de polvo y resistente al agua para una profundidad de 1.8 metros. Esto permite al Heliomote, o a los dispositivos de su interior, medir características en el exterior [HEL2008-1].

Figura 27 Heliomote

La tecnología MicaZ puede adaptarse perfectamente a este dispositivo, conectándose a través de su conector de 51 pins, ya que dispone de un material que evita la estática, para proteger al nodo, y se adapta al tamaño y posición de este. Proporciona una tensión de salida de 3 v, que es suficiente para alimentar un MicaZ.
La siguiente figura muestra como se comporta el panel solar del Heliomote ante varias condiciones diferentes de iluminación solar que van desde una noche regular a un día brillante. El pico de potencia de salida es 94 mW. En un día normal el Heliomote puede llegar a almacenar 1800 J de energía. Un MicaZ operando a pleno funcionamiento consume 0.3 W aproximadamente. Ésto quiere decir que si el MicaZ está trabajando constantemente a pleno funcionamiento consume más energía de la que el Heliomote puede producir y deja de funcionar rápidamente [HEL2008-2].

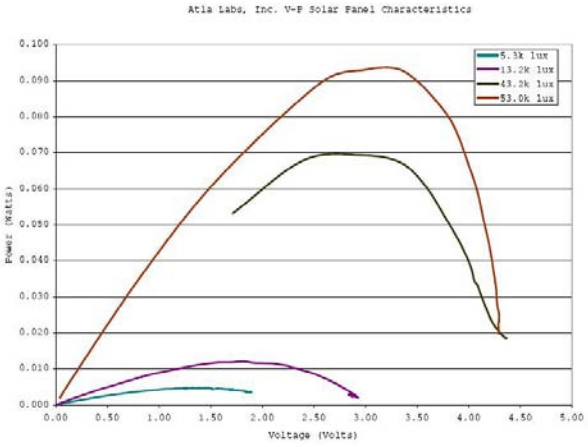

Figura 28 Comportamiento del panel solar del Heliomote

Para que el funcionamiento del MicaZ se alargue lo máximo posible en el tiempo debe optimizarse el software. Esta optimización significa que el dispositivo pase la mayor parte de su tiempo dormido, para despertar en los momentos en que deba leer de los sensores o transmitir información.

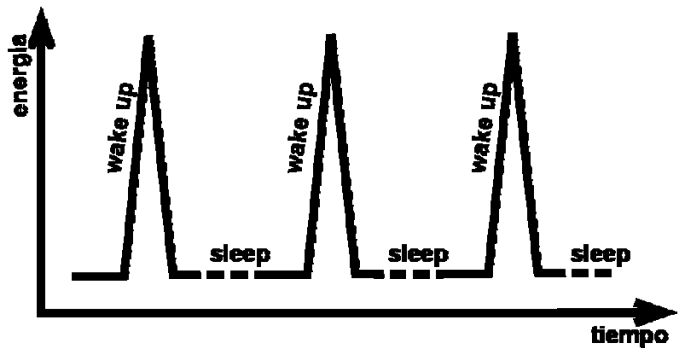

Figura 29 Funcionamiento de bajo consumo

El funcionamiento del programa utilizado en los MicaZ sigue el funcionamiento analizado en la siguiente figura. Este funcionamiento es el utilizado en la topología tipo malla asíncrona y cada 125 ms dan un signo de vida esperando una sincronización en el caso de que se tenga que realizar una comunicación.

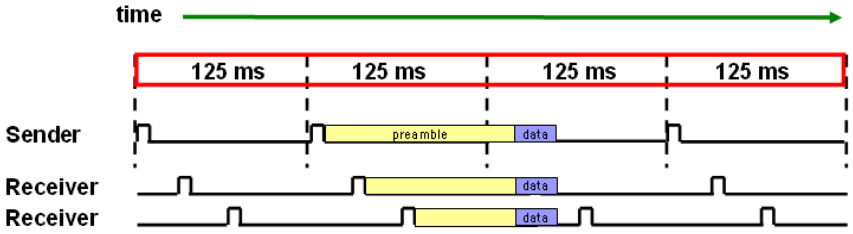

Figura 30 Sincronización del envío de datos

Las diferencias entre funcionar constantemente, en el nodo, la radio y el procesador en modo constantemente activo, a funcionar sólo en intervalos, donde el nodo esta durmiendo durante la mayoría del tiempo, son bastante significativas.

Tabla 12 Diferencias en los usos tiempo-consumo

Parámetro	Uso constante	Sleep-wake up
Intervalo de routeo	36 sec.	360 sec.
Ratio de datos del mensaje	10 sec.	180 sec.
Media de corriente usada	20-30 mA	<400 μA

8.2 PUERTA DE ENLACE

Dentro de nuestra red es necesario disponer de elementos que sean capaces de recoger las lecturas realizadas por los motes y almacenarlas posteriormente.
Una vez realizado un estudio de los posibles aparatos para realizar estas operaciones, el elegido fue el Stargate SPB400 de Crossbow [XBOW2008-4].

Figura 31 Stargate SPB400

Este dispositivo tiene instalado un sistema operativo Linux donde puede ejecutarse código realizado para captar los datos enviados por los motes. Una vez captados los datos, éstos son analizados y almacenados en una base de datos para su uso posterior.
Como el Stargate dispone de conexión con el exterior, puede conectarse a una red local o a Internet y así hacer que los datos estén disponibles en cualquier lugar.

8.3 ESTACIÓN BASE

Los datos recogidos por el gateway son filtrados, analizados y enviados a un servidor externo donde se dispone de una base de datos para su almacenamiento. Esta base de datos es accesible desde un simple sistema Web que muestra los datos almacenados.

8.4 EJEMPLO DE MUESTRAS OBTENIDAS

Las pruebas realizadas con la placa MTS310 nos dan los siguientes valores en cuanto a consumo de tensión.

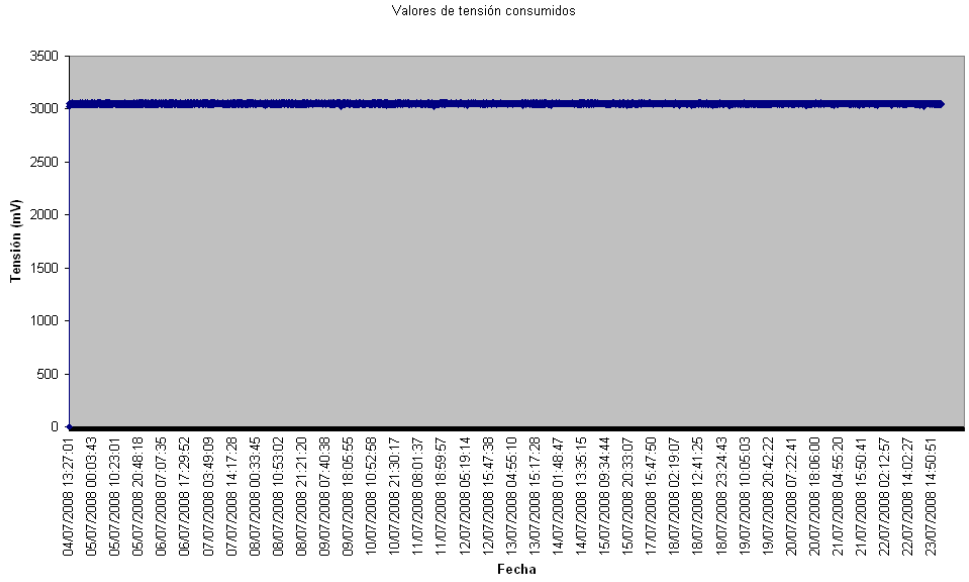

Figura 32 Valores de tensión consumidos en la placa MTS310

Ampliando la escala sobre la línea de valores podemos ver lo siguiente.

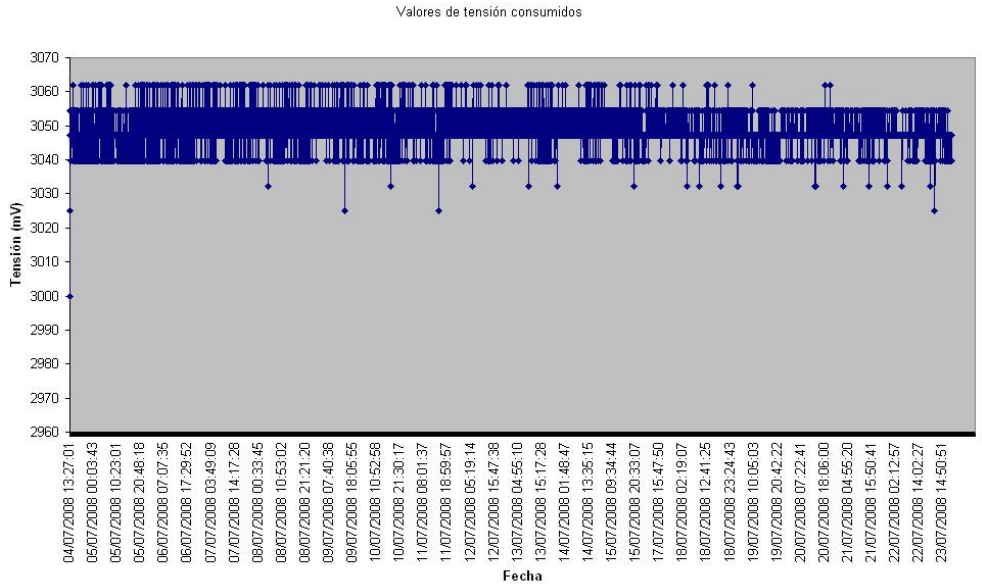

Figura 33 Detalle de los valores de tensión consumidos en la placa MTS310

Podemos observar que la línea sí tiene una tendencia ha reducir la tensión, pero no nos da una idea clara de que la reducción sea la causa de que pare el funcionamiento del mote.

Otra de las opciones a observar era la temperatura.

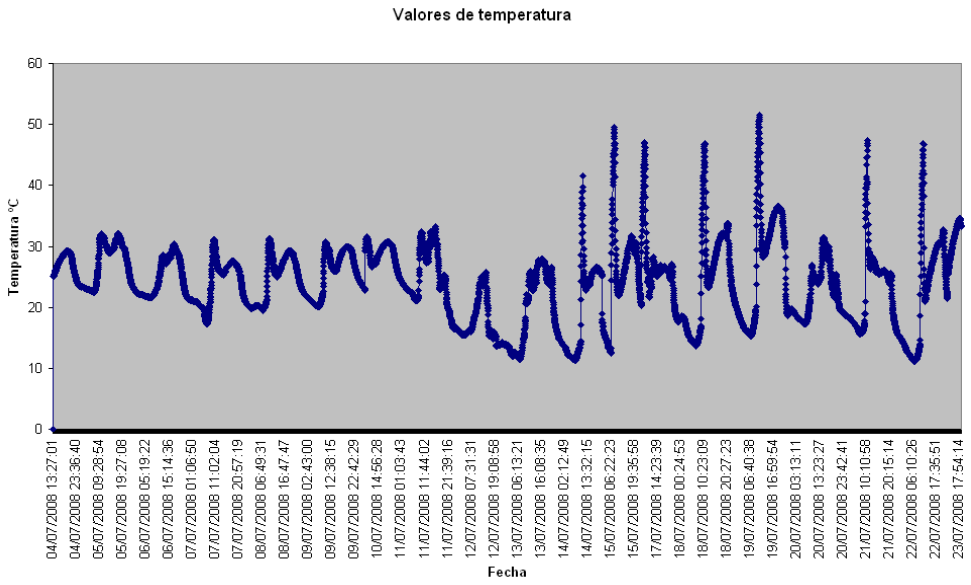

Figura 34 Valores de temperatura leídos en la placa MTS310

Como podemos observar a partir de la mitad de la grafica el mote fue trasladado de zona y los picos que aparecen son consecuencia de la aparición del sol. Al estar introducido en el Heliomote ante la presencia de sol, la temperatura se disparaba.

Los datos obtenidos al medir la cantidad de luz son los siguientes.

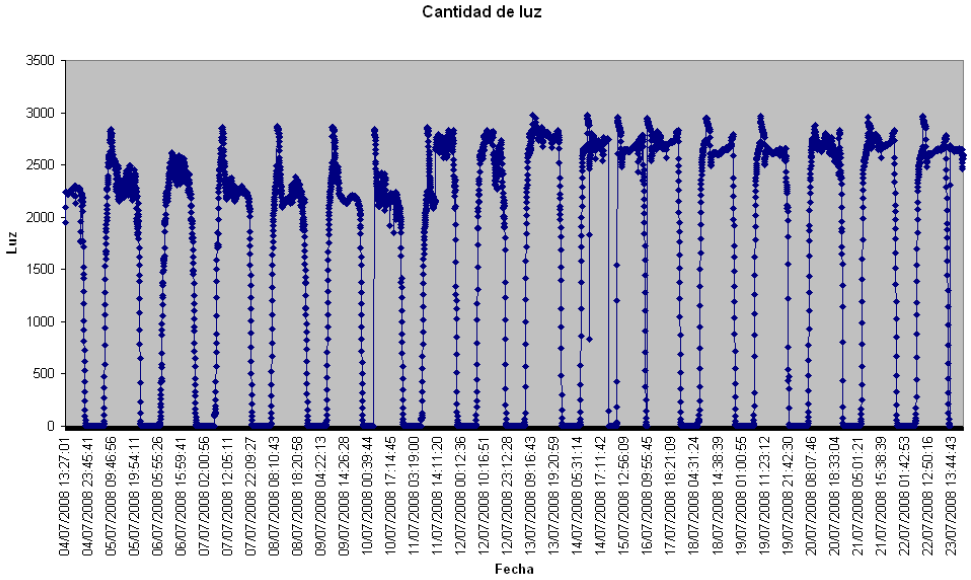

Figura 35 Valores de cantidad de luz leídos en la placa MTS310

Los tonos captados por el micrófono fueron los siguientes.

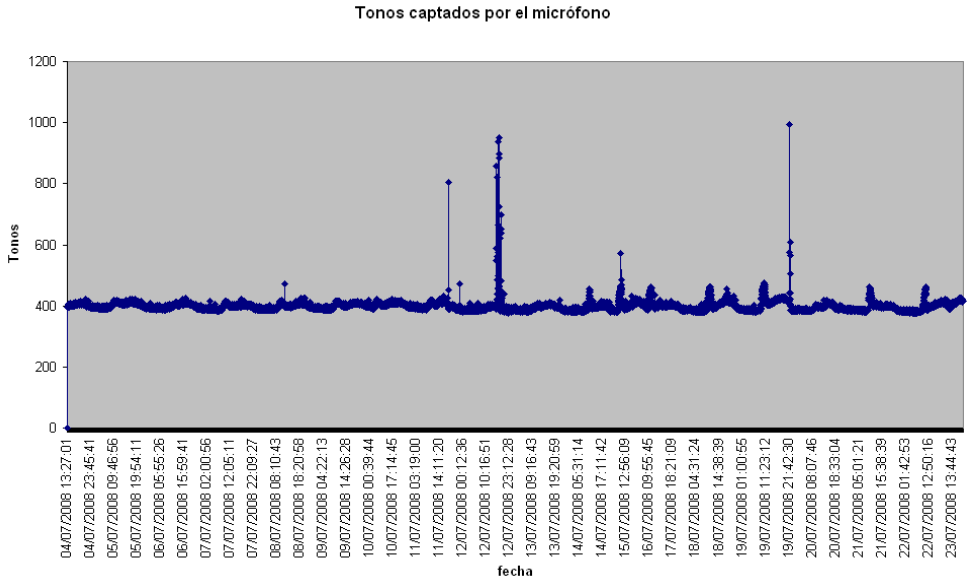

Figura 36 Tonos captados por el micrófono en la placa MTS310

Las variaciones de aceleración fueron las siguientes. En el eje X.

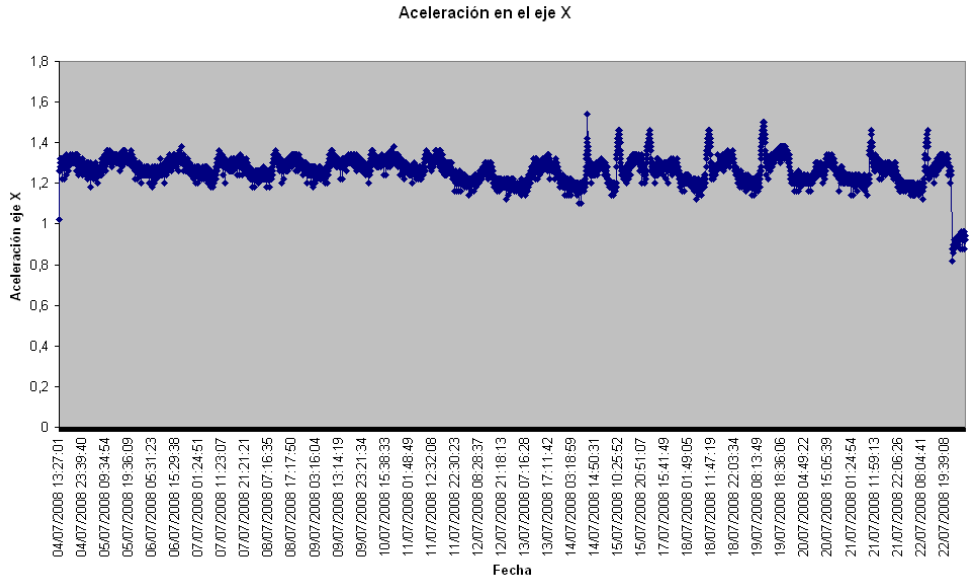

Figura 37 Valores de aceleración en el eje X leídos en la placa MTS310

Y en el eje Y.

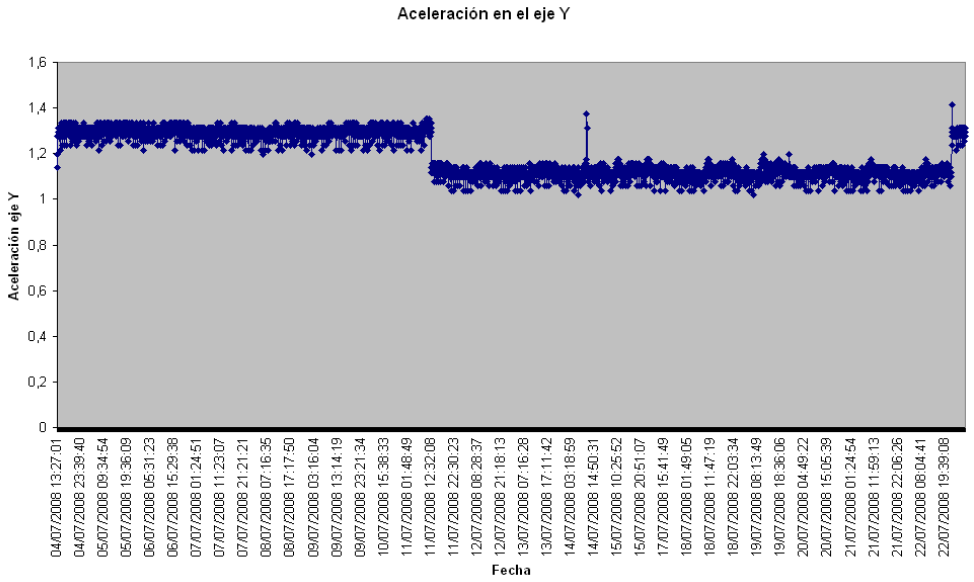

Figura 38 Valores de aceleración en el eje Y leídos en la placa MTS310

Otra de las mediciones realizadas fue el campo magnético. En el eje X.

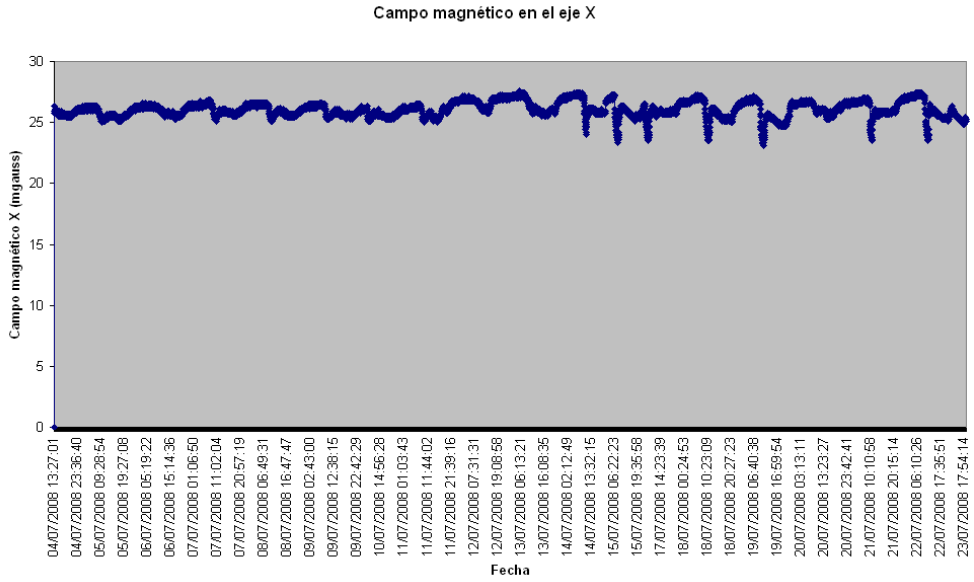

Figura 39 Valores de campo magnético en el eje X leídos en la placa MTS310

Y en el eje Y.

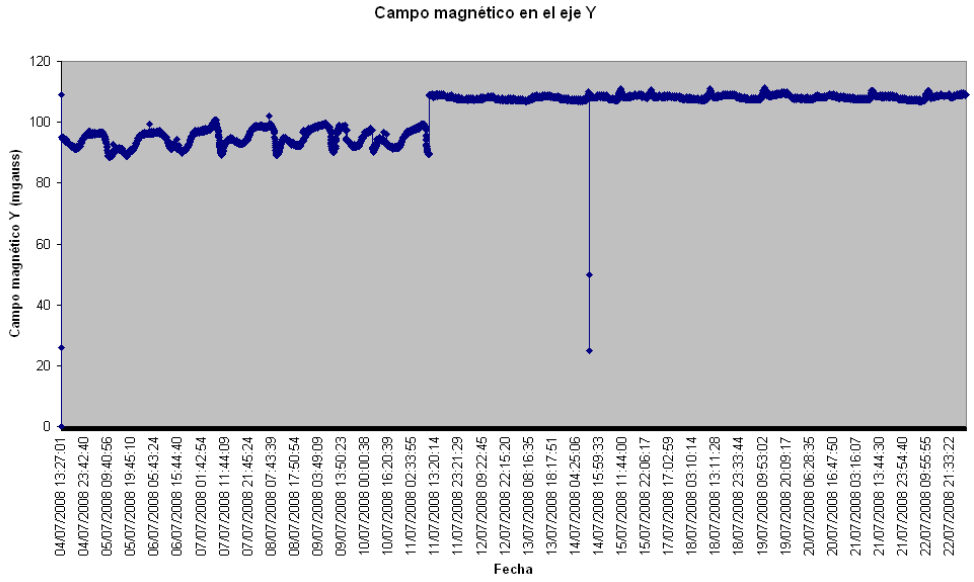

Figura 40 Valores de campo magnético en el eje Y leídos en la placa MTS310

8.5 DESCRIPCIÓN DE LOS MENSAJES

Cuando se recibe en una estación base un paquete se necesita saber que contiene la información contenida en él.
Varios puntos importantes para entender estos mensajes son:

- Un paquete TinyOS tiene una longitud máxima de 255 bytes.
- El paquete en crudo está contenido entre un carácter especial 0x7E, que indica donde empieza y donde acaba el paquete. Éste es el primer y ultimo carácter del mensaje.
- Los mensajes tienen un byte de escape, 0x7D. Este mensaje es necesario siempre que tenga que aparecer el byte reservado 0x7E dentro del mensaje. En este caso, el byte dado pasará a estar formado por dos byte. El primero es el byte de escape y el segundo será una XOR del byte original con el valor 0x20. Por ejemplo, si tuviera que aparecer el byte 0x7D en el mensaje, lo que realmente aparecerá será 0x7D 0x5E.
- En una máquina XP, múltiples bytes son encadenados en la misma cadena. Por ejemplo, los dos bytes del campo de la dirección UART (0x007E) aparecerán como 7E 00 en la cadena de bytes.

8.5.1 ESTRUCTURA GENERAL DEL MENSAJE

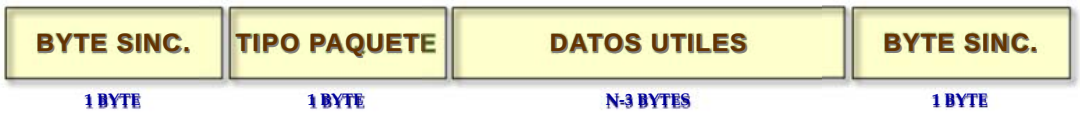

Figura 41 Estructura general del mensaje

Tabla 13 Campos de la estructura general del mensaje

Tamaño (byte)	Campo	Descripción
1	sync_byte	Siempre 0x7E
1	Type	Tipo del mensaje activo
n-3	payload data	Datos útiles del mensaje
1	sync_byte	Siempre 0x7E

Los tipos de mensajes que se pueden tener son:

- P_PACKET_NO_ACK (0x42) – Paquete que no requiere ACK.
- P_PACKET_ACK (0x41) – Paquete que requiere ACK. Incluye un prefijo. El receptor debe enviar un paquete de respuesta del tipo P_ACK conteniendo el prefijo enviado.
- P_ACK (0x40) – Es la respuesta al tipo de paquete P_PACKET_ACK. Debe incluir el prefijo en su contenido.
- P_UNKNOWN (0xFF) – Tipo de paquete desconocido.

Los datos útiles se componen de la siguiente manera:

Figura 42 Estructura de los datos útiles del mensaje

8.5.2 CABECERA TinyOS

Los datos útiles de los paquetes empiezan con una cabecera definida como cabecera TinyOS que tiene la siguiente estructura.

Figura 43 Estructura de la cabecera TinyOS

Tabla 14 Campos de la estructura de la cabecera TinyOS del mensaje

Tamaño (byte)	Campo	Descripción
2	addr	Dirección de destino del salto
1	type	Tipo del mensaje activo
1	group	ID del grupo del mensaje activo
1	lenght	Longitud del mensaje entero

Las direcciones de destino pueden ser de tres tipos:

- TOS_BCAST_ADDR: es la dirección de broadcast (0xFFFF) donde el mensaje es para todos los nodos.
- TOS_UART_ADDR: es la dirección UART (0x007E) y el mensaje va desde un nodo al puerto serie del gateway.
- TOS_LOCAL_ADDRESS: es la dirección del nodo, es un ID único del nodo que recibe el mensaje.

8.5.3 CABECERA DE LA TOPOLOGÍA EN MALLA

Los paquetes enviados dentro de la malla tienen una cabecera como la siguiente. Si la topología es en estrella y el mensaje sólo tiene un salto esta parte no se usa.

Figura 44 Estructura de cabecera de topología en malla

Tabla 15 Campos de la estructura de la cabecera de topología en malla

Tamaño (byte)	Nombre del campo	Descripción
2	sourceaddr	Dirección de la fuente del mensaje
2	originaladdr	ID del nodo enviador del mensaje original
2	seqno	Numero de secuencia estimado por el enlace
1	socket	ID de la aplicación

8.5.4 CABECERA BÁSICA DE MENSAJE

Cada mensaje tiene una cabecera básica con información que identifica el nodo emisor, así como sobre la placa sensora.

Figura 45 Estructura de la cabecera básica del mensaje

Tabla 16 Campos de la cabecera básica del mensaje

Tamaño (byte)	Nombre del campo	Descripción
1	idsensorboard	Identificador de la placa sensora
1	idsensorpacket	ID del nodo emisor del mensaje original
2	parent	ID de la aplicación

Cada placa sensora disponible para la tecnología MicaZ dispone de un número de identificación único. Los de las placas mas utilizadas de esta tecnología son las siguientes.

Tabla 17 Numero de identificación de las placas sensoras

Modelo	ID (hex)	ID (dec)	Descripción
MTS310	0x84	132	Acelerómetro (2 ejes), zumbador, fotoresistor, micrófono acústico, magnetómetro (2 ejes) y termistor.
MTS400	0x85	133	Acelerómetro (2 ejes), presión barométrica, luz, humedad relativa y temperatura.
MTS420	0x86	134	Acelerómetro (2 ejes), presión barométrica, modulo GPS, luz, humedad relativa y temperatura.
MDA100	0x91	140	Fotoresistor, termistor y entradas externas para ADC 10 bits.
MDA300	0x81	129	Entradas externas para ADC de 12 bits, humedad relativa y temperatura.

8.5.5 MENSAJE MTS310

La placa sensora MTS310 es una placa con una variedad de sensores de varios tipos. Estas modalidades incluyen temperatura, luz, micrófono, zumbador, magnetómetro en dos ejes y aceleración en dos ejes.
La estructura del paquete es la siguiente:

Figura 46 Estructura del mensaje MTS310

Tabla 18 Campos de la estructura del mensaje en MTS310

Tamaño (byte)	Nombre del campo	Descripción
2	voltage	Lectura de la batería
2	thermistor	Lectura de la temperatura
2	light	Lectura de la cantidad de luz
2	microphone	Medida del rango acústico
2	accel_x	Lectura de la aceleración en el eje X
2	accel_y	Lectura de la aceleración en el eje Y
2	mag_x	Lectura del magnetismo en el eje X
2	mag_y	Lectura del magnetismo en el eje Y

8.5.6 MENSAJE MTS400

La placa sensora MTS400 tiene una variedad de sensores meteorológicos. Estas modalidades incluyen temperatura, humedad, presión barométrica, luz y aceleración en dos ejes.
La estructura del paquete es la siguiente:

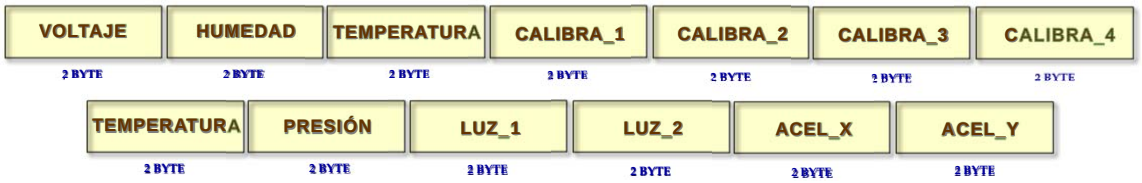

Figura 47 Estructura del mensaje MTS400

Tabla 19 Campos de la estructura del mensaje en MTS400

Tamaño (byte)	Nombre del campo	Descripción
2	voltage	Lectura de la batería
2	humidity	Lectura de la humedad
2	temp	Lectura de la temperatura
2	cal_word1	Parte 1 utilizada para calibrar la presión
2	cal_word2	Parte 2 utilizada para calibrar la presión
2	cal_word3	Parte 3 utilizada para calibrar la presión
2	cal_word4	Parte 4 utilizada para calibrar la presión
2	intersematemp	Lectura de temperatura del sensor intersema
2	intersemapressure	Lectura de presión barométrica del sensor intersema
2	taosch0	Parte 1 de la lectura de la cantidad de luz
2	taosch1	Parte 2 de la lectura de la cantidad de luz
2	accel_x	Lectura de la aceleración en el eje X
2	accel_y	Lectura de la aceleración en el eje Y

8.5.7 MENSAJE MTS420

La placa sensora MTS420 tiene una variedad de sensores meteorológicos además de un sistema GPS. Estas modalidades incluyen temperatura, humedad, presión barométrica, luz y aceleración en dos ejes.
La estructura del paquete es la siguiente:

Figura 48 Estructura del mensaje MTS420

Tabla 20 Campos de la estructura del mensaje en MTS420

Tamaño (byte)	Nombre del campo	Descripción
2	voltage	Lectura de la batería
2	humidity	Lectura de la humedad
2	temp	Lectura de la temperatura
2	cal_word1	Parte 1 utilizada para calibrar la presión
2	cal_word2	Parte 2 utilizada para calibrar la presión
2	cal_word3	Parte 3 utilizada para calibrar la presión
2	cal_word4	Parte 4 utilizada para calibrar la presión
2	intersematemp	Lectura de temperatura del sensor intersema
2	intersemapressure	Lectura de presión barométrica del sensor intersema

Tamaño (byte)	Nombre del campo	Descripción
2	taosch0	Parte 1 de la lectura de la cantidad de luz
2	taosch1	Parte 2 de la lectura de la cantidad de luz
2	accel_x	Lectura de la aceleración en el eje X
2	accel_y	Lectura de la aceleración en el eje Y
1	hours	Tiempo GPS: horas
1	minutes	Tiempo GPS: minutos
1	lat_deg	Latitud GPS: grados
1	long_deg	Longitud GPS: grados
4	dec_sec	Tiempo GPS: segundos
4	lat_dec_min	Latitud GPS: minutos
4	long_dec_min	Longitud GPS: minutos
1	NSEWind	Velocidad GPS
1	fixed	Estado de reparación de GPS

8.5.8 MENSAJE MDA100

La placa MDA100 es una placa sensora y de adquisición de datos, que dispone de un termistor de precisión, una fotocélula para captar la cantidad de luz y un área general de prototipazo.

Figura 49 Estructura del mensaje MDA100

Tabla 21 Campos de la estructura del mensaje en MDA100

Tamaño (byte)	Nombre del campo	Descripción
2	vref	Lectura de la batería
2	thermistor	Lectura de la temperatura
2	photo	Lectura de la cantidad de luz
2	adc2	Lectura del convertidor analógico digital
2	adc3	Lectura del convertidor analógico digital
2	adc4	Lectura del convertidor analógico digital
2	adc5	Lectura del convertidor analógico digital
2	adc6	Lectura del convertidor analógico digital

8.5.9 MENSAJE MDA300

La MDA300 es una placa de monitorización medioambiental. Tiene una variedad de modalidades sensoras, las cuales incluyen, entradas analógicas de 12 bits, ocho entradas/salidas digitales y una excitación seleccionable de 2.5v, 3v y 5v. Además tiene incorporado un sensor de humedad y temperatura.

Figura 50 Estructura del mensaje MDA300

Tabla 22 Campos de la estructura del mensaje en MDA300

Tamaño (byte)	Nombre del campo	Descripción
2	vref	Lectura de la batería
2	humid	Lectura de la temperatura
2	humtemp	Lectura de la cantidad de luz
2	adc0	Lectura del convertidor analógico digital
2	adc1	Lectura del convertidor analógico digital
2	adc2	Lectura del convertidor analógico digital
2	dig0	Lectura de la entrada digital
2	dig1	Lectura de la entrada digital
2	dig2	Lectura de la entrada digital

8.5.10 CRC

Al final del paquete se envía un CRC que calcula la correcta emisión y recepción del mensaje. El CRC son dos bytes que se calculan haciendo una XOR del byte actual con un acumulador de CRC.

```
public static int calcByte(int crc, int b)
{
  crc = crc ^ (int)b << 8;
  for (int i = 0; i < 8; i++)
  {
    if ((crc & 0x8000) == 0x8000)
      crc = crc << 1 ^ 0x1021;
    else
      crc = crc << 1;
  }
  return crc & 0xffff;
}

public static int calc(byte[] packet, int index, int count)
{
  int crc = 0;
  int i;
  while (count > 0)
  {
    crc = calcByte(crc, packet[index++]);
    count--;
  }
  return crc;
}
```

8.6 CONVERSIÓN DE LA INFORMACIÓN

La información recibida desde los motes debe transformarse hasta convertirse en una información útil que pueda ser entendida por la persona a la que tiene que prestar un servicio.
Aquí se explica por qué y cómo se realiza la conversión de los datos a unidades de ingeniería**.**

8.6.1 CONVERSIÓN DEL VOLTAJE DE LA BATERÍA

Para todas las placas estudiadas la conversión de la lectura realizada en la batería del nodo es del mismo tipo.

$$BV = \frac{RV \times ADC\_FS}{dato}$$

Donde:
BV = voltaje de la batería
ADC_FS = 1023
RV = voltaje de referencia del Micaz 1.252352 voltios
dato = es la lectura del canal 1

```
/* Convierte la lectura del voltaje a mV */
float mts3x0_convert_battery(float vref)
{
        float x = (float)vref;
   float vdata = (float) (1252352 / x);
   return vdata;
}
```

8.6.2 CONVERSIÓN DE LA TEMPERATURA DEL TERMISTOR EN MTS310

Para leer la temperatura del termistor del que dispone la placa MTS310 se debe calcular primero la resistencia de éste.

$$Rthr = \frac{R1 \times (ADC\_FS - dato)}{dato}$$

Donde R1 = 10 K.

```
/* calcula la resistencia del termistor para luego sacar la temperatura*/
float mts3x0_convert_thermistor_resistance(float thermistor)
{
   float x = (float)thermistor;
   float vdata = 10000* (1023-x) /x;
   return vdata;
}
```

Una vez calculada la resistencia se puede obtener la medida de temperatura en ºC.

$$Temp = \frac{1}{\left(a + b * \log(Rthr) + c * \log(Rthr)^3\right)} - 273.15$$

```
/* calcula la temperatura en grados centigrados */
float mts3x0_convert_thermistor_temperature(float thermistor)
{
   float temperature, a, b, c, Rt;
   a  = 0.001307050;
   b  = 0.000214381;
   c  = 0.000000093;
   Rt = mts3x0_convert_thermistor_resistance(thermistor);

   temperature = 1 / (a + b * log(Rt) + c * pow(log(Rt),3));
   temperature -= 273.15;   // Convierte de Kelvin a Celcius

   return temperature;
}
```

8.6.3 CONVERSIÓN DEL FOTOSENSOR EN MTS310

La cantidad de luz leída por el sensor Clairex CL94L viene determinada en mv y la manera de obtener la lectura es la siguiente.

$$Light = \frac{dato * vref}{1023}$$

```
/* Convierte la lectura del fotosensor a mV */
float mts3x0_convert_light(float light, float vref)
{
   float Vbat = mts3x0_convert_battery(vref);
   float Vadc = (float) (light * Vbat / 1023);
   return Vadc;
}
```

8.6.4 CONVERSIÓN DE LOS ACELERÓMETROS

Los acelerómetros en las placas estudiadas funcionan de la misma manera, por lo tanto la conversión también. Los acelerómetros son ADXL202E y tienen una sensibilidad aproximada de 167mv/g.

$$AcelX = \frac{\frac{(calplus - calmi)}{2} - (calplus - dato)}{\frac{(calplus - calmi)}{2}} \qquad AcelY = \frac{\frac{(calplus - calmi)}{2} - (calplus - dato)}{\frac{(calplus - calmi)}{2}}$$

Donde los valores calplus y calmi son valores provenientes de la calibración del acelerómetro. La salida de los acelerómetros pasa a ser dada en g.

```
/* Convierte la lectura del acelerometro en el eje X a mV */
float mts3x0_convert_accel_x(float data)
{
   float minus_one_calibration;
   float plus_one_calibration;

   float zero_value;
   float reading;

   minus_one_calibration = 0x1B9;
   plus_one_calibration = 0x21D;

   zero_value =  ( plus_one_calibration - minus_one_calibration ) / 2;
   reading =   (zero_value - (plus_one_calibration - data) ) / zero_value;

   return reading;
}

/* Convierte la lectura del acelerometro en el eje Y a mV */
float mts3x0_convert_accel_y(float data)
{
   float minus_one_calibration;
   float plus_one_calibration;

   float zero_value;
   float reading;

            minus_one_calibration = 0x237;
            plus_one_calibration = 0x1D1;

   zero_value =  ( plus_one_calibration - minus_one_calibration ) / 2;
   reading =   ( zero_value - (plus_one_calibration - data) ) / zero_value;

   return reading;
}
```

8.6.5 CONVERSIÓN DEL MAGNETÓMETRO EN LA PLACA MTS310

La salida del magnetómetro mide en dos ejes igual que los acelerómetros, X e Y, y debe ser convertida a mgauss de la siguiente manera.
El sensor es Honeywell HMC1002 y tiene una sensibilidad de 3.2mv/Vex/gauss, una excitación nominal de 3.0v, un amplificador de ganancia 2262 y una entrada ADC de 22mv/mgauss.

$$MagX = \frac{dato}{1{,}023 * 2{,}262 * 3{,}2} \qquad MagY = \frac{dato}{1{,}023 * 2{,}262 * 3{,}2}$$

```
/* Convierte la lectura del magnetometro en el eje X a mgauss */
float mts3x0_convert_mag_x(float data, float vref)
{
    float Magx = data / (1.023*2.262*3.2);
    return Magx;
}

/* Convierte la lectura del magnetometro en el eje Y a mgauss */
float mts3x0_convert_mag_y(float data, float vref)
{
    float Magy = data / (1.023*2.262*3.2);
    return Magy;
}
```

8.6.6 CONVERSIÓN DE LA PRESIÓN Y TEMPERATURA EN MTS4X0

El sensor de que dispone la placa MTS4X0 para medir la presión es el Intersema MS5534A y mide la presión barométrica en mbar y la temperatura en ºC.

$$UT1 = (8 * C5 + 20224)$$

$$dT = Temp - UT1$$

$$OFF = \left(C2 * 4 + \left(\frac{(C4 - 512) * dT}{1024} \right) \right)$$

$$SENS = C1 + \frac{C3 * dT}{1024} + 24576$$

$$\Pr e = \frac{\frac{(sens * (dato - 7168))}{16384} - OFF}{32} - 250$$

Donde C1 a C5 son medidas tomadas del sensor correspondientes a la calibración.

```
/*convierte a milibares la presion leida con el sensor INTERSEMA */
float mts4x0_convert_intersemapressure(float cal_word1, float cal_word2, float cal_word3, float cal_word4, float intersemapressure,
float intersematemp)
{
        float UT1,dT;
        float OFF,SENS,X,Press;
        unsigned short int PressureData, TempData;
        unsigned short int calibration[4];      //intersema calibration words
        unsigned short int C1,C2,C3,C4,C5,C6;   //intersema calibration coefficients


        calibration[0] = cal_word1;
        calibration[1] = cal_word2;
        calibration[2] = cal_word3;
        calibration[3] = cal_word4;
        PressureData = intersemapressure;
        TempData = intersematemp;

        C1 = calibration[0] >> 1;
        C2 = ((calibration[2] & 0x3f) << 6) | (calibration[3] & 0x3f);
        C3 = calibration[3] >> 6;
        C4 = calibration[2] >> 6;
        C5 = ((calibration[0] & 1) << 10) | (calibration[1] >> 6);

        UT1=8 * (float)C5 + 20224;
        dT = (float)TempData - UT1;
        OFF = (float)C2 * 4 + (((float)C4 - 512.0) * dT) / 1024;
        SENS = (float)C1 + ((float)C3 * dT) / 1024 + 24576;
        X = (SENS * ((float)PressureData - 7168.0)) / 16384 - OFF;
        Press = X / 32.0 + 250.0;

        return (float)Press;
}
```

Para poder medir la presión barométrica debemos calcular la temperatura con el mismo sensor.

$$Temp = \frac{200 + \frac{(dato - (8 * C5 + 20224)) * (C6 + 50)}{1024}}{10}$$

```
/* Convierte a º C la temperatura leida con el sensor INTERSEMA */
float mts4x0_convert_intersematemp(float cal_word1, float cal_word2, float intersematemp)
{
        float UT1,dT,Temp;
        unsigned short int TempData;
        unsigned short int calibration[2];              //intersema calibration words
        unsigned short int C5,C6;                       //intersema calibration coefficients

        TempData = intersematemp;
        calibration[0] = cal_word1;
        calibration[1] = cal_word2;

        C5 = ((calibration[0] & 1) << 10) | (calibration[1] >> 6);
        C6 = calibration[1] & 0x3f;

        UT1= 8 * (float)C5 + 20224;
        dT = (float)TempData - UT1;
        //temperature (degCx10)
        Temp = 200.0 + dT * ((float)C6 + 50.0) / 1024.0;
  //temperature (degC)
        Temp /=10.0;

        return (float)Temp;
}
```

8.6.7 CONVERSIÓN DE LA LUZ EN MTS4X0

El sensor de que dispone la placa MTS4X0 para medir la cantidad de luz es el Taos tsl2250 y mide la cantidad de luz en lux.

```
/* Convierte a lux la lectura del sensor de luz */
float mts4x0_convert_light(float taosch00, float taosch01)
{
        unsigned short int taosch0, taosch1;
        unsigned short int ChData,CV1,CH1,ST1;
        int ACNT0,ACNT1;
        float CNT1,R,Lux;
        taosch0 = taosch00;
        taosch1 = taosch01;

        ChData = taosch0 & 0x00ff;
        if (ChData == 0xff) return -1.0; // Taos Ch0 data: OVERFLOW
        ST1 = ChData & 0xf;
        CH1 = (ChData & 0x70) >> 4;
        CV1 = 1 << CH1;
        CNT1 = (int)(16.5 * (CV1-1)) + ST1 * CV1;
        ACNT0 = (int)CNT1;

        ChData = taosch1 & 0xff;
        if (ChData == 0xff) return -1.0; // Taos Ch1 data: OVERFLOW
        ST1 = ChData & 0xf;
        CH1 = (ChData & 0x70) >> 4;
        CV1 = 1 << CH1;
        CNT1 = (int)(16.5 * (CV1-1)) + ST1 * CV1;
        ACNT1 = (int)CNT1;

        R = ((float)ACNT1) / ((float)ACNT0);
        Lux = (float)ACNT0 * 0.46 / exp(3.13 * R);

        return (float)Lux;
}
```

8.6.8 CONVERSIÓN DE LA HUMEDAD Y TEMPERATURA EN MTS4X0

El sensor de que dispone la placa MTS4X0 para medir la humedad es el Sensirion SHT11 y mide la humedad en % y la temperatura en ºC.

$$Temp = -38{,}4 + 0{,}0084 * dato$$

$$Hum = (Temp - 25) * (0{,}01 + 0{,}00008 * dato) + (-4 + 0{,}0405 * dato - 0{,}0000028 * dato * dato)$$

```
/* Convierte a % relativo la lectura del sensor de humedad */
float mts4x0_convert_humidity(float humidity, float temp)
{
        float fTemp,fHumidity;
        unsigned short int HumData, TempData;

        HumData = humidity;
        TempData = temp;

        fTemp = -38.4 + 0.0098 * (float)TempData;
        fHumidity =  -4.0 + 0.0405 * HumData -0.0000028 * HumData * HumData;
        fHumidity= (fTemp-25.0)* (0.01 + 0.00008 * HumData) + fHumidity;

        return (float)fHumidity;
}
float mts4x0_convert_temp(float temp)
{
        float fTemp;
        unsigned short int TempData;

        TempData = temp;

        fTemp = -38.4 + 0.0098 * (float)TempData;
        return (float)fTemp;
}
```

8.6.9 CONVERSIÓN DE LOS DATOS DEL GPS

Los conversión de los datos leídos de la placa MTS420 a través del módulo GPS incorporado es el siguiente, además los datos que no necesitan ser convertidos.

```
/* Obtencion de los segundos en decimal */
double mts420_convert_dec_sec(XbowSensorboardPacket *packet)
{
        XSensorMTS420GPSData *data;

        data = (XSensorMTS420GPSData *) packet->data;
        return (data->dec_sec)/1000.0;
}

/* Obtencion de la latitud en segundos */
double mts420_convert_Lat_dec_min(XbowSensorboardPacket *packet)
{
        XSensorMTS420GPSData *data;
        data = (XSensorMTS420GPSData *) packet->data;
        return (data->Lat_dec_min)/10000.0;
}

/* Obtencion de la longitud en segundos */
float mts420_convert_Long_dec_min(XbowSensorboardPacket *packet)
{
        XSensorMTS420GPSData *data;
        data = (XSensorMTS420GPSData *) packet->data;
        return (data->Long_dec_min)/10000.0;
}

/*Indicador note o sur */
char mts420_convert_NS_ind(XbowSensorboardPacket *packet)
{
        XSensorMTS420GPSData *data;
        data = (XSensorMTS420GPSData *) packet->data;
        return ((data->NSEWind>>4)==0)?'S':'N';
```

```
}

/* Indicador este oeste */
char mts420_convert_EW_ind(XbowSensorboardPacket *packet)
{
        XSensorMTS420GPSData *data;
        data = (XSensorMTS420GPSData *) packet->data;
        return ((data->NSEWind&0xf)==0)?'E':'W';
}
```

8.6.10 EJEMPLO DE LA CONVERSIÓN DE DATOS EN UN MENSAJE

El mensaje original es este:

7E 42 FF FF 00 7D 5D 1D 86 01 01 02 8D 01 76 03 08 19 0E AE 98 B2 67 AD CF B5 A8 67 46 4A C7 00 99 00 E8 01 07 8A 22 39 7E

La placa es una MTS420 y los datos obtenidos son:

Tabla 23 Ejemplo de datos obtenidos en un mensaje tipo

Campo	Valor
id	01
vref	018D
humidity	0376
temp	1908
intersema calibration words(1..4)	AE0E,B298,AD67,B5CF
intersematemp	67A8
intersemapressure	4A46
taosch0 y taosch1	00C7, 0099
accel_x y accel_y	01E8, 078A
CRC	2239

Convertidos los datos leídos a unidades de ingeniería queda:

Tabla 24 Ejemplo de datos convertidos en un mensaje tipo

Campo	Valor
node id	1
battery	3154 mv
humidity	29 %
temperature	24 ºC
intersema temperature	24 ºC
intersema pressure	1001 mbar
light	122.775696 lux
X-axis Accel	183.333344 mg
Y-axis Accel	7416.243652 mg

9 APLICACIÓN BAJO TECNOLOGÍA IMOTE2

Una de las nuevas gamas de productos para el desarrollo de redes de sensores inalámbricas es la familia Imote2. Comparada con las otras gamas tiene un mayor número de características, ya que su posterior salida al mercado permitía disponer de dispositivos más avanzados en su diseño. En contraposición, dispone de una gama menor de placas acoplables, como pueden ser placas sensoras.
Uno de los principales objetivos de una WSN es obtener datos del medio a través de sensores. Si no disponemos de sensores integrados, una de las formas de cumplir el objetivo es usar sensores conectados al conversor analógico digital (ADC) de que disponen los Imote2.
Un ADC toma un voltaje de entrada analógico y después de cierto tiempo produce un código de salida digital que representa la entrada analógica. Esta señal digital ya puede ser tratada por el mote y a partir de aquí trabajar de la misma forma dentro de la WSN.
Dependiendo de los sensores que deseen utilizarse conectados al ADC, dentro del software del Imote2 y del gateway deberán realizarse las adaptaciones pertinentes para que los datos sean leídos y procesados correctamente.

9.1 TECNOLOGÍA IMOTE2

El Intel mote 2 es un nodo avanzado para WSN. Esta plataforma está construida alrededor de un procesador de bajo consumo PXA271 Intel XScale. Este procesador puede operar a bajos voltajes (0.85 v) y a baja frecuencia (13 Mhz). Lleva integrado 256 KB de memoria SRAM dividida en 4 bancos de 64 KB. El PXA271 es un módulo multichip que incluye tres chips en el mismo paquete, el procesador, 32 MB SDRAM y 32 MB de flash. Incluye también varias opciones de I/O.
Además, integra un chip de radio 802.15.4 (ChipCon 2420) y una antena de 2.4 GHz. Soporta un envio de 250 kb/s con 16 canales en la banda de 2.4 GHz. La antena integrada provee de un rango nominal de unos 30 metros.

Figura 51 Plataforma Imote2

9.1.1 PLACA SENSORA ITS400

Desarrollada por el laboratorio Intel Research y licenciada por Crossbow, la placa ITS400 ha sido diseñada para interactuar con la plataforma Imote2. Esta contiene un acelerómetro de tres ejes (ST Micro LIS3L02DQ 3d 12 bit +-2g), un sensor avanzado de temperatura y humedad (Alta precision, +-3ºC Sensirion SHT15), un sensor de luz (TAOS TSL2651) y un convertidor analógico digital de cuatro canales (Maxim MAX1363).

Figura 52 Placa sensora para Imote2

9.1.2 GATEWAY IIB2400

Esta placa esta diseñada para poder conectarse mediante cable USB o JTAG al Imote2. Gracias a ella nos podemos conectar desde un PC al Imote2.

Figura 53 Placa de desarrollo y conexión para Imote2

Como veremos más adelante gracias a la instalación de Linux en el Imote2 y a la posibilidad de utilizar un driver que nos permite utilizar la conexión directa entre el PC y el Imote2 como si fuera una conexión IP, esta placa sólo nos será útil en los comienzos para poder configurar dicho driver.

9.2 SISTEMAS OPERATIVOS POSIBLES PARA IMOTE2

Para la tecnología Imote2 podemos usar varios sistemas sobres los que correr nuestras aplicaciones.

- TinyOS 1.x: Ésta es la distribución original de software para Imote2 y la que más gente está usando. Tiene un elevado grupo de características y está bastante probado, ya que mucha gente ha realizado numerosas aplicaciones para él. Soporta redes de bajo consumo. La programación es soportada por el cargador bajo USB que viene con la versión que no es para .net. La programación en TinyOS tiene una curva de aprendizaje con una pendiente elevada.
- TinyOS 2.x: La nueva versión del sistema operativo TinyOS tiene menos características y está menos testeado que la versión 1.x, aunque poco a poco se va intentando solucionar esto gracias a que la gente lo está desarrollando. No soporta el cargador de TinyOS y necesita un cable JTAG para ser programado. La programación en TinyOS tiene una curva de aprendizaje con una pendiente elevada también.
- SOS 1.x: Este sistema operativo está siendo usado por la gente de la Universidad de Yale. Se necesita un cable JTAG para programar.
- Linux: La versión basada en el Kernel 2.6 está disponible para este dispositivo. Su uso está expandiéndose y numerosos grupos tienen discusiones activas sobre ello. Tiene un consumo más elevado comparado con las versiones de TinyOS pero las características disponibles son mucho más estándar. Las instrucciones para la radio todavía son muy básicas. Se necesita un cable JTAG para un flasheo inicial del kernel de Linux, pero después de esto se pueden transferir datos y programas vía cable USB. Se necesitan conocimientos de herramientas de desarrollo de compilamiento cruzado.
- .NET: Probablemente el más adecuado para aquellos que están familiarizados con las herramientas de Microsoft MS .NET Framework y Visual Studio. Un agradable GNU basado en el entorno de programación. Se requiere un mayor consumo de energía que con TinyOS y además, la velocidad de ejecución es menor. Necesitan un especial sistema de carga bajo USB que sólo viene implementado en la versión para .net.

La opción que aporta más opciones para trabajar con los dispositivos Imote2 es Linux. Probablemente tenga menos desarrolladas las características de red inalámbrica, pero nos da más características generales que compensan esa carencia.

9.3 FUNCIONAMIENTO BAJO LINUX

Imote2 es una plataforma empotrada desarrollada por investigadores de Intel. Se compone de un procesador Intel PXA271 XScale, un chip de comunicaciones funcionando bajo el protocolo 802.15.4 Chipcon CC2420 y varios slots de expansión. El PXA271 consiste en un procesador combinado con 32Mb de memoria flash y 32 Mb de memoria SRAM.
El sistema de archivos de los 32 Mb de memoria flash disponible se divide como muestra la figura:

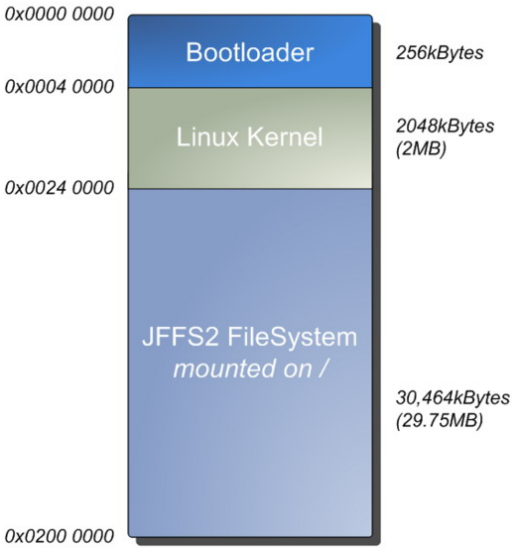

Figura 54 Sistema de archivos Linux en Imote2

Sistema de arranque (Bootloader): también llamado BLOB, es la primera parte del software que corre cuando el sistema es arrancado. Se encarga básicamente de lanzar el sistema y de inicializar varios de sus componentes. Por ejemplo, cuando el procesador es encendido debe de inicializarse el sistema de memoria, la configuración del sistema de reloj, GPIOs y los sistemas periféricos.
Una vez que el sistema de arranque ha finalizado de iniciar el sistema, éste pasará a lanzar el arranque del kernel de Linux. BLOB tiene varios módulos adicionales que permiten programar el sistema vía puerto serie, ethernet o USB, aunque hay ciertas partes que no lo permiten y deben realizarse vía cable JTAG .

Kernel de Linux: El kernel de Linux es almacenado en un área diferenciada de la memoria flash, en vez de en la parte correspondiente al sistema de ficheros. Tradicionalmente el kernel es leído desde el sistema de ficheros montado por el sistema de arranque y entonces es cargado. Si el kernel es más extenso que la ventana de 2 Mb asignada para ello, el kernel será estropeado al programar el sistema de archivos.

Sistema de ficheros JFFS2: La raíz principal del sistema de ficheros en el iMote2 es almacenada al final del rango de la memoria flash destinada al sistema de ficheros JFFS2. JFFS2 es un sistema de ficheros con soporte para transacciones especializado en memorias flash que incorpora compresión, donde se utilizan tres algoritmos: zlib, rubin y rtime; wear levelling, técnica que prolonga la vida de memorias borrables como la memoria flash; y recolector de basura, mecanismo por el cual los bloques sucios se convierten en bloques libres. La memoria flash se agota después de unas 10.000 operaciones de escritura por lo que se considera necesario que se disponga de wear levelling.

9.3.1 COMO COMPILAR LINUX PARA Imote2

La aplicación desarrollada basada en aparatos imote2 va ha requerir la instalación de Linux en ellos. Para poder instalar este software dentro de los aparatos debe compilarse adecuadamente, siguiendo los siguientes pasos.

- Disponer del compilador adecuado. Se debe hacer con el compilador cruzado arm-linux-gcc, con la versión 3.4.1 que podemos encontrar en http://www.handhelds.org/download/projects/toolchain/

```
bzip2 -d arm-linux-gcc-3.4.1.tar.bz2
tar xvf arm-linux-gcc-3.4.1.tar
export PATH=$PATH:/usr/local/arm/3.4.1/bin/
```

- Actualizar el PATH para que el compilador funcione correctamente

```
export PATH=$PATH:/usr/local/arm/3.4.1/bin/
```

- Crear un directorio para desarrollar la operación.

```
rm -rf borrar
mkdir borrar
cd borrar
```

- Descargar la versión de Linux adecuada y los parches correspondientes. La versión de Linux carece de drivers específicos que necesita el imote2. Estos drivers, como pueden ser el tosmac, especial para la comunicación inalámbrica entre motas; o led, especial para hacer lucir los leds del dispositivo; son creados al aplicar el parche antes de compilar el kernel de la versión de Linux.

```
wget ftp://ftp.kernel.org/pub/linux/kernel/v2.6/linux-2.6.24.7.tar.gz
tar zxpf linux-2.6.24.7.tar.gz
cp ../patchsg_2.6.24-rc7 .
```

- Aplicar el parche sobre el kernel correspondiente.

```
echo "----------"
echo "Building and compiling Kernel"
cd linux-2.6.24.7
echo "Applying Patches"
patch -p1 < ../patchsg_2.6.24-rc7
```

- Una vez que el código es el adecuado, hay que compilar y generar las imágenes necesarias para instalar en el Imote2.

```
echo "Building kernel"
export ARCH=arm
export CROSS_COMPILE=arm-linux-
make stargate2_defconfig
make oldconfig
make zImage

echo "Modules"
make modules
make modules_install
```

- Una vez generada la imagen, la copiamos en una carpeta para tener las tres juntas.

```
echo "Copying zImage"
cd ..
mkdir images
cd images
cp ../linux-2.6.24.7/arch/arm/boot/zImage ./
cd ..
```

- Una vez compilado el kernel debemos tener el sistema de archivos. Primero nos descargamos un sistema de archivos compatible con Imote2 y lo adecuamos.

```
wget http://ubi.cs.washington.edu/files/imote2/mirrors/platformX/SG2Release-1_0.tgz
tar zxpf SG2Release-1_0.tgz

cd ./SG2Release-1_0/buildTree
tar zxpf platx-10.tgz
cd ../../

cd ./SG2Release-1_0/src
tar zxpf root-10.tgz
cd ../../
```

- Para más tarde crear la imagen del sistema de archivos y guardarla en la carpeta de las imágenes.

```
echo "Creating system image"
./SG2Release-1_0/buildTree/platx/util/mkfs.jffs2 --output=fs.jffs2 --root=./SG2Release-1_0/src/root --pad=0x1DC0000 --
eraseblock=0x20000
mv fs.jffs2 ./images
```

- Por último se construye el sistema de arranque de la versión Linux.

```
echo "--------------------"
echo "Building blob bootloader"

cd SG2Release-1_0/src
tar zxpf blob-px2-10.tgz
cd blob-px2
make -f Makefile.cvs
make -f Makefile.cvs
ln -s ../../../linux-2.6.14 linux
./configure --host=arm-linux --with-board=stargate2 --with-linux-prefix=$PWD/linux --enable-xlli --enable-network --enable-xmodem
./configure  --host=arm-linux  --with-board=stargate2  --with-linux-prefix=/path/to/linux-2.6.14  --enable-xlli  --enable-network  --enable-xmodem#make
cp src/blob/blob ../../../images/
cd ../../../
```

9.3.2 COMO CARGAR LINUX EN imote2

Para poder trabajar correctamente con el imote2 bajo el sistema operativo Linux o bajo tinyos-2.x necesitamos instalar correctamente el software jflashmm. Es un programador de memorias flash a través de un cable genérico JTAG.
Intel dispone de una versión de programa más rápida y eficiente llamada xflash, pero en este proyecto se utiliza la versión jflashmm, que aunque más lenta es de libre distribución.
Para lograr la conexión con el imote2, se necesita el módulo IIB2400. Con este modulo se tiene acceso al imote2 mediante USB o JTAG a través de un ordenador.
Para realizar la programación de la memoria flash del dispositivo PXA27x se necesita un cable JTAG como el de la figura. Este cable dispone por un lado de un conector JTAG de 20 pines y por otro de un conector paralelo para un ordenador.

Figura 55 Cable JTAG

Las características de los pines del cable son:

Tabla 25 Características de los pines del cable JTAG

Pin	Descripción	Pin	Descripción	Pin	Descripción
1	VCC3	2	VCC3	3	-TRST
4	GND	5	TDI	6	GND
7	TMS	8	GND	9	TCK
10	GND	11	RTCK	12	GND
13	TDO	14	GND	15	-SRST
16	GND	17	DBGRQ	18	GND
19	DBGACK	20	GND		

Para pasar la imagen de Linux se deben pasar los siguientes archivos binarios con el programa jflashmm:

```
JFlashmm.exe bulbcx16 blob P 0x00000000
JFlashmm.exe bulbcx16 zImage P 0x00040000
JFlashmm.exe bulbcx16 fs.jffs2  P 0x00240000
```

Donde como se dijo antes:

- blob es el encargado de cargar el arranque.

- zImage es la imagen del kernel de la versión de Linux 2.6.
- fs.jffs2 es el sistema de ficheros.

En el caso de usar tinyOS-2.x en imote2 podemos cargar el software de forma similar con el cable JTAG comentado anteriormente ejecutando la siguiente orden.

```
JFlashmm.exe bulbcx16 main.bin P 0x00000000
```

9.3.3 CONFIGURACIONES PREVIAS EN Imote2 BAJO LINUX

Una vez cargado Linux en la plataforma Imote2 debemos realizar varias configuraciones antes de poner a funcionar el dispositivo. En el primer arranque hay que introducir una contraseña, ya que por defecto viene sin nada y por motivos de seguridad conviene que disponga de una. Mediante la orden 'passwd' podemos añadírsela.
Por defecto, esta compilación de Linux puede ser usada para la plataforma Stargate2 y la Imote2. Es conveniente ejecutar el archivo /root/configure para que el dispositivo funcione como nodo y no como puerta de enlace.

```
#!/bin/sh

isSG2

if [ $? -eq 1 ]; then
  echo Configuration is Stargate 2
  cp /etc/modules-sg2 /etc/modules
  cp /etc/init.d/banner-sg2 /etc/init.d/banner
  mknod /dev/tosmac c 240 0
  mknod /dev/led c 252 0
else
  echo Configuration is Intel Mote 2
  cp /etc/modules-im2 /etc/modules
  cp /etc/init.d/banner-im2 /etc/init.d/banner
  mv /etc/rc2.d/S10networking /etc/rc2.d/_S10networking
fi
```

Para poder utilizar los drivers específicos para una WSN, se deben cargar los módulos correspondientes. Esto se hace en el archivo /etc/modules, donde deben estar todos estos módulos para su carga en el arranque del sistema.

```
 g_ether
led

i2c-core
i2c-pxa
eeprom
pcf8574

tos_mac
i2c-dev
max1363

spi-dev
pxa2xx_spi
lis3l02dq
```

Hay varios aparatos que no crea automáticamente, por ello se realiza un .sh que hace que se creen al arranque del sistema. El archivo es /etc/init.d/creanod.sh y contiene el siguiente código.

```
mknod /dev/tosmac c 240 0
mknod /dev/led c 252 0
```

A este archivo se le da permiso de ejecución y se hace que se inicie en el arranque. En /etc/rc2.d creamos un enlace dinámico.

```
ln -s ../init.d/crearnod.sh S55crearnod
```

9.4 CAPTURA DE DATOS DESDE EL BUS I2C

Una vez se tiene cargado el sistema operativo Linux en los motes se debe desarrollar el software necesario para poder captar datos del ADC y de los sensores que éstos tengan conectados. Para ver como captar datos, hay que ver como funciona el bus al que está conectado el ADC.

9.4.1 FUNCIONAMIENTO DEL BUS I2C

El bus I2C (Inter IC) es un bus serie de dos cables desarrollado por Philips Semiconductor a principios de los años 80. Originalmente este bus fue inventado para conectar varias placas ICs a una televisión. Sin embargo, gracias al fácil uso, pasó a ser un estándar universal y ahora es usado para conectar una amplia gama de periféricos en con amplia gama de configuraciones. Inicialmente era un bus de baja velocidad, pero ha evolucionado hasta ofrecer una variedad de velocidades desde 100 KB/sec hasta 3.4 KB/sec. El bus I2C ofrece varias ventajas como ahorrar espacio en placa, ahorrar en el coste del hardware y ofrecer más facilidades de depuración.
Hoy en día el bus I2C es altamente usado en sistemas empotrados y es muy difícil encontrar placas sin un bus I2C.
El bus I2C tiene dos líneas: la línea SDA (datos) y la línea SCL (reloj). La línea SDA lleva información como la dirección, datos y respuestas, un bit en cada pulso de reloj. El emisor y el receptor se sincronizan mediante la línea de reloj. Varios aparatos I2C pueden trabajar en un solo bus I2C, ya que los aparatos son clasificados como maestro y esclavo. Un maestro es un aparato que inicia y para la transferencia y genera la señal de la línea de reloj. Un esclavo es un aparato que es diseccionado por el maestro. Cada aparato conectado al bus I2C tiene una única dirección de 7 o 10 bits que es usada para identificar ese aparato en particular.

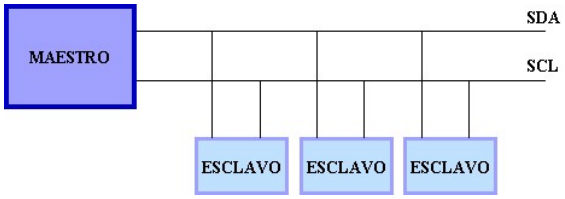

Figura 56 Bus I2C

La transferencia de datos está dividida en varias fases [RAG2005]:

- Fase inactividad: Cuando el bus I2C no está en uso, ambas líneas SDA y SCL están mantenidas en un estado alto.
- Fase de inicio: Cuando la línea SDA cambia de un nivel alto a bajo y la línea SCL es mantenida a un nivel alto, indica el inicio de una transmisión de datos. Es el maestro el encargado de realizar esta fase.
- Fase de dirección: Durante esta fase, el maestro envía la dirección del aparato esclavo seleccionado y el modo de transferencia de datos (lectura o escritura). El esclavo replica con un acuse de recibo para indicar que la fase de transferencia puede ser inicializada.
- Fase de transferencia de datos: Los datos son transmitidos bit a bit por el bus I2C. Al final de cada byte transferido, un bit de acuse es enviado del receptor al emisor.
- Fase de parada: El maestro indica este estado cambiando el estado de la línea SDA de bajo a alto mientras la línea SCL esta en estado alto.

El funcionamiento cuando un maestro necesita enviar datos a un esclavo se describe en los siguientes pasos:

1. El maestro marca la condición de inicio.
2. El maestro envía la dirección del esclavo al que desea enviar datos y además indica el modo de transferencia, en este caso escritura.
3. El esclavo envía una contestación al maestro.
4. El maestro envía la dirección donde los datos han de ser escritos en el esclavo.
5. El esclavo envía una contestación al maestro.
6. El maestro envía datos para ser escritos en el esclavo por la línea SDA.

7. Al final de cada byte transferido, el esclavo envía un bit de recibo.

8. Los dos pasos anteriores son repetidos hasta que todos los datos requeridos han sido escritos. La dirección de escritura se incrementa automáticamente.

9. El maestro envía la señal de parada.

Figura 57 Escribir datos en el bus I2C

El funcionamiento cuando un maestro necesita leer datos de un esclavo se describe en los siguientes pasos:

1. El maestro marca la condición de inicio.

2. El maestro envía la dirección del esclavo del que desea recibir datos y además indica el modo de transferencia, en este caso lectura.

3. El esclavo envía una contestación al maestro.

4. El maestro envía la dirección desde donde los datos han de ser leídos en el esclavo.

5. El esclavo envía una contestación al maestro.

6. El esclavo envía datos para ser escritos en el esclavo por la línea SDA.

7. Al final de cada byte transferido, el maestro envía un bit de recibo.

8. Los dos pasos anteriores son repetidos hasta que todos los datos requeridos han sido leídos. La dirección de lectura se incrementa automáticamente. Sin embargo para reconocer el último byte el maestro no envía recibo. Esto impide que el esclavo envíe más datos al bus.

9. El maestro envía la señal de parada.

Figura 58 Leer datos del bus I2C

9.4.2 RECOGIDA DE DATOS DEL ADC

Un conversor analógico digital es un dispositivo electrónico que convierte una entrada analógica de voltaje a un número digital. El ADC de que dispone la placa sensora ITS400 es el modelo MAX1363 de la casa Maxim y dispone de 4 canales de propósito general [MAX2008].
Además dispone de una resolución de 12 bits. La resolución de un conversor indica el número de valores discretos que éste puede producir sobre un rango de valores de voltaje. Generalmente es expresado en bits. Por ejemplo, un conversor que codifica una entrada analógica de 4096 valores discretos (0..4095) tiene una resolución de 12 bits: o sea, 2 elevado a 12.
Cada canal soporta 0-3 V. de señal de entrada. Esto debe tenerse muy en cuenta ya que si conectamos a estas entradas sensores que den una salida superior, ésta puede dañar el ADC y la placa sensora.
Para poder acceder al bus dentro del sistema operativo Linux se crea un programa en C desde donde se leen los datos de ADC. Primero se abre el dispositivo.

```
open("/dev/i2c-0", O_RDWR)
```

Se le indica la dirección donde está situado el dispositivo que queremos manejar, en este caso el ADC MAX1363 está situado en la dirección 0x34.

```
ioctl(file, I2C_SLAVE_FORCE, addr)
```

Una vez que ya estamos conversando con el dispositivo se le manda la configuración con la que se desea que trabaje.

```
write(file, &conf, 1)
```

Y a partir de aquí leemos los datos.

```
read(file, &a, 4)
```

El dispositivo dispone de 4 canales y de varias formas de leer los canales. Para que la lectura se adapte a lo que se desea realizar se debe configurar mediante el bit de configuración siguiendo la siguiente tabla.

Tabla 26 Bit de configuación del bus I2C

Bit	Nombre	Descripción
7 (MSB)	CONFIG	El byte de configuración siempre empieza por 0
6	SCAN1	SCAN1, SCAN0 = [0,0] escanea desde el canal 0 hasta el indicado en CS1, CS0.
5	SCAN0	SCAN1, SCAN0 = [0,1] convierte un sólo canal elegido en CS1, CS0 8 veces. SCAN1, SCAN0 = [1,0] modo monitor desde el canal 0 hasta el elegido en CS1, CS0. SCAN1, SCAN0 = [1,1] convierte un sólo canal elegido en CS1, CS0.
4	CS3	
3	CS2	CS3, CS2 = [1, 1] habilita volver a leer en el modo monitor
2	CS1	Selecciona el canal superior si SCAN = [0,0] o SCAN = [1,0]
1	CS0	Selecciona el canal de conversión si SCAN = [0,1] o SCAN = [1,1]
0	SE/(NOT)DIF	1 = entradas single-ended. 0 = entradas diferenciales.

9.4.3 TRANSMISIÓN DE DATOS BAJO LINUX

La transmisión de datos inalámbricamente en Linux se realiza usando el driver tosmac, introducido en Linux al realizar la compilación.

Para enviar un mensaje abrimos el dispositivo.

```
Open(TOSMAC_DEVICE, O_RDWR)
```

Y mandamos la información.

```
write(tosmac_dev, (TOS_Msg*)&send_pkt, sizeof(TOS_Msg))
```

Para finalmente cerrar el dispositivo.

```
close (tosmac_dev);
```

Para recibir un mensaje algo similar. Abrimos el dispositivo.

```
open(TOSMAC_DEVICE, O_RDWR)
```

Leemos la información recibida.

```
read(tosmac_dev, &recv_pkt, sizeof(TOS_Msg))
```

Para finalmente cerrar el dispositivo.

```
close (tosmac_dev);
```

9.5 FUNCIONAMIENTO DE LA APLICACIÓN

La aplicación desarrollada consiste en conectar un sensor, un inclinómetro, al ADC de uno de los Imote2, el cual enviará la información a otro de los Imote2 conectado a una máquina Linux. En esta máquina se recogerán los datos leídos.

Los inclinómetros son sensores que detectan el movimiento angular con una determinada sensibilidad y estabilidad.

Las mediciones recaen en la referencia más estable, el vector vertical de la gravedad. Se realiza la conexión a un sistema de adquisición de datos simplemente, dándole un conocimiento absoluto de qué está pasando en cualquier momento.

El inclinómetro usado para las pruebas es el modelo LCF196 de la casa Jewell Instruments [JEW2008]. Es un inclinómetro biaxial con un rango de 14.5º a 90.0º de gran precisión desarrollado para ser usado en aplicaciones donde altos niveles de choque y vibración están presentes. Mide en dos ejes y tiene una forma cilíndrica de 22 mm de diámetro en acero inoxidable.

Figura 59 Inclinómetro LCF196 de Jewell Instruments

La tensión de alimentación de este dispositivo es desde +-12 a +-18 voltios. Como los dispositivos Imote2 no pueden alimentar a este tipo de sensor, se utiliza una fuente de alimentación estándar para suministrar la alimentación.
La tensión de salida del sensor es +-5 voltios. Teniendo en cuenta que el ADC del dispositivo Imote2 sólo admite una tensión de entrada de 0-3 voltios, esta tensión debe ser adaptada para el correcto funcionamiento del ADC.
Se ha realizado un circuito adaptador de la señal con el que se conseguía pasar de la tensión de salida del sensor a la tensión necesaria en la entrada del ADC mediante amplificadores operaciones y resistencias. La tensión de alimentación de estos amplificadores es la misma que la de alimentación de sensor, por lo tanto con una única fuente podemos alimentar al sensor y el circuito adaptador.

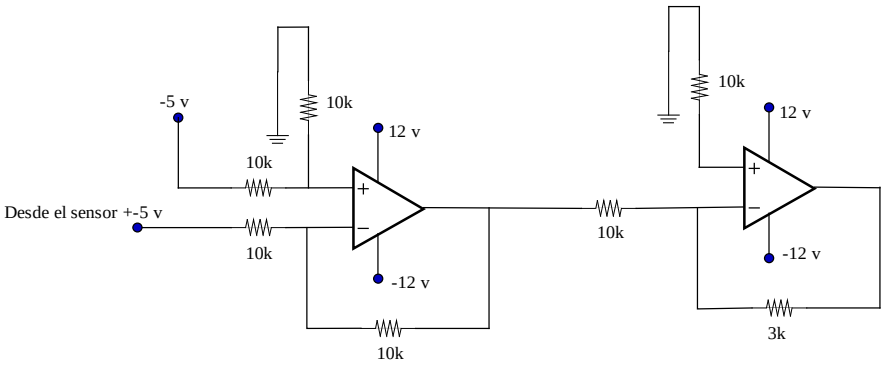

Figura 60 Circuito adaptador de señal

Una vez adaptada la información, ésta es leída por el Imote2 a través del ADC y enviada hacia el otro dispositivo que gracias a la comunicación IP existente a través del cable USB hace que el Servidor al que está conectado pueda acceder a estos datos y leerlos.

10 CONCLUSIONES

En la actualidad, la tecnología de sensores inalámbricos se encuentra en plena efervescencia. Es por ello que se espera que en los próximos años siga evolucionando de forma significativa. Pero de momento las WSN presentan tantas capacidades potenciales como limitaciones reales aún sin resolver. El tipo de hardware y software que finalmente dominará en este tipo de redes es todavía desconocido, si bien las actuales capacidades de energía, cómputo y comunicación, limitan la manera en que se ha de construir su software.

Las investigaciones apuntan en direcciones relativamente claras. Resulta fundamental disponer de sensores de pequeño tamaño y gran autonomía, por lo que se investiga tanto en técnicas de miniaturización como de bajo consumo de potencia.

Si bien el uso de pequeñas baterías es la opción más simple, resultaría clave disponer de otro tipo de técnicas para la alimentación de dichos sensores. Los problemas encontrados en cuanto a alimentación han sido el mayor obstáculo en esta investigación, ya que el consumo de los dispositivos superaba ampliamente la energía disponible en los nodos y esto llevaba a una corta duración en la vida de la red.

La utilización de este tipo de redes en aplicaciones donde la fiabilidad deba ser crítica todavía está pendiente de la mejora de éstas, ya que una cantidad no muy elevada de datos se pierde en el proceso o es erróneo y eso lleva a problemas en la gestión de los datos.

La facilidad de uso de estos dispositivos brilla por su ausencia, la multitud de plataformas y de sistemas operativos con los que trabajar dentro de ellas hace imposible tener un conocimiento de todos los tipos de nodos existentes. Además, el coste de aprender los lenguajes de programación en los que se desarrollarán los programas también es elevado. Aunque este punto cada día tiene menos inconvenientes, ya que las últimas tecnologías permiten incluso usar sistemas operativos de alto nivel como Linux donde el coste asociado al conocimiento de éste es menor. Adicionalmente, también son muy importantes los protocolos de comunicación y los interfaces radioeléctricos (Zigbee, Bluetooth, Wi-Fi, WiMAX, FHSS,...), especialmente conforme aumenta el número de dispositivos y las potenciales interferencias. Actualmente se dispone de un rango amplio de interfaces que solucionan nuestras necesidades, sobre todo con Zigbee que es el que mejor se adapta a las necesidades de este tipo de redes.

Ya existen investigaciones sobre protocolos de comunicaciones en redes tipo malla, y esto hace que actualmente existan muchos tipos de protocolos con formas diferentes de trabajar. A pesar de ello hay una perdida de datos, aunque reducida, que hace que estos protocolos deban ser mejorados. Tambien se busca una reduccion en el consumo de energia, por lo tanto cuanto menor sea la transmisión de datos innecesaria, menor sera el consumo.

11 ANEXOS

11.1 ANEXO 1: CONFIGURACIÓN DEL MIB600

Uno de los posibles Gateway a utilizar en una red de MicaZs es el MIB600 [XBOW2008-5]. Para configurar el dispositivo y hacerlo accesible desde la red se deben seguir los siguientes pasos.
Instalar en un PC conectado en la misma red el software 'Device Installer' de Lantronix. No sabemos la IP por defecto de que dispone el dispositivo y se va ha obtener gracias a este programa. Para detectar la IP pulsamos la opción: buscar aparatos en la red.

Figura 61 Software para detectar IP del MIB600

Como el programa indica, el aparato viene de serie con una IP en un rango diferente al de nuestro PC y por lo tanto deberemos cambiarla. En el menú Device debemos seleccionar 'Assign IP Address'. Lo primero es decirle que dirección MAC tiene para que lo localice.

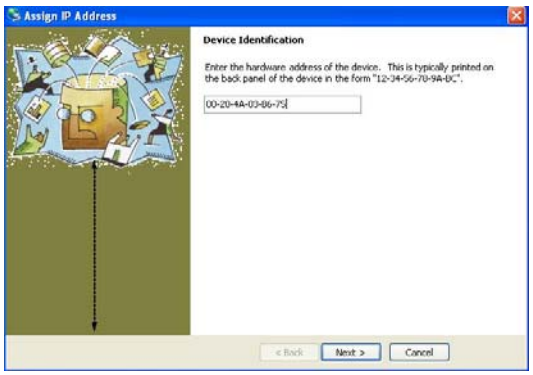

Figura 62 Introducción de la MAC para realizar búsqueda de IP

Después le indicamos que queremos una IP estática.

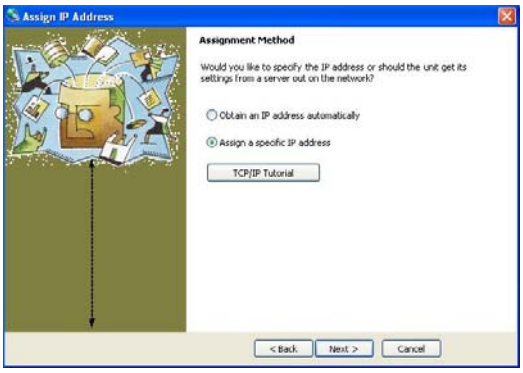

Figura 63 Asignar IP estática al dispositivo MIB600 (I)

Para finalmente asignar la IP deseada y acabar con el proceso.

Figura 64 Asignar IP estática al dispositivo MIB600 (II)

11.2 ANEXO 2: CONFIGURACIÓN DEL STARGATE

11.2.1. CONFIGURACIÓN DE CONEXIÓN A RED

Mediante el cable null modem serial conectar el Stargate [XBOW2008-6] con el PC.

Figura 65 Cable null modem serial para conectar al Stargate

Desde un ordenador Windows arrancar HyperTerminal y crear una nueva conexión que se llamara Stargate. Las características de la conexión son:

Tabla 27 Características de la conexión con HyperTerminal

Característica	Valor
Bits per second	115200
Data bits	8
Parity	0
Stop bits	1
Flow control	None

Figura 66 Características de la conexión con HyperTerminal

Ahora ya se puede ver como arranca la maquina Linux hasta que nos pida login y password: root rootme por defecto.
Una vez con acceso al sistema se puede configurar la IP del dispositivo Ethernet LAN o la del dispositivo wireless, para poder acceder desde nuestra red.

COMO CONFIGURAR IP TARJETA ETHERNET FIJA.

La forma mas conveniente de configurar la IP del Stargate es editar el archivo networking de /etc/init.d como muestra la figura.

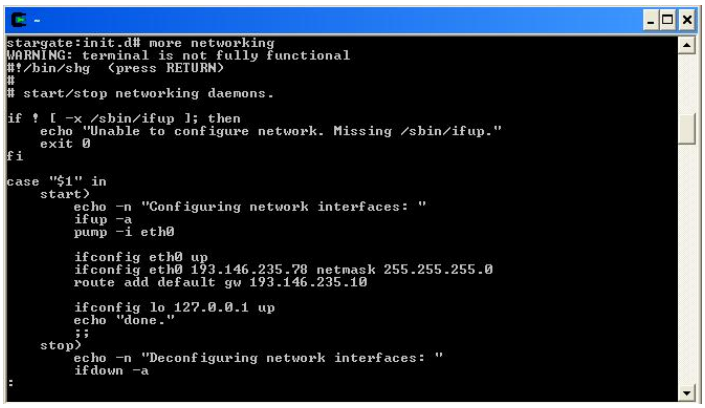

Figura 67 Configuración de la IP estática del Stargate

Vemos como queda configurada la red con el comando ifconfig.

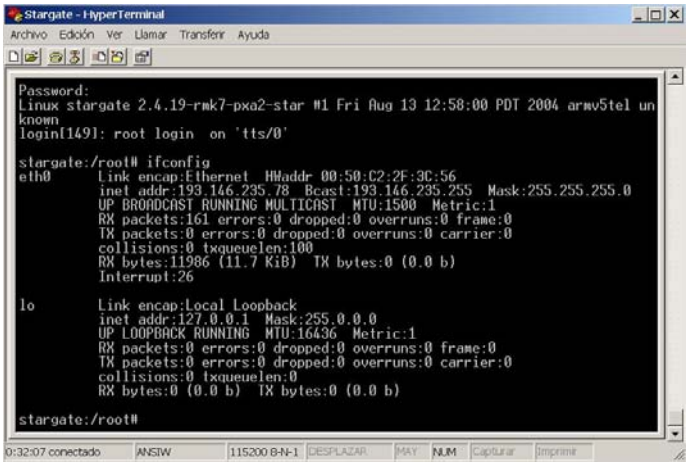

Figura 68 Comprobación del cambio de IP en Stargate

COMO CONFIGURAR IP TARJETA INALÁMBRICA.

Lo primero que se debe hacer es identificar la tarjeta inalámbrica. La más usada para este dispositivo es de la marca "AmbiCom" y el modelo "WL1100C 802.11b CF-Card".

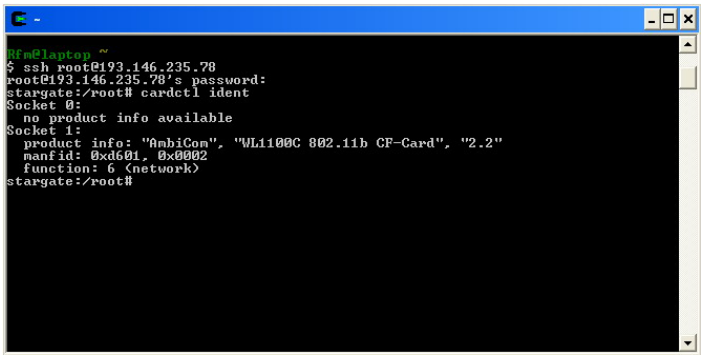

Figura 69 Identificación de la tarjeta inalámbrica en el Stargate

Ahora podemos configurar los parámetros de la tarjeta de red, situándonos en el directorio "/etc/pcmcia" y modificando el archivo "wireless.opts", introduciendo los datos pertenecientes a la tarjeta AmbiCom 802.11 Card.

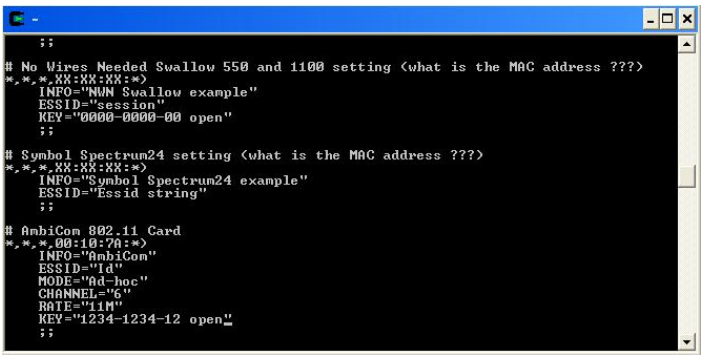

Figura 70 Configuración de la IP estática de la tarjeta inalámbrica del Stargate

Se modifica el archivo 'network.opts' configurando los valores:

Tabla 28 Parámetros para la configuración de la tarjeta de red

Característica	Valor
DHCP	n
IPADDR	193.146.235.75
NETMASK	255.255.255.0
NETWORK	193.146.235.0
BROADCAST	193.146.235.225
GATEWAY	193.146.235.10

Hasta dejar el archivo así:

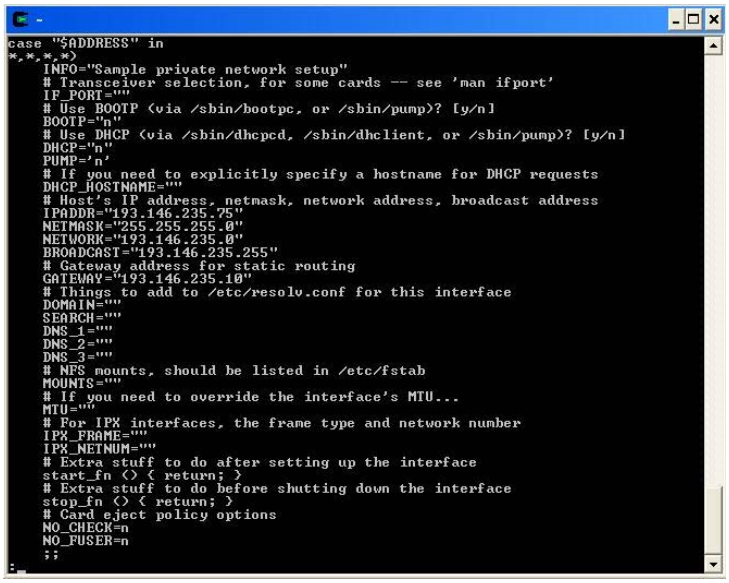

Figura 71 Visualización final del archivo de configuración

Una vez configurado se resetea el Stargate: "shutdown –r now" para que una vez ha arrancado de nuevo ya tenemos la tarjeta de red activada.

11.2.2. PUESTA EN HORA EL STARGATE

El Stargate por defecto solo puede situarse en las zonas horarias americanas de Chicago, Los Angeles, Denver y Nueva York. Para poder actualizar a una hora española debemos crear en la carpeta /usr/share/zonainfo/ una carpeta llamada Europe que acompañe a America y dentro de esta pasar el archivo de zona horario de Madrid desde otra maquina Linux.

Hay que hacer un link simbólico al archivo Madrid que enlace a /etc/localtime.

```
# ln -sf /usr/share/zoneinfo/America/New_York /etc/localtime
```

El que aquí se pone es el que esta funcionando por defecto.

Poner la hora en una maquina Linux no tiene mayor misterio que hacer que dicha máquina se sincronice con un servidor ntp. Lo que vamos a hacer es que en un script diario la máquina se sincronice con un servidor ntp y así aunque la máquina se apague o tenga un fallo de alimentación al arrancar volverá a estar puesto en hora.

En la carpeta /etc/cron.daily vamos a introducir un script con las siguientes líneas.

```
Echo "La hora se esta actualizando…."
ntpdate -b -p 8 -u 129.132.2.21
```

Como la versión de Linux es reducida y sobrepasa el registro asignado a la diferencia horaria debemos escribir en el archive /etc/init.d las siguientes líneas:

```
date --set "01/01/2008 00:00:01"
ntpdate -b -p 8 -u 129.132.2.21
ntpdate -b -p 8 -u 129.132.2.21
```

De este archivo vamos a crear un enlace dinámico a la carpeta rc2.d que vamos a llamar S55arranquentp. El perfil que usa el Stargate es el 2 que corresponde con esa carpeta.

En /dev/rc2.d/

```
ln -s ../init.d/arranquentp.sh S55arranquentp
```

Estas se ejecutaran al arrancar y así tendremos la hora al arrancar.

11.3 ANEXO 3: COMO PROGRAMAR UN MOTE MICAZ

11.3.1 COMO PROGRAMAR UN MOTE MicaZ CON EL MIB600

Para programar un MicaZ debemos situar el MicaZ correspondiente en el dispositivo MIB600 y este a su vez conectado a la misma red que el ordenador.

Dentro de la consola situarse en la misma carpeta donde este el programa que deseamos introducir en el mote, para teclear la orden correspondiente.

```
'make micaz route,lp install,0 eprb,193.146.235.77'
```

Donde:

- micaz es la plataforma
- route,lp indica que va a ser un dispositivo para red en malla y de bajo consumo.
- install,0 indica que queremos instalarlo y con el numero de nodo 0.
- eprb, 193.146.235.77 indica que el programador es el MIB600 y esta en la IP numero 193.146.235.77

Un ejemplo de cómo instalarlo es el siguiente:

```
librero:/opt/tinyos-2.x/apps/BaseStation # make micaz install,0 eprb,193.146.235
.77
mkdir -p build/micaz
    compiling BaseStationC to a micaz binary
ncc -o build/micaz/main.exe -Os -finline-limit=100000 -Wall -Wshadow -Wnesc-all
-target=micaz -fnesc-cfile=build/micaz/app.c -board=micasb -DCC2420_NO_ACKNOWLED
GEMENTS -DCC2420_NO_ADDRESS_RECOGNITION -DIDENT_PROGRAM_NAME=\"BaseStationC\" -D
IDENT_USER_ID=\"root\" -DIDENT_HOSTNAME=\"librero\" -DIDENT_USER_HASH=0xdb99ab71
L -DIDENT_UNIX_TIME=0x4772267dL -DIDENT_UID_HASH=0x5bfa02a4L -fnesc-dump=wiring
-fnesc-dump='interfaces(!abstract())' -fnesc-dump='referenced(interfacedefs, com
ponents)' -fnesc-dumpfile=build/micaz/wiring-check.xml BaseStationC.nc -lm
    compiled BaseStationC to build/micaz/main.exe
          13898 bytes in ROM
           1569 bytes in RAM
avr-objcopy --output-target=srec build/micaz/main.exe build/micaz/main.srec
avr-objcopy --output-target=ihex build/micaz/main.exe build/micaz/main.ihex
    writing TOS image
tos-set-symbols build/micaz/main.srec build/micaz/main.srec.out-0 TOS_NODE_ID=0
ActiveMessageAddressC$addr=0
    installing micaz binary using eprb
uisp -dprog=stk500 -dhost=193.146.235.77 --wr_fuse_h=0xd9 -dpart=ATmega128 --wr_
fuse_e=ff  --erase --upload if=build/micaz/main.srec.out-0
Firmware Version: 1.7
Atmel AVR ATmega128 is found.
Uploading: flash

Fuse High Byte set to 0xd9

Fuse Extended Byte set to 0xff
rm -f build/micaz/main.exe.out-0 build/micaz/main.srec.out-0
```

Figura 72 Ejemplo de programación de un MicaZ

11.3.2 COMO PROGRAMAR UN MOTE MicaZ CON EL STARGATE

También puede programarse un MicaZ con el Stargate. Igual que en el caso anterior debemos conectar el dispositivo Stargate SPB400 en la misma red que el ordenador y situarnos dentro de la consola en la misma carpeta donde este el programa que deseamos introducir en el mote y teclear la orden correspondiente.

```
$ make micaz
```

Una vez creado el ejecutable le vamos a asignar el número de nodo que debe llevar. Dependiendo de si estamos programando en tinyos-1.x:

```
$ set-mote-id build/micaz/main.srec build/micaz/main.srec.out 1
```

O si es en tinyos2.x:

```
$ tos-set-symbols build/micaz/main.srec build/micaz/main.srec.out-1 TOS_NODE_ID=1
```

Para más tarde copiar la imagen al Stargate:

```
$ scp build/micaz/main.srec.out root@<Stargate IP Address>/usr
```

Parar xlisten-arm si se esta ejecutando, porque necesitamos el acceso al puerto:

```
$ ps –A | grep xlisten-arm
$ kill <process id for xlisten-arm>
$ cd /usr
```

Y finalmente programar el mote:

```
$ uisp -dprog=sggpio -dpart=ATmega128 --wr_fuse_h=d9 --wr_fuse_e=ff --erase --
upload if=main.srec.out
```

11.4 ANEXO 4: COMO INSTALAR TINYOS

11.4.1 COMO INSTALAR TinyOS-1.x

Sistema operativo de código abierto diseñado para redes inalámbricas de sensores empotrados. Ofrece una arquitectura basada en componentes, que permite innovación y una rápida puesta en marcha, mientas se minimiza el tamaño del código como necesario para los requerimientos de memoria en redes con sensores.

Paso 1

Descargar e instalar Sun's 1.4 JDK desde http://java.sun.com

Paso 2

Descargar el paquete instalación de cygwin http://webs.cs.berkeley.edu/tos/dist-1.1.0/tools/windows/tinyos-cygwin-1.1.zip descomprimir y ejecutar install.bat.

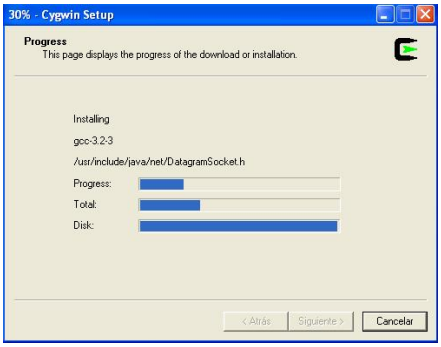

Figura 73 Instalación de TinyOS 1.x

Paso 3

Descargar Sun's javax.comm desde http://java.sun.com/products/javacomm/, descomprimir el archivo 'unzip javacomm20-win32.zip' y copiar los siguientes archivos dependiendo de donde se instalo JDK:

```
cp win32com.dll "c:\j2sdk1.4.2_16\jre\bin"
chmod 755 "c:\ j2sdk1.4.2_16\jre\bin\win32com.dll"
cp comm.jar "c:\ j2sdk1.4.2_16\jre\lib\ext"
cp javax.comm.properties "c:\ j2sdk1.4.2_16\jre\lib"
cp javax.comm.properties "c:\Program Files\jdk\lib"
```

Paso 4

Descargar e instalar graphviz desde http://webs.cs.berkeley.edu/tos/dist-1.1.0/tools/windows/graphviz-1.10.exe

Paso 5

Descargar e instalar los siguientes archivos .rpm. Comprobar si son las últimas versiones**.**

avarice-2.0.20030825cvs-1w.cygwin.i386.rpm
avr-binutils-2.13.2.1-1w.cygwin.i386.rpm
avr-gcc-3.3tinyos-1w.cygwin.i386.rpm
avr-insight-pre6.0cvs.tinyos-1w.cygwin.i386.rpm
avr-libc-20030512cvs-1w.cygwin.i386.rpm
nesc-1.1.2b-1.cygwin.i386.rpm
tinyos-1.1.15Dec2005cvs-1.cygwin.noarch.rpm
tinyos-contrib-1.1.4Apr2005cvs-1.cygwin.noarch.rpm
tinyos-micaz.16Jun2004-1.cygwin.noarch.rpm
tinyos-msp430.31May2004-2.cygwin.noarch.rpm
tinyos-tools-1.1.0-1.cygwin.i386.rpm
tinyos-vm-1.1.1Mar2004cvs-1.cygwin.noarch.rpm

Instalar mediante la orden 'rpm --ignoreos -ivh *.rpm'

11.4.2 COMO INSTALAR TinyOS-2.x

Versión 2 del sistema operativo de código abierto diseñado para redes inalámbricas de sensores empotrados.

Paso 1

Instalar Java 1.5 (Java 5) JDK. Java es la manera más común de interactuar con motes o gateways conectados a un PC.

Para Linux instalamos IBM JDK 1.5.

http://www-128.ibm.com/developerworks/java/jdk/

Descargar e instalar los archivos siguientes:

SDK

ibm-java2-i386-sdk-5.0-6.0.i386.rpm

Java Communication API

ibm-java2-i386-javacomm-5.0-6.0.i386.rpm

Instalar con la orden "rmp -ivh xxxxx.rpm"

Paso 2

Instalar compiladores nativos. En nuestro caso para los micaz tenemos que instalar las herramientas para Atmel AVR.

avr-binutils-2.15tinyos-3.i386.rpm
avr-gcc-3.4.3-1.i386.rpm
avr-libc-1.2.3-1.i386.rpm
avarice-2.4-1.i386.rpm
avr-insight-6.3-1.i386.rpm

Instalar con la orden "rmp -ivh xxxxx.rpm"

Paso 3

Instalar TinyOS Toolchain. Se instala las herramientas de compilación NesC y las herramientas de desarrollo para TinyOS 2.

nesc-1.2.8a-1.i386.rpm
tinyos-tools-1.2.4-3.i386.rpm

Instalar con la orden "rmp -Uvh xxxxxx.rpm"

En caso de tener instalado la versión TinyOs 1.x se debe ejecutar la orden de diferente forma. Instalar con la orden "rmp -ivh --force xxxxxx.rpm"

Paso 4

Instalar árbol fuente TinyOS 2.x.

Descargar e instalar el archivo "tinyos-2.0.2-2.noarch.rpm".

Instalar con la orden "rmp -ivh xxxxx.rpm"

Configuración del medio:

- Crear un script con las siguientes variables para mas tarde pasarlo a la carpeta /etc/profile.d.
- Comprobar que los permisos de los puertos series es -rw-rw-rw- por lo menos.

Paso 5

Instalar Graphviz. Descargar el archivo apropiado de la pagina web http://www.graphviz.org/Download_linux.php y proceder a instalar. En nuestro caso graphviz-2.17.20071121.0540-1.fc8.i386.rpm y se instalara con la orden "rmp -i xxxxxx.rpm".

Paso 6

Instalar librerías python para poder usar el simulador TOSSIM. . Descargar el archivo apropiado de la pagina web http://rpmfind.net/linux/RPM/opensuse/10.2/i586/python-devel-2.5-19.i586.html y proceder a instalar. En nuestro 1 - python-2.5-19.4.i586.rpm, 2 - blt-2.4z-243.i586.rpm, 3 - python-tk-2.5-19.i586.rpm y 4 - python-devel-2.5-19.4.i586.rpm; instalándose con la orden "rmp -Uvh xxxxxx.rpm".

Paso 7

Para actualizar el sistema desde cvs escribir: cvs -z3 -d:pserver:anonymous@tinyos.cvs.sourceforge.net:/cvsroot/tinyos co tinyos-2.x

En la línea de comandos y listo.

11.5 ANEXO 5: TRABAJAR CON IMOTE2 A TRAVÉS DE CABLE USB BAJO IP.

Para conectar mediante protocolo IP a través del USB tenemos que conectar el imote2 por medio del hyperterminal y la placa gateway IIB2400. Se le asigna la IP deseada en el archivo /etc/networks/interface. Ya podemos conectar el Imote2 a una maquina Linux, pero debemos levantar la IP del Linux ya que no se realiza automáticamente.
Con ifconfig se observan los dispositivos de red de que se dispone y aparece uno como USB. Este debe ser levantado:

```
ifconfig usb0 192.168.0.1 netmask 255.255.255.0 up
```

Una vez funcionando correctamente debe que dar como en la figura.

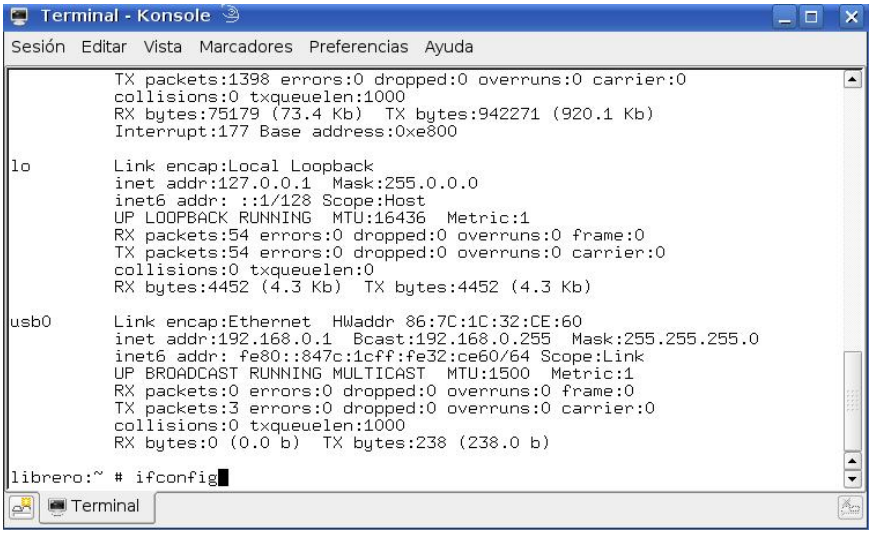

Figura 74 Configuración IP en la estación base donde conectar el Imote2

Y una vez levantada la IP ya podemos conectarnos al imote2 vía ssh o cualquier otro programa.

12 REFERENCIAS

[FORD2008] http://cse498t03s.cse.msu.edu/index.htm, CSE 498 Team 3, Ford Capstone Project.

[WEN2006] Towards Embedded Wireless-Networked Intelligent Daylighting Systems for Commercial Buildings; Yao-Jung Wen, Jessica Granderson, Alice M. Agogino.

[POL2002] Wireless Sensor Networks for Habitat Monitoring; Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler and John Anderson.

[MAR2004] Glacial Environment Monitoring using Sensor Networks; K. Martinez, P. Padhy, A. Riddoch, H.L.R. Ong, J.K. Hart.

[EVER2005] Wireless Sensor Networks and Beyond: A Case Study on Transport and Logistics; L. Evers, M. J. J. Bijl, M. Marin-Perianu, R. Marin-Perianu, P. J. M. Havinga.

[MOR2003] Ubiquitous Computing for Cognitive Decline: Findings from Intel's Proactive Health Research; Margaret Morris y Jay Lundell

[ECO2003] The Economist, In Dust We Trust, Technology Quarterly, Vol.371, No.8379, pp.10-12.

[CAM2008] Camalie Net Wireless Sensing; http://camalie.com/.

[PRA2005] An integrated multi-modal sensor network for video surveillance; Andrea Prati, Roberto Vezzani, Luca Benini, Elisabetta Farella y Piero Zappi

[THO2004] Auto-sensing and distribution of traffic information in vehicular ad hoc networks; Michael Thomas, Evtim Peytchev, David Al-Dabass; I.J. of Simulation Vol. 5 No 3-4.

[SHA2007] Multi-Purpose Wireless Accelerometers for Civil Infrastructure Monitoring; Shamim N. Pakzad, Sukun Kim, Gregory L Fenves, Steven D. Glaser, David E. Culler, James W. Demmel.

[SUK2007] Health Monitoring of Civil Infrastructures Using Wireless Sensor Networks; Sukun Kim, Shamim Pakzad, David Culler, James Demmel, Gregory Fenves, Steven Glaser y Martin Turon.

[XBOW2008] www.xbow.com

[HAT2005] Wireless Sensor Networks: Growing Markets, Accelerating Demand; Mareca Hatler and Charlie Chi; On world; Julio 2005.

[LEV2005] TinyOS: An Operating System for Sensor Networks; Philip Levis, Sam Madden, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer y David Culler.

[GRA2003] The nesC Language: A Holistic Approach to Networked Embedded Systems; David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, David Culler.

[BHA2005] MANTIS OS: An Embedded Multithreaded Operating System for Wireless Micro Sensor Platforms; Shah Bhatti, James Carlson, Hui Dai, Jing Deng, Jeff Rose, Anmol Sheth, Brian Shucker, Charles Gruenwald, Adam Torgerson y Richard Han; ACMKluwer Mobile Networks & Applications (MONET) Journal, Special Issue on Wireless Sensor Networks; Agosto 2005.

[BTN2008] The BTnut Operating System, BTnut System Software Project; http://www.btnode.ethz.ch/static_docs/doxygen/btnut/main.html

[ESW2005] Nano-RK: an Energy-aware Resource-centric RTOS for Sensor Networks; Anand Eswaran, Anthony Rowe y Raj Rajkumar.

[XBOW2008-2] MPR-MIB Users Manual; http://www.xbow.com

[XBOW2008-3] MTS/MDA Sensor Board Users Manual; http://www.xbow.com

[HYU2008] Instruction-Level Power Estimator for Sensor Networks; Hyunwoo Joe, Jaebok Park, Chaedeok Lim, Duk-Kyun Woo, y Hyungshin Kim; ETRI Journal, vol.30, No.1, Feb. 2008, pp.47-58.

[HEL2008-1] Heliomote User manual; http://www.atlalabs.com/

[HEL2008-2] Technical Support, http://www.atlalabs.com/

[XBOW2008-4] Stargate Developer's Guide Rev. B; http://www.xbow.com

[RAG2005] Embedded Linux system design and development; P. Raghavan, Amol Lad, Sriram Neelakandan; Capiulo 5; Pag 127-164.

[MAX2008] Maxim Max1363 data sheet; http://www.maxim-ic.com/

[JEW2008] http://www.jewellinstruments.com/

[XBOW2008-5] MPR-MIB Users Manual, http://www.xbow.com

[XBOW2008-6] Stargate Users Manual, http://www.xbow.com

Material didáctico

Ingenierías, nº 26

UNIVERSIDAD
DE LA RIOJA