INFORME TÉCNICO

Sistemas embebidos y en red para el control de sistemas físicos. Estudio de redes inalámbricas de sensores.

Bruno Caballero Retamosa

Rodolfo Haber Guerra

Febrero de 2008

Índice general

1

Resumen

En este informe se presenta un estudio sobre los sistemas embebidos y en red para sistemas físicos, a través de la revisión de la literatura disponible, específicamente enfocado en las redes de sensores inalámbricas. Se trata de dispositivos (hardware) que interactúan con el entorno y que se comunican a través de una tecnología de red inalámbrica, y cuyo objetivo es la toma de decisión y realización de una acción inteligente coordinada. Del estudio se concluye que en el campo de las plataformas hardware hay se requiere continuar los esfuerzos hacia la miniaturización. Cuanto más pequeño sea el dispositivo, más posibilidades tendrá de ser integrado en diversos entornos. El éxito de las redes de sensores va a depender en gran medida de esta capacidad. Los desarrollos software analizados intentan ofrecer un entorno que facilite el desarrollo de aplicaciones en las redes de sensores. Se ha revisado el estándar ZigBee/IEEE 802.15.4 y la literatura reciente en este tema, resumiendo las soluciones adoptadas en el estándar y las nuevas propuestas surgidas en los últimos años. Los problemas energéticos todavía no han sido solucionados, hace falta una mayor autonomía energética de los nodos. Además, las redes inalámbricas se consideran aún una opción arriesgada para el control de procesos, debido a la que su fiabilidad no ha sido completamente demostrada.

Introducción

El objetivo de este estudio es analizar y estudiar los sistemas embebidos y en red para sistemas físicos reales, a través de la revisión de la literatura disponible. Se trata de dispositivos (hardware) que interactúan con el entorno y que se comunican a través de una tecnología de red inalámbrica, y cuyo objetivo es la toma de decisión y realización de una acción inteligente coordinada.

El diseño y desarrollo de sistemas embebidos y en red para el control abarca varias tareas diferenciadas [31], tales como la elección de arquitecturas hardware adecuadas, el desarrollo de software (sistemas operativos, middleware, etc.) y de protocolos de comunicación que cumplan requisitos específicos para dichos sistemas. Así mismo, son necesarias metodologías de diseño y verificación. Es- tas metodologías deben de garantizar un comportamiento eficiente, robusto y seguro, así como escalabilidad y adaptabilidad del sistema en red. Ocupan un papel relevante el diseño del sistema encargado de la toma de decisión inteligente y su implementación eficiente [26, 28].

Los avances recientes en microprocesadores, tecnologías de comunicación inalámbricas y sistemas microelectromecánicos (MEMS), han propiciado la aparición de dispositivos de bajo coste y bajo consumo que pueden observar y

reaccionar ante fenómenos físicos en el entorno. En los últimos años, una de las áreas de investigación más activa es la referente a las redes de sensores (Wireless Sensor Networks) [33]. Por red de sensores entendemos una red de computadores embebidos que interactúa con el entorno. Estos computadores embebidos, o nodos sensores, son físicamente pequeños, de relativo bajo coste y están provistos de algún conjunto de sensores o actuadores. Son dispersados en el entorno cerca de los objetos que están midiendo y se comunican por red, permitiendo la cooperación para una tarea común, por ejemplo, monitorizar el entorno y efectuar cambios en él. Son estas tres características (dispositivos embebidos, ser capaz de medir y actuar, y la capacidad de comunicarse entre ellos), las que definen una red de sensores.

Algunos artículos [3, 33] discuten varios aspectos de las redes inalámbricas de sensores. En este trabajo se realiza una revisión de las recientes aportaciones y una discusión de los retos actuales. Algunos temas, como la seguridad y la localización de nodos no han sido tomados en cuenta en este estudio, debido a que se ha preferido centrarse en los aspectos que guardan relación con el resto de las asignaturas cursadas en el master.

El resto del trabajo se organiza de la siguiente manera. En el capítulo 2 se

revisa el hardware utilizado para redes de sensores. En el capítulo 3 trata los desarrollos software para estos sistemas, tales como sistema operativo y middleware. En el capítulo 4 se discutirán las tecnologías de redes inalámbricas y protocolos utilizados. El capítulo 5 trata el importante tema de la eficiencia energética. En el capítulo 6 se revisarán las diferentes propuestas de sistemas inteligentes implementados en dichos sistemas. Por último, se presentarán las conclusiones y el trabajo futuro en el capítulo 7.

2

Plataformas Hardware

El amplio éxito de las redes de sensores, unido del bajo coste y alta disponibilidad del hardware, ha provocado la aparición de una amplia variedad de dispositivos, tanto en ámbitos académicos como en comerciales.

Los dispositivos de estas redes se caracterizan por su pequeño tamaño, la capacidad de medir fenómenos ambientales a través de ciertos sensores, un transmisor de radio para comunicaciones y batería autónoma. Cada nodo contiene normalmente una CPU de propósito general. La capacidad de almacenamiento es muy diversa en las distintas soluciones. La mayoría de ellas posee interfaces de entrada salida para conectar nuevos sensores. A continuación, se analizarán algunas de las soluciones, las más prometedoras entre las disponibles y se enumerarán algunas de sus ventajas e inconvenientes.

2.1. Plataformas hardware

Los dispositivos que pueden componer una red de sensores se puede dividir en dos grandes categorías: dispositivos pequeños, con microcontroladores de arquitectura de 8 bits, 10-100 KB de memoria y posiblemente 100-1000 KB de memoria flash para almacenamiento secundario; y dispositivos más potentes: con CPU de 32 bits y memoria del orden de los MB.

La serie Mote [54] es una de la más representativa del primer grupo. Estos dispositivos suelen contener una pequeña PCB con un microcontrolador de 8 bit muy sencillo (por ejemplo el Atmel Atmega128), memoria del orden de KB, conversores analógico digital y varias interfaces de comunicación estándar tales como UART (Universal Asynchronous Receiver / Transmitter) o SPI (Serial Peripheral Interface). Esta familia ha tenido un gran éxito en los últimos años. Originalmente fueron desarrollados en la Universidad de Berkeley, pero ahora están disponibles comercialmente a través de diversas compañías tales como Crossbow [13] y Dust Networks [18]. Otras instituciones tanto académicas como comerciales han desarrollado dispositivos similares: por ejemplo, Nymph de la Universidad de Colorado [1], BTnodes de ETH Zurich [9] o tmote de Mote iv [49] recientemente adquirido por Sentilla [56]. En la figura 2.1 se puede observar algunos de estos dispositivos.

Entre los dispositivos con mayor potencia podemos encontrar productos como Imote2 (diseñado por Intel, disponible a través de Crossbow [13], o Cerfcube

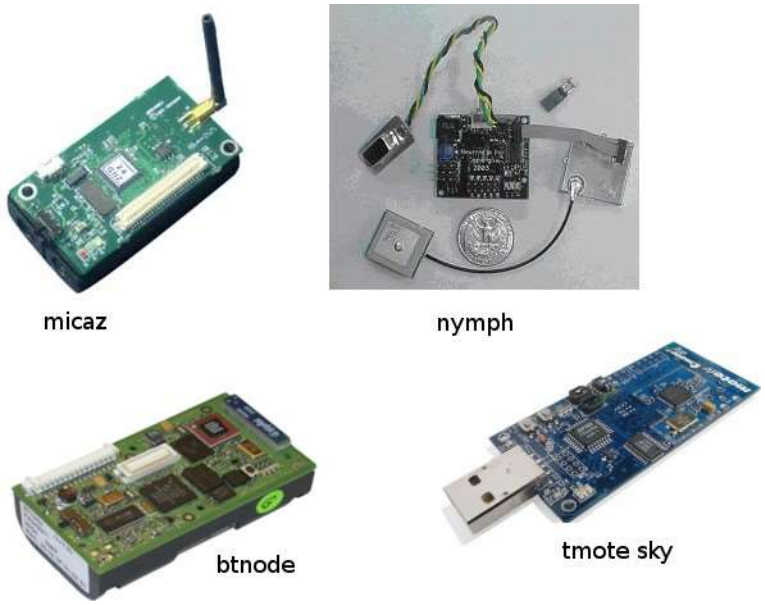

Figura 2.1: Ejemplos de plataformas hardware disponibles para redes de sensores

[36]. Estos dispositivos son utilizados en una amplia variedad de aplicaciones. En el contexto de las redes de sensores se utilizan como gateways para un grupo de nodos, o para aplicaciones que requieran mayor potencia de cálculo. Suelen utilizar procesadores tales como X-Scale o ARM. Este segundo tipo de dispositivos, al disponer de mayores recursos computacionales, tienen un consumo mayor de energía.

En este trabajo sólo se tendrá en cuenta el primer grupo de dispositivos, debido a que son los que mejor se adaptan a las necesidades de las redes de sensores tal cómo las hemos de nido anteriormente, es decir, tienen menos recursos, pero también mucho menos consumo energético. No obstante, es interesante conocer que hay dispositivos más potentes para determinadas aplicaciones que realmente lo necesiten.

2.2. Diseño de dispositivos

Un aspecto muy importante en el desarrollo de las redes de sensores es que el tamaño de los dispositivos tiene que ser lo más pequeño posible, con el objetivo de que pueda ser integrado en diversos escenarios de la manera más sencilla. Así mismo, debido a la naturaleza de estas redes, el consumo de energía también debe ser minimizado, prolongando al máximo la vida útil del dispositivo. La industria debe hacer un esfuerzo en conseguir estos objetivos. Uno de los dispositivos más pequeños encontrados en la literatura es SAND [51] desarrollado por IMEC en el HUMAN++ project (ver figura 2.2).

Hay principalmente dos enfoques en el diseño de los dispositivos para redes de sensores. El más general y expandible consiste en desarrollar una placa principal con el microcontrolador y un conjunto de placas que puedan ser unidas

Figura 2.2: Módulos de la plataforma SAND

(apiladas una encima de otras) a la primera. Un ejemplo de este tipo de dispositivos son los desarrollados por Crossbow anteriormente mencionados. El segundo enfoque consiste en disponer los transductores directamente en la pla- ca del microcontrolador, soldados, en un paquete único. Se trata de desarrollar un dispositivo específico para la aplicación que se necesite implementar.

El primer enfoque proporciona mayor posibilidad de reconfiguración y es más útil para desarrollo y pruebas de distintas aplicaciones. El segundo enfoque, por otro lado, reduce el coste y el tamaño del dispositivo, debido a que se desarrolla con unas especificaciones concretas para una aplicación determinada.

2.3. Sensores

La tecnología de sensores también tiende a la miniaturización. A pesar de la diversidad de los mismos, su principio de operación es más o menos el mismo. Se basan en que los factores ambientales induzcan cambios en las propiedades eléctricas de un material apropiado. El sensor incorpora un circuito que detecta estos cambios eléctricos en el material y, correctamente calibrado, es capaz de medir fenómenos ambientales. El tipo de material dependerá de la funcionalidad requerida. Por ejemplo, los sensores de temperatura se basan en el cambio de resistividad de algunos materiales con respecto a este fenómeno, estos pueden ser desde metales a semiconductores, dependiendo de la sensibilidad requerida. Los sensores de luz usan materiales foto-conductivos, cuyas características eléctricas dependen de la cantidad de luz que incida sobre ellos. Por último, los acelerómetros miden el voltaje inducido por deformaciones estructurales en materiales piezoeléctricos [27].

2.4. Comparación de los nodos disponibles

La elección de una plataforma va a depender de los requisitos propios de la aplicación a la que va a ser destinada. Por ello, es difícil hacer una buena comparación. Entre las diferencias más notables encontramos que hay algunos dispositivos que se comercializan con el circuito electrónico sin empaquetar y aislar. Para propósitos académicos es perfectamente válido, pero para aplicaciones en el medio ambiente es necesario algún tipo de protección. La potencia de cálculo es también un elemento diferenciador, en la mayoría de los casos las

aplicaciones simples, por ejemplo, aquellas que se encargan de sólo la recolección de datos, requerirán un microcontrolador de 8 bits, pero si nuestra aplicación necesita que los dispositivos tengan mayor potencia de cálculo, inevitablemente serán de mayor tamaño y mayor consumo energético. El tipo de sensores a utilizar también va a limitar la selección de una plataforma, debido a que se deberá comprobar que el dispositivo tenga la interfaz adecuada para los sensores que deseemos utilizar.

En la tabla 2.1 se expone a modo de ejemplo algunas características de algunos dispositivos más utilizados [3].

	Btnode 3	mica2	mica2dot	micaz	telos A	tmote sky	EYES
Fabricante	Art of Technology	Crossbow	Crossbow	Crossbow	Mote iv	Mote iv	Univ. of Twente
Microcontrolador	Atmel Atmega 128L	Atmel Atmega 128L	Atmel Atmega 128L	Atmel Atmega 128L	Texas Instruments MSP430	Texas Instruments MSP430	Texas Instruments MSP430
Frecuencia de reloj	7.37 MHz	7.37 MHz	7.37 MHz	7.37 MHz	8 MHz	7.37 MHz	5 MHz
RAM (KB)	64+180	4	4	4	2	10	2
ROM (KB)	128	128	128	128	60	48	60
Almacenamiento (KB)	4	512	512	512	256	1024	4
Tecnología de Radio	Chipcon CC1000 315 / 433 / 868 / 916 MHz 38.4 Kbps	Chipcon CC1000 315 / 433 / 868 / 916 MHz 38.4 Kbps	Chipcon CC1000 315 / 433 / 868 / 916 MHz 38.4 Kbps	Chipcon CC2420 2.4 GHz 250Kbps IEEE 802.15.4	Chipcon CC2420 2.4 GHz 250Kbps IEEE 802.15.4	Chipcon CC2420 2.4 GHz 250 Kbps IEEE 802.15.4	RFM TR1000 868 MHz 57.6 Kbps
Rango máximo	150-300 m	150-300 m	150-300 m	75-100 m	75-100 m	75-100 m	75-100 m
Alimentación energética	2 Baterias AA	2 Baterias AA	Coin cell	2 Baterias AA	2 Baterias AA	2 Baterias AA	2 Baterias AA
Conector de PC	Placa de programación conectada al PC	Placa de programación conectada al PC	Placa de programación conectada al PC	Placa de programación conectada al PC	USB	USB	Serial Port
Sistema Operativo	TinyOS	TinyOS	TinyOS	TinyOS	TinyOS	TinyOS	PEEROS
Transductores	Placa de adquisición	Placa de adquisición	Placa de adquisición	Placa de adquisición	En placa	En placa	Placa de adquisición

Cuadro 2.1: Características de los dispositivos más utilizados

3

Software en Redes de Sensores

La mayoría de las plataformas para redes de sensores se han desarrollado orientadas a la investigación. Por ello casi todo el desarrollo software que hay disponibles para ellas se basan en una mezcla de componentes of-the-shelf COTS y componentes a medida. Esto provoca que el desarrollo de aplicaciones prácticas sea una tarea más complicada. No hay un concepto generalmente aceptado de driver de dispositivo para Redes de Sensores, y el desarrollo de un sistema operativo o entorno de ejecución están aún en sus primeras etapas de desarrollo y es propenso a errores [62].

Hay algunos intentos para conseguir una abstracción de hardware flexible para diversos dispositivos de redes de sensores, se trata de desarrollar modelos de programación para drivers de dispositivos y lenguajes de programación específicos para este tipo de arquitecturas. De todas maneras, se necesita aclarar cómo los diferentes sistemas operativos soportan los diversos modelos de programación y cuales de ellos son eficientes para los requerimientos específicos de las redes de sensores.

3.1. Sistema operativo (abstracción de hardware)

En las redes de sensores los nodos deben comunicarse y coordinarse entre ellos para conseguir los resultados esperados por la aplicación. Implementar esa coordinación de una manera eficiente, minimizando el ancho de banda necesario y el consumo de energía, y a la vez maximizando el rendimiento, es una tarea muy compleja. Un paso fundamental es decidir que coordinación y modelo de comunicación adoptar para conseguir un rendimiento eficiente. Por tanto, se hace necesario un modelo de programación no-local y distribuido.

Los nodos ofrecen sus servicios a las aplicaciones distribuidas en la red como conjunto. Para hacer esto, es necesario que trabajen distintas capas y componentes software. El modelo de programación de dichas capas no es necesariamente el mismo en la red o como en el nivel de aplicación.

En cada nodo, podemos identificar cuatro capas software diferentes:

1. Abstracción de hardware y drivers de dispositivo: Esta capa es muy es-

pecífica en cada dispositivo y difícil de generalizar.

2. Sistema operativo y servicios básicos: Algunos servicios básicos de los sistemas operativos tales como administración de memoria, control de entrada salida estructurada, manejo de procesos e hilos. Estos módulos software requieren esfuerzo en estandarización, encapsulación, mantenibilidad, configurabilidad, etc.

3. Servicios de aplicación específica: Se construyen sobre servicios básicos y ofrecen una vista más abstracta del nodo.

4. Aplicación: Implementa la funcionalidad particular requerida para el nodo.

La duda es: puede encontrarse un modelo de programación que se ajuste a las necesidades de las redes de sensores en general, o necesitamos utilizar diferentes modelos de programación en diferentes aplicaciones. A continuación se introducen distintas propuestas.

3.1.1. Solución simple: arquitectura orientada a eventos

Uno de los modelos mas populares encontrados en las redes de sensores es el modelo orientado a eventos. Algunos ejemplos son Contiki [17], TinyOS [34], BTnodes [5]. TinyOs es el sistema operativo más utilizado en las redes de sensores. Está escrito utilizando el lenguaje NesC [23], una adaptación del lenguaje de programación C diseñado para embeber los conceptos estructurales de TinyOs, TinyOs adopta el modelo basado en componentes para construir aplicaciones de redes de sensores en un entorno orientado a eventos. Los componentes se enlaZAN estáticamente para construir una imagen binaria que es programada en la memoria flash del sensor.

Estos sistemas operativos están diseñados para microcontroladores con recursos computacionales bastante limitados. El problema de estos sistemas es que no encajan perfectamente con una semántica de ejecutar hasta completar en las tareas individuales. Se necesita, por tanto, dividir la aplicación en pequeños subtareas. La experiencia con este tipo de modelos muestra que es más complejo de mantener una aplicación, porque los cambios en tareas individuales pueden condicionar el rendimiento de otras, e incluso una tarea que tarde mucho tiempo en ejecutarse puede impedir que otras los hagan.

3.1.2. Solución clásica: arquitectura multi-hilo

Otros investigadores proponen utilizar el modelo de programación clásico de los ordenadores, un sistema operativo preemptible y multi-hilo. El más conocido para redes de sensores es MANTIS [6]. En este modelo, las tareas pueden ser interrumpidas por tareas de mayor prioridad. Por ejemplo, una tarea es parada para recibir un paquete de red, luego es retomada la primera después de la recepción del paquete. Es el modelo más utilizado y en el que la mayoría de los programadores están más acostumbrados a trabajar.

3.2. Middleware

Como hemos visto anteriormente, las características propias de las redes provocan que el desarrollo de aplicaciones más o menos complejas sea una actividad en principio no trivial. La propuesta de usar una capa middleware es una alternativa bastante útil.

Por middleware entendemos software de conectividad, que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red). El Middleware nos abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, proporcionando una API para la fácil programación y manejo de aplicaciones distribuidas.

El desarrollo de middleware debe asumir los siguientes retos debido a las características de las redes de sensores:

- Administrar recursos limitados y bajo consumo.
- Escalabilidad, movilidad y topología de red dinámica.
- Permitir la heterogeneidad en los nodos que componen la red.
- Permitir la re-organización dinámica de la red.
- Técnicas relacionadas con calidad de servicio.
- Seguridad.

Una solución de middleware completa podría contener un entorno de ejecución que soporte y coordine múltiples aplicaciones, que estandarizase los servicios del sistema y tuviera los mecanismos para hacer un uso eficiente de los recursos para prolongar la vida útil del sensor en la red [32].

A continuación, comentaremos algunas de las propuestas de middleware para redes de sensores.

3.2.1. Middleware basado en máquina virtual

Por máquina virtual entendemos software que crea un entorno virtual entre la plataforma y una aplicación, permitiendo que este ejecute un software más sencillo en distintas plataformas. Este sistema es muy flexible. Entre las venta- jas de middleware basado en máquina virtual podemos destacar que permite a los desarrolladores escribir aplicaciones en módulos separados y que el sistema puede distribuir estos módulos a través de la red. Entre las desventajas está la sobrecarga computacional introducida por el uso de un intérprete o máquina virtual.

Mate [41, 42] es un middleware con este tipo de enfoque. El proyecto se centra en la necesidad de nuevos paradigmas de la programación para superar las restricciones de ancho de banda limitado y bajo consumo energético. En el caso de las redes de sensores, enviar un bit consume la misma energía que ejecutar miles de instrucciones que producen ese bit de datos. Mate es un intérprete de código**byte** que se ejecuta sobre TinyOS. La comunicación entre los nodos se basa en el paradigma de mensajes activos. Cada mensaje contiene el ID y un

manejador para ser invocado en el nodo destino. Mate utiliza código troceado en cápsulas de 24 instrucciones, cada una de ella de 1 byte. Esto bene cia a los programas grandes que se componen de múltiples cápsulas, ya que de esta manera puede ser fácilmente esparcido por la red. A cada cápsula se le añade un número de versión, así la comparación tiene lugar entre vecinos, permitiendo un proceso de actualización en cascada.

Magnet [4] es otro ejemplo de este tipo de middleware. MagnetOS es un sistema operativo adaptativo, de bajo consumo energético especialmente diseñado para redes de sensores. Constituye una capa definida como Imagen única del sistema que provee un nivel de abstracción alto para la naturaleza heterogénea y distribuida de estas redes. La abstracción hace que el sistema aparezca como una simple y unificada máquina virtual Java. Divide automáticamente las aplicaciones en componentes y migra de manera transparente estos componentes a los nodos adecuados de la red. Una sola imagen del sistema provee la abstracción necesario para aplicaciones sean adaptativas, eficientes enérgicamente e independientes de la plataforma. Este middleware requiere nodos con mayor potencia computacional que la anterior propuesta.

3.2.2. Middleware basado en programación modular

La clave en este enfoque es que las aplicaciones sean lo más modulares posibles, de tal manera que faciliten su inserción y distribución en la red de sensores. Transmitir pequeños módulos de programa consume menos energía que transmitir la aplicación entera. Un ejemplo es Impala [44], provee un mecanismo para actualizar la red de sensores para sostener aplicaciones dinámicas. Fue desarrollado originalmente para una plataforma específica (HP/Compaq iPAQ Pocket PC). Sus problemas son que no permite la heterogeneidad del hardware y no es aceptable para dispositivos con recursos limitados.

3.2.3. Middleware orientado a base de datos

Este enfoque considera la red entera como un sistema de base de datos virtual. Proporciona un interfaz fácil de usar y permite al usuario realizar peticiones a la red de sensores y extraer los datos de interés. Sin embargo, este modelo no soporta aplicaciones de tiempo real que necesiten detectar relaciones espacio-temporales entre los eventos.

Cougar [7] introduce un nuevo concepto en el middleware al adoptar este enfoque de base de datos, donde los datos de los sensores son considerados como datos de una base de datos relacional virtual. Permite implementar la administración de la red de sensores mediante peticiones al estilo SQL.

TinyDB [46] es un sistema de procesamiento de peticiones SQL para extraer información de la base de datos usando el sistema operativo TinyOS. TinyDB elimina esta complejidad aportando una interfaz el estilo SQL para extraer los datos de interés. Sin ella, otras soluciones basadas en TinyOS requerirían utilizar el lenguaje NesC para extraer información de los sensores. Las peticiones usan manipulación sencilla de datos indicando el tipo de sensor y el conjunto de nodos de interés. Para ello TinyDB mantiene una base de datos donde cada la es un sensor, y las columnas contienen información tales como el tipo de sensor, el

identi cador o la cantidad de batería que queda. Un ejemplo de una petición al estilo SQL en TinyDB sería:

```
SELECT nodeid, light, temp
FROM sensors
SAMPLE PERIOD 1s FOR 10s
```

3.2.4. Middleware orientado por la aplicación

Los parámetros de red in uyen notablemente tanto en el rendimiento de la aplicación distribuida, así como en el consumo de energía. Se necesita poder a nar los parámetros de red partiendo de los requerimientos de la aplicación, incluir característica de calidad de servicio. Se trata de que el middleware pueda actuar sobre la pila de protocolo de red.

Milan (Middleware Linking Applications and Networks) [50] permite controlar continuamente la funcionalidad de la red en función de los requerimientos cambiantes de la aplicación. Se trata de que el programador se centre en preocupaciones de alto nivel, dejando al sistema con los detalles de bajo nivel. Milan permite a las aplicaciones de la red de sensores especificar sus necesidades y ajusta las características de las redes para incrementar el tiempo de vida de la aplicación a la vez que mantiene las necesidades. Para conseguirlo necesita la calidad de servicio individual necesaria para las aplicaciones, identificar la importancia relativa de cada una de ellas y los recursos disponibles en la red (tanto ancho de banda de comunicación como energía).

3.2.5. Middleware orientado a mensaje.

Middleware orientado a mensajes usa el mecanismo productor/consumidor para facilitar el intercambio de mensajes entre nodos. La fuerza de este paradigma radica en que soporta comunicación asíncrona, permitiendo un bajo acoplamiento entre emisor y receptor. Este esquema es muy apropiado para las redes de sensores, donde la mayoría de las aplicaciones son orientadas a eventos.

La comunicación en este modelo es más sencilla a gran escala. Sólo se consideran los nodos como productores y consumidores, y hay un servicio de eventos. Los suscriptores registran su interés en recibir notificaciones de ciertos eventos. Los productores pueden crear y enviar eventos. Este modelo ofrece varios tipos de desacoplamiento entre el nodo que envía (productor) y el que recibe (consumidor) [12, 19]:

1. Desacoplamiento en el espacio: productor y consumidor no necesitan saber dónde se encuentra el otro nodo.

2. Desacoplamiento en el tiempo: Ambos nodos no necesitan estar conectados al mismo tiempo, un servicio de eventos se encarga de retransmitir la información.

3. Desacoplamiento de sincronización: los productores no necesitan estar bloqueados mientras los receptores procesan la información.

El fuerte desacople entre compañeros de comunicación hace a este modelo particularmente apropiado para las Redes de Sensores.

Mires [59] propone una adaptación del middleware orientado a mensaje para sistemas distribuidos tradicionales y jos. Mires proporciona un modelo de comunicación asíncrona que es apropiado para aplicaciones de Redes de Sensores, que son dirigidas por eventos en la mayoría de los casos. Está desarrollado sobre tinyOS usando NesC, adoptando el modelo de programación basado en componentes y usando mensajes activos para implementar el modelo productor-consumidor.

Este desacoplamiento es aún mayor en Content-based Publish/Subscribe, en donde las suscripciones son expresadas en el contenido del mensaje [14, 11]. En este caso, todo se decide en tiempo de envío, por ejemplo, la conjunto de nodos intermediarios en la comunicación, proporcionando una mayor flexibilidad.

3.2.6. Comportamiento global

Se propone un nuevo concepto sobre cómo programar redes de sensores, se trata de definir el comportamiento global de la red. Esta macro-programación implica programar la red de sensores como un todo, más que programar software de bajo nivel para cada nodo individual. El resultado es que se obtiene un programa con una especificación de alto nivel, permitiendo la generación automática de comportamiento de cada nodo.

Kairos [25] permite expresar en un solo programa dicho comportamiento global y genera subprogramas que puedan ejecutar los nodos locales. Incluye un compilador que se encarga de los detalles de bajo nivel y de generar el código distribuido, acceso remoto y administración.

4

Red

La red es la parte esencial de las redes de sensores. Es el elemento que permite que los nodos colaboren entre sí para llevar a cabo una acción conjunta. Además, es el elemento hardware que consume mayor cantidad de energía dentro de los nodos, a menudo entre el 20 % y el 40 % del total. Por ello optimizando los protocolos de comunicación se puede extender la vida útil de los sensores de nuestra red.

Era necesario la aparición de nuevas tecnologías de red inalámbrica para los requisitos de las redes de sensores, es decir, baja tasa de envío de datos y maximización de la vida útil de sus baterías. Se requería, además, capacidades autoorganizativas y bajo coste. ZigBee surge como una especificación basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal. Propone cambios con respecto a otras tecnologías, por ejemplo, comparado con Bluetooth, ZigBee tiene un menor consumo eléctrico, una velocidad menor y permite un mayor número de nodos en la red.

4.1. ZigBee y el estándar 802.15.4

La alianza ZigBee [69] es una asociación de compañías que trabajan juntas para conseguir productos de monitorización y control ables, de bajo coste, de bajo consumo energético, conectados mediante redes inalámbricas, basándose en un estándar abierto y global. Probablemente la tecnología ZigBee se introduzca en muchos productos y aplicaciones en los próximos años. ZigBee se basa en el estándar IEEE 802.15.4, en el cual de ne la capa física y MAC para redes inalámbricas de área personal con baja tasa de transferencia. ZigBee define la capa de red y la capa de aplicación.

4.1.1. Estándar IEEE 802.15.4

El estándar 802.15.4 de ne las características de las capas física y de acceso al medio para Low-Rate Wireless Personal Area Networks (LR-WPAN). Las ventajas de este tipo de red son facilidad de instalación, fiabilidad, bajo coste, bajo consumo y manteniendo una pila de protocolos flexible.

La capa física funciona en tres bandas de frecuencia: 2450 MHz (con 16 canales), 915 MHz (con 10 canales) y 868 MHz (1 canal), todas ellas usando

	2450 MHz	915 MHz	868 Mhz
Tasa de transferencia	250 kbps	40 kbps	20 kbps
Número de canales	16	10	1
Modulación	O-QPSK	BPSK	BPSK
Periodo de símbolos	16 us	24 us	49 us

Cuadro 4.1: Especificaciones de la capa física IEEE 802.15.4

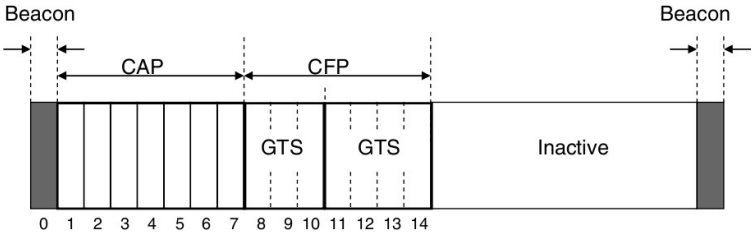

Figura 4.1: Supertrama MAC de IEEE 802.15.4

el modo de acceso Direct Sequence Spread Spectrum (DSSS). La tabla 4.1 [3] resume las principales características de las tres bandas. La capa física dispone también de funcionalidad para la selección de un canal, estimación de la calidad del enlace, entre otros servicios.

La capa de acceso al medio define dos tipos de nodos:

- Dispositivos con funcionalidad completa (Full Function Devices - FFD). Están equipados con el conjunto completo de funcionalidad para la capa de acceso al medio (MAC), lo que le permite actuar como coordinador de red o como dispositivo final. Cuando actúan como coordinador, estos dispositivos mandan tramas beacons que permite la sincronización, comunicación y servicios de unirse a la red.
- Dispositivos de funcionalidad reducida (Reduced Function Devices - RFD). Sólo pueden actuar como dispositivos finales de red, interactuando con un dispositivo coordinador.

El estándar 802.15.4 considera sólo dos tipos de topología: la topología en estrella y la topología peer-to-peer. En la primera se adopta el modelo maestro-esclavo. Un dispositivo FFD toma el rol de coordinador de la red de área personal (Personal Area Network, PAN). Los otros dispositivos pueden ser RFD o FFD y sólo se comunicarán con el coordinador. En la topología peer-to-peer los dispositivos FFD pueden comunicarse con otros FFD en su rango de comunicación, y enviar mensajes a otros nodos FFD a través de nodos intermedios FFD.

Un coordinador de la red de área personal (PAN) puede operar con una supertrama o sin ella. En el primer caso la comunicación empieza con la supertrama (figura 4.1) con una señal beacon al inicio que sirve para sincronizar, así como para describir la estructura de la supertrama y mandar información de control a la PAN. La supertrama se divide en dos secciones una activa y otra

inactiva donde el coordinador puede dormir y ahorrar energía. En la sección activa está divida en un ranuras de tamaño fijo y contiene una parte llamada Contention Access Period (CAP), donde los nodos compiten por el canal usando el protocolo CSMA-CA y otra Contention Free Period (CFP), donde los nodos pueden transmitir sin competir por el canal en ranuras Guaranteed Time Slots (GTS) asignadas y administradas por el coordinador de la PAN. Cuando un nodo quiere enviar información al coordinador, debe esperar una trama beacon para sincronizarse y después competir por el acceso al canal. Sin embargo, la comunicación desde el coordinador hacia un nodo final es indirecta. El coordinador almacena el mensaje y anuncia el envío pendiente en una trama beacon. Los nodos finales normalmente están inactivos largo tiempo, y sólo se activan periódicamente para ver si tienen que recibir algún mensaje. Cuando un nodo
nal advierte que hay un mensaje para él, lo pide explícitamente durante la sección CAP. Si un coordinador quiere comunicarse con otro coordinador, debe sincronizarse con su trama beacon y actuar como un nodo nal.

Hay una segunda opción, y es funcionar sin supertrama. El coordinador de la PAN nunca manda tramas beacons y la comunicación se realiza a través del protocolo CSMA-CA. El coordinador está siempre activo esperando a recibir tramas. La comunicación inversa se realiza mediante polling, los nodos finales se activan periódicamente y preguntan al coordinador por mensajes pendientes [3].

El protocolo CSMA-CA significa Carrier Sense, Multiple Access, Collision Avoidance, acceso múltiple por detección de portadora con evasión de colisiones. Cada equipo anuncia opcionalmente su intención de transmitir antes de hacerlo para evitar colisiones entre los paquetes de datos. Aunque el protocolo asegure que un nodo va a obtener un acceso al medio, no se asegura que el nodo destino esté en contacto con el nodo origen. Para solucionar este problema se ha añadido un procedimiento de saludo adicional al protocolo de la capa MAC. Para enviar una trama, el equipo origen primero envía una trama corta de control de solicitud de transmisión RTS (Request To Send). Este mensaje de control RTS contiene las direcciones de MAC del equipo origen y destino. Si el equipo destino recibe esta trama, devolverá una trama de contestación: preparado para transmitir CTS (Clear To Send) o receptor ocupado (RxBUSY). Si la respuesta es afirmativa el equipo origen transmite la trama (DATA). Si el equipo destino recibe correctamente el mensaje contesta con la trama de confirmación positiva ACK (ACKnowledged) y si no la recibe correctamente contesta con la trama de confirmación negativa NACK (Not ACKnowledged) y el equipo origen tratará de volver a enviarlo. Este procedimiento se repite un número predefinido de veces hasta conseguirse una transmisión correcta de la trama DATA.

Además de transferencia de datos, la capa de acceso al medio tiene otras funcionalidades tales como escanear los distintos canales disponibles y asociarse o desasociarse de un canal.

4.1.2. Estándar ZigBee

El estándar ZigBee trata las capas red y aplicación de la pila de protocolos. La capa de red tiene como función hacer que los datos lleguen desde el origen al destino, aún cuando ambos no estén conectados directamente. La capa de aplicación provee un framework para el desarrollo de aplicaciones distribuidas. ZigBee está construido sobre el estándar 802.15.4. Ver figura 4.2.

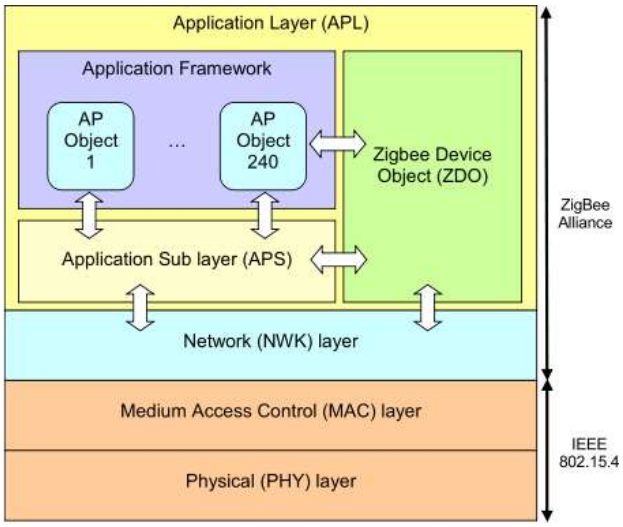

Figura 4.2: Pila de protocolos de ZigBee y 802.15.4

Capa de red

ZigBee identifica tres tipos de dispositivos. El primero, un dispositivo final ZigBee con RFD o FFD actuando como dispositivo final simple. En segundo lugar, un router ZigBee, es decir, un dispositivo FFD con capacidades de enrutar paquetes. Por último, un coordinador (solamente uno por cada red) que administra toda la red. ZigBee soporta tres tipologías de red, aparte de la topología estrella de 802.15.4, existen la topología en árbol y la topología en malla (Figura 4.3). La capa de red proporciona enrutado de paquetes entre los distintos dispositivos, descubrimiento y mantenimiento de rutas, seguridad, posibilidad de unirse o dejar una red y la asignación de una dirección única de 16-bit.

Una red se forma por medio de un procedimiento de unión de cada nodo a la red. Cuando un dispositivo desea unirse a una red existente, la capa de red inicia el procedimiento de escanear los diversos canales disponibles con la ayuda de la capa de acceso al medio. Después de haber considerado las opciones, la capa de red selecciona un nodo padre de su vecindario, y solicita a la capa de acceso al medio el procedimiento de asociación. El nodo padre asigna al nuevo nodo una dirección de 16 bits. La relación padre hijo establecida es resultado de la red en forma de árbol, y es también la base del algoritmo distribuido para asignación de nombres. El coordinador de la red ja el máximo número de routers y dispositivos finales que cada router puede tener como hijo, y también
ja la profundidad máxima del árbol. Los routers reciben un rango consecutivo de direcciones (16-bit), la primera es para ellos, las siguientes para asignar a sus dispositivos hijos.

Los algoritmos de enrutado dependen de la topología de la red. En una topología de red de tipo árbol, los paquetes solo pueden enviarse a través del link establecido entre los nodos padre e hijo (tree-based routing). Por la manera en que se asignan las direcciones, un router que necesite enviar un paquete puede fácilmente determinar a quién debe enviarlo, si a uno de sus router hijo o a uno

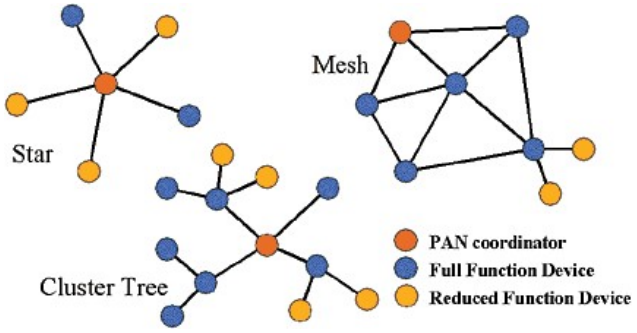

Figura 4.3: Topologías de red en ZigBee

de los dispositivos finales hijos. En caso de no pertenecer el paquete a alguno de sus nodos hijo, lo envía a su respectivo nodo padre. Este algoritmo de enrutado no es el más eficiente en consumo energético, pero es muy simple de implementar y puede utilizarse en redes que utilicen el protocolo de control de acceso al medio con beacon (Todos los routers y el coordinador envían beacons, se comunican a través de un protocolo CSMA-CA en ranura y duermen en la parte inactiva de la supertrama). El método obvio para minimizar el consumo energético es tener regiones su suficientemente grandes de inactividad.

La topología en malla es más compleja y no se puede utilizar el protocolo de acceso al medio con señalización beacon. Sin embargo, es más robusta y resistente a fallos. Los routers mantienen una tabla de enrutado y emplean un algoritmo de descubrimiento de rutas para construir y actualizar dicha tabla.

El descubrimiento de rutas es un proceso requerido para establecer las entradas en la tabla de rutas que permitan determinar el camino entre dos nodos que deseen comunicarse. Se basa en el conocido algoritmo de ruta Ad hoc On Demand Distance Vector (AODV) [53]. Cuando un nodo necesita la ruta a un cierto destino, se manda en broadcast una petición de ruta RREQ, y se propaga por la red hasta llegar a su destino. Este mensaje guarda en uno de sus campos el coste, que es la suma de los costes de todos los enlaces que ha atravesado. Este coste puede ser un valor fijo o ser calculado dinámicamente a través de la estimación de calidad de enlace provisto por la interfaz IEEE 802.15.4. Cada mensaje tiene un identificador único. Cuando un router recibe una petición consulta su tabla de descubrimiento de rutas. Si la dirección no existe, se crea una nueva entrada. Si existe, se compara el coste y se actualiza si es necesario. Si el nodo es el destino de la petición responde con un mensaje RREP destinado al primer nodo. Los nodos intermedios actualizan en el proceso sus tablas de enrutado.

Capa de aplicación

Una aplicación ZigBee consiste en un conjunto de objetos (Application Objects, APOs) dispersos en varios nodos de la red. Un APO es software que controla un elemento hardware del dispositivo (un sensor, un switch). A cada componente se le asigna un identificador único local, los otros componentes de la aplicación pueden utilizar dicho identificador, junto a la dirección de red del dispositivo en el que se encuentra, para comunicarse con el. El ZigBee Device

Object es un componente especial que ofrece servicios a los APO para que estos puedan formar de manera colectiva la aplicación distribuida: permite descubrir dispositivos en la red que ofrecen servicios determinados y proporciona, también, servicios de administración de red, comunicación y seguridad. La subcapa de aplicación, Application Sublayer (APS), provee el servicio de transferencia de información entre diferentes APO. Ver la figura 4.2.

4.2. Capa de acceso al medio

La propiedad más importante que debe de tener un protocolo de acceso al medio en las redes de sensores es la e ciencia energética. Otros atributos son la escalabilidad y la adaptabilidad a cambios, debido a la propia naturaleza de las redes. Los cambios en el tamaño, la densidad de nodos y la topología deberían ser manejados de forma rápida y eficiente. Otros atributos tales como la latencia y el ancho de banda son secundarios. Las propuestas de mejoras en la capa de control de acceso al medio (capa MAC) deben ser energéticamente eficientes, reduciendo las potenciales pérdidas [16]:

1. Colisiones. Cuando un nodo recibe al mismo tiempo dos paquetes, la información llega errónea y se debe descartar y retransmitir, lo que implica mayor consumo de energía.
2. Overhearing. Tiempo gastado en recibir paquetes que están destinados a otros nodos.
3. Control-packet overhead. Se debería minimizar la cantidad de paquetes de control que son enviados para realizar la transmisión de datos.
4. Idle listening. El tiempo que se gasta en escuchar un canal vacío.

Las redes de sensores deben ser tratadas teniendo en cuenta las propias características de las aplicaciones. Kulkarni [39] define tres tipos de patrones de comunicación en una red de sensores:

1. Broadcast. Es usado normalmente por una estación base para transmitir alguna información a toda la red. Por ejemplo una petición en una arquitectura de base de dato, o una actualización de programa para la red de sensores.
2. Local gossip: Cuando un nodo se comunica con sus vecinos.
3. Convergecast: Después de que los sensores detecten un evento, esa información debe llegar a un centro de información.

Las diferentes propuestas para la capa MAC se comentarán en el siguiente capítulo, ya que su objetivo principal es la eficiencia energética.

4.3. Enrutado

Las técnicas de enrutado tradicional, por ejemplo, las que se utilizan en TCP/IP, no son válidas para las redes de sensores debido a las restricciones que

imponen sus características ya comentadas. En las redes de sensores los nodos son dispuestos aleatoriamente, y el proceso de reconfiguración es relativamente habitual, debido, por ejemplo, a fallos de nodos intermedios, o a decisiones de cambio de estructura motivadas por e ciencia energética. Por tanto, asignar y mantener una estructura jerárquica es impracticable.

Los protocolos reactivos tales como AODV [47] anteriormente presentado en ZigBee, alivian algunos de de estos problemas, pero no son apropiados para redes con muchos nodos. Esto es debido a que para establecer una nueva ruta es necesario enviar un mensaje mediante broadcast a todos los nodos de la red. Los protocolos de enrutado para redes de sensores deberán tener en con- sideración, tanto el consumo energético, como el uso de memoria, y deberán requerir la mínima sobrecarga de mensajes. Deberán ser capaces de enrutar paquetes basados solo en la información intercambiada con los vecinos, ser resistentes a cambios en la red y a fallos de nodo.

4.3.1. Enrutado

Las aplicaciones mas simples son las que un conjunto de sensores envían la información hacia un único nodo. En este tipo de aplicaciones el enrutado es trivial. Cada nodo debe conocer el padre hacia el nodo destino y enviar los mensajes por el. Un ejemplo es propuesto en [45]. Este tipo de técnicas son muy sencillas pero no son apropiadas para aplicaciones más complejas que requieran comunicación nodo a nodo.

4.3.2. Enrutado geográ co avaricioso

A continuación se proponen algoritmos que puedan soportar la comunicación peer-to-peer, esto es entre dos nodos cualesquiera de la red. La idea principal es que a los nodos se les asigna una localización, y por tanto se puede definir una distancia entre dos localizaciones. Cada nodo podría enviar periódicamente por broadcast su localización a sus vecinos. En el algoritmo de enrutado un nodo envía el paquete al vecino que minimice la distancia restante al destino. Aunque este algoritmo muy sencillo tiene dos problemas principales:

1. Cómo un nodo descubre sus coordenadas, es decir, cómo determinamos la localización de cada nodo.

2. Qué pasa cuando la ruta avariciosa falla.

El primero de ellos, el problema de localización, consiste en asignar a cada nodo una tupla con las coordenadas en que se encuentra. Una posibilidad sería utilizar las coordenadas físicas (mediante un GPS, incorporado en el dispositivo o de manera manual) o permitir a los nodos aproximar su posición física a través de la información de conexión a otros dispositivos equipados con GPS. Otra alternativa son los protocolos que asignan coordenadas virtuales a todos los nodos, mediante posición relativa en función de la conectividad entre los nodos. En la siguiente sección se tratará el segundo problema y las soluciones propuestas.

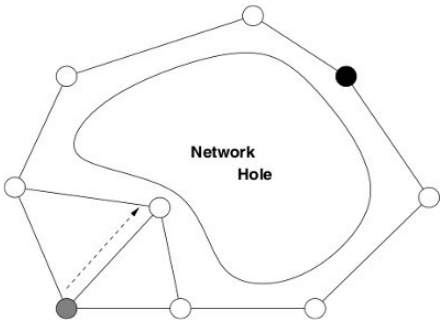

Figura 4.4: Fallo de enrutado en algoritmo avaricioso.

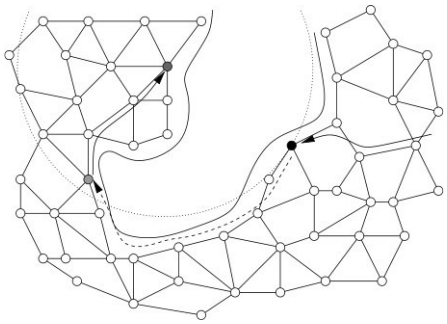

Figura 4.5: Enrutado avaricioso con modo perímetro .

Fallo en el enrutado por ruta avariciosa

El enrutado avaricioso sólo no puede garantizar el envío de las tramas en todas las topologías de red. En la figura 4.4 se puede ver una ejemplo en el que un nodo no puede remitir el paquete porque él está más cerca del destino que sus vecinos. La solución es incluir algún tipo de técnica para cuando la búsqueda avariciosa falle.

Cuando las coordenadas son en dos dimensiones, una de las propuestas [37] es utilizar lo que ellos denominan modo perímetro cuando la búsqueda avariciosa falle, y volver al modo avaricioso cuando sea posible (ver figura 4.5). Estas técnicas son sólo aplicables a grafos planares (grafos que pueden ser pintados en un plano sin que se corten sus aristas). Sólo son conocidos algoritmos distribuidos para crear grafos planares para dos dimensiones. Hay otra posibilidad, las coordenadas de más de dos dimensiones pueden ser mapeadas en el espacio de 2 dimensiones y los anteriores algoritmos pueden ser utilizados. Sin embargo, esta conversión normalmente pierde información y el resultado es un comportamiento peor.

La alternativa es utilizar otro método de recuperación distinto. BVR [21] usa un conjunto de nodos seleccionado aleatoriamente como nodos ancla, y define coordenadas como las distancia hop a dichos nodos ancla. Los nodos tienen preferencia de enviar el paquete al nodo ancla si está más cerca que el destino. Cuando la búsqueda avariciosa falla, este algoritmo enruta el paquete hacia el camino del nodo ancla que está más cerca del destino. Cada nodo del camino

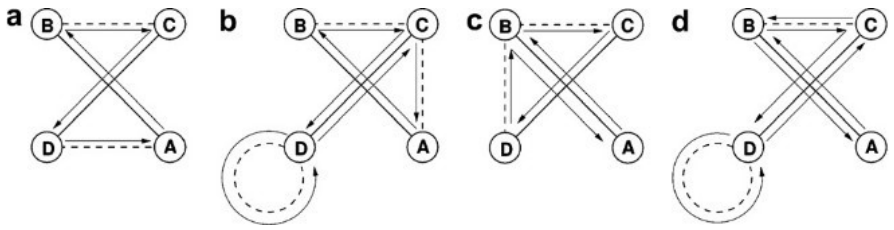

Figura 4.6: Los cuatro posibles casos donde el algoritmo CDLP decide eliminar o mantener un enlace.

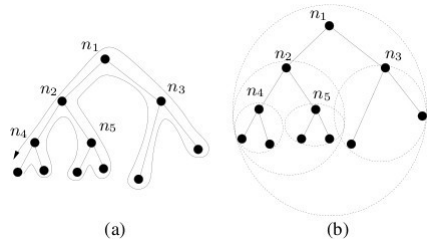

Figura 4.7: Árbol abarcador del algoritmo GDSTR.

intentará la búsqueda avariciosa o mandará el paquete a su padre en caso de fallo. Si el paquete llega al ancla se produce un envío por inundación pero solo a la profundidad que se encuentre el nodo.

Otra propuesta, CLDP [38], es e caz incluso en caso de información de localización equivocada, obstáculos, etc. Envía sondas a sus nodos vecinos siempre al primero por la derecha, grabando cada nodo y volviendo al original. Los resultados posibles se pueden reducir a los siguientes 4 ejemplos (figura 4.6). En el caso (a) el eje (A,B) es cruzado por (C,D) y cada eje es atravesado una vez, luego el nodo A puede eliminar (A,B) sin problemas. En el caso (b), el eje (C,D) es atravesado en ambas direcciones, lo que significa que eliminando dicho eje, desconectaríamos D. Sin embargo, (A,B) puede ser eliminado sin problemas. Si el eje (A,B) es atravesado 2 veces en ambas direcciones pero (C,D) es atravesado solo es sólo atravesado una vez (c), puede eliminarse (C,D) pero no (A, B), el nodo A informa a C y D de eliminar la comunicación. El último caso (d), ocurre cuando ambos ejes (A, B) y (C, D) son atravesados dos veces. Aquí no se puede eliminar ningún segmento porque desconectaría la red. CDLP mantiene el mínimo número de conexiones para asegurar conectividad y eliminar completamente links unidireccionales. Sin embargo, tiene gran cantidad de sobrecarga de mensajes inicialmente, y también necesita un protocolo de bloqueo para que dos nodos no modifiquen la red al mismo tiempo.

GDSTR [40] es un algoritmo más reciente que evita usar la planarización. Se trata de crear un árbol abarcador con todos los nodos de la red (ver figura 4.7). Cada nodo almacena información geométrica de su subárbol de descendientes. Cuando la ruta avariciosa falla, el algoritmo busca en los descendientes del nodo en el que se ha parado. En caso de error, pasa el paquete al nodo al padre, con el objetivo de explorar el resto de la red. El algoritmo solo explora un subárbol

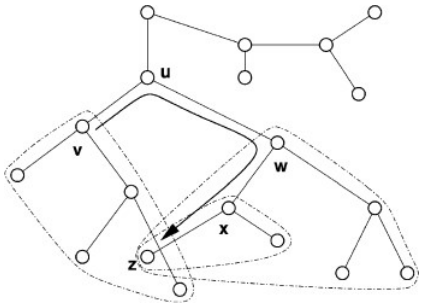

Figura 4.8: Búsqueda del nodo z en el algoritmo GDSTR.

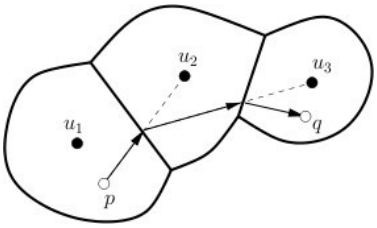

Figura 4.9: El enrutamiento jeráquico en GLIDER

si se tiene la información de localización de dicho destino.

La figura 4.8 muestra el algoritmo de recuperación cuando la búsqueda avariciosa falla. El envío de un paquete hacia z falla a través del nodo v. El árbol ha sido ya explorado antes de que pasar el paquete, luego v decide enviar el paquete a su padre u. El nodo w contiene z luego el proceso de búsqueda vuelve a empezar en dicho sub-árbol.

4.3.3. Enrutado jerárquico

Los algoritmos de enrutado avaricioso son eficientes si los nodos están dispersados de manera densa y regular. Pero estos algoritmos pueden fallar en presencia de zonas vacías o en zonas donde hay obstáculos que introducen discontinuidades en la estructura topológica. Recientemente se han propuesto crear una representación global de la red entera y guardar dicha información en cada nodo. Esta representación dividiría e identificaría la red como un conjunto de regiones de topología regular. Cada región dispondría de un nodo coordinador y se establecería un algoritmo greedy para utilizarlo dentro de la región. Las decisiones de enrutamiento consisten en identificar un camino entre las distintas regiones hasta llegar a la región destino. Algunos de los inconvenientes son la complejidad en dividir una estructura muy compleja en regiones más o menos regulares, y que el tamaño de dicha representación debería ser el menor posible.

GLIDER [20] selecciona un conjunto de nodos como puntos de referencia y de ne elementos de ese conjunto los nodos que están más cerca de este punto de referencia que de ningún otro. La topología de la red de alto nivel consiste en la representación como grafo de nodos de los puntos de referencia. Esto es

suficiente para planear el enrutado entre conjuntos. Dentro de cada conjunto, a cada nodo se le asigna coordenadas basado en el identificador del punto de referencia y la distancia a este último como a los puntos de referencia vecinos. El enrutamiento en dos puntos se realiza de la siguiente manera: (ver la figura 4.9) en primer lugar se consulta la topología de red de alto nivel para determinar cuál es la siguiente región. Entonces el enrutado intra-región determina cual es el siguiente nodo para acercarse a la siguiente intra-región. Cuando elpaquete por n llega a la región destino, el enrutamiento intra-región lo dirige hasta el nodo destino. El enrutamiento intra-región se colapsa si cae en mínimo local. La selección de los puntos de referencia puede ser manual o automática.

5

Eficiencia energética

La eficiencia energética es una de las cuestiones más importante en las redes de sensores. Es importante desarrollar técnicas que maximicen la vida útil de la red. El consumo de energía se debe evitar en todos los niveles tanto hardware como software, tanto en el diseño de bajo nivel (sistema operativo, middleware) como en el de alto nivel (aplicaciones). Los últimos dispositivos permiten desactivar determinados componentes durante un tiempo (sensores, convertidores analógico digitales y radio transmisores) con el objetivo de poder ahorrar energía. El transmisor de radio es el componente que suele consumir más energía de todo el dispositivo. Por tanto, es deseable no tener el transmisor siempre encendido. Mientras que los sensores pueden ser activados y desactivados cuando el nodo lo decida individualmente, la activación y desactivación de la comunicación es un tema mucho más complejo, y se debe de tener en cuenta la red de manera global.

Como hemos visto en la anterior sección, ha habido muchos intentos de mejorar los algoritmos de acceso al medio, pero cada vez se le está dando más importancia a las capas superiores (red y aplicación). Se trata de adaptar el uso de la radio a las necesidades y el patrón de comportamiento de la aplicación para conseguir ahorros de energía más notables. A continuación comentaremos algunas de las soluciones propuestas para ahorrar energía.

5.1. Conjunto dominante conectado

En las redes densas hay muchos nodos muy cercanos que son prácticamente iguales desde el punto de vista de las rutas. La idea es seleccionar un conjunto de nodos para que siempre esté conectado (backbone) y que puedan almacenar mensajes para los demás nodos si están inactivos. De esta manera, el resto de nodos puede permanecer inactivos durante periodos de tiempo considerables. Como los nodos que están siempre conectados consumen más energía, estos roles debería intercambiarse cada cierto tiempo.

GAF [65] es un ejemplo de este tipo de técnicas. Se basa en GPS para determinar la localización y dividir la red en una cuadrícula. Utiliza un algoritmo distribuido para elegir un líder en cada cuadrícula. Los líderes utilizan protocolos estándares como AODV o DSR para comunicarse.

Span [10] propone otra solución en la que no se utiliza información de GPS

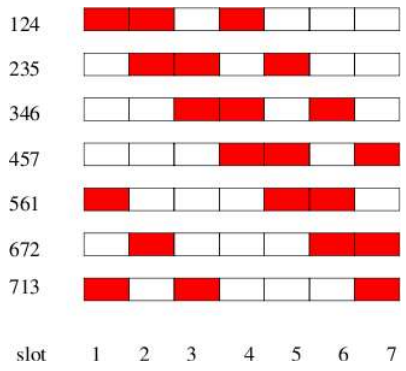

Figura 5.1: Programa de ranuras activas para los diferentes nodos de una red.

para localizar los nodos y mantener el backbone. En este caso no hay una estructura ja para el backbone, cada nodo periódicamente determina si debe pertenecer al backbone o no. Cada nodo evalúa la utilidad relacionada con el número de vecinos que estarían conectados si fuera un nodo del backbone. Un método aleatorio tiene en cuenta también la energía que le queda al nodo decide la transición.

5.2. Enfoque en la capa de acceso al medio

Este tipo de soluciones solo se preocupan de la capa de acceso al medio. De esta manera las capas superiores no necesiten preocuparse de ese problema. Las soluciones que veremos a continuación suelen ser bastante inflexibles en la tasa de envío, y tener bastante sobrecarga, lo que puede provocar largas latencias en redes grandes, ya que en este nivel la coordinación solo se realiza a nivel de nodo.

5.2.1. Ranuras de tiempo

En este tipo de soluciones el tiempo se divide en periodos, cada uno conteniendo un cierto número de ranuras de tamaño fijo [67]. Los nodos tienen un conjunto prede nido de ranuras en las que anuncian su programa de ranuras activas y escuchan las peticiones. La figura 5.1 ilustra un ejemplo con 7 ranuras, para todos los nodos de la red siempre es posible que dos vecinos coincidan en una ranura activa para comunicarse. Para un numero jo de ranuras, cuanto mayor sea el numero de ranuras activas, menor será la latencia hop-to-hop, pero también se consumirá mayor cantidad de energía.

5.2.2. TDMA

Es la solución obvia para evitar colisiones y el tiempo de escucha innecesario. Cada nodo sabe exactamente cuando estar activo y cuando enviar para evitar colisiones. En los protocolos TDMA clásicos todos los nodos pueden comunicarse entre sí, el nodo Master envía al principio una supertrama y los demás nodos reservan ranuras de tiempo. Este algoritmo es útil para redes pequeñas en las

que todos los elementos están conectados, pero para grandes redes multi-hop es totalmente impracticable por razones de sincronización.

La capa de acceso al medio IEEE 802.15.4 ofrece un modo de operación basado en supertrama, donde un protocolo CSMA-CA es seguido de un periodo libre de colisiones. Las ranuras son asignadas por el coordinador de red y hay una región inactiva. ZigBee utiliza este modo para las topología estrella, donde los nodos pueden hablar solo con su coordinador, y en la topología de árbol, donde cada los nodos internos se coordinan con sus nodos padres. Los nodos internos deben empezar su supertrama con un o set apropiado para evitar solapamiento de regiones activas.

TRAMA [55] es un algoritmo basado en TDMA propuesto para hacer más eficiente el uso energético. En cada cierto periodo un algoritmo distribuido es usado para seleccionar un transmisor entre los vecinos situados a dos saltos. Este tipo de elección elimina el problema del terminal oculto y asegura que los nodos conectados a un salto recibirán la transmisión sin colisiones. El tiempo se divide en una parte de acceso aleatorio y otra de acceso programado. Se consiguen altos porcentajes de tiempo inactivo, sin embargo, tiene algunas desventajas tales como la necesidad de sincronización en toda la red y una añadida complejidad computacional considerable del algoritmo.

5.2.3. S-MAC, T-MAC, DS-MAC

Otro enfoque propone dividir el tiempo en periodos de activación y desactivación. Los nodos vecinos deben comunicarse para intercambiar información acerca de sus regiones activas.

Sensor-MAC (S-MAC) [66] propone sincronización administrada localmente y periodos de actividad e inactividad. Los nodos vecinos forman clusters virtuales para organizar el tiempo de desactivación. Si dos nodos vecinos residen en dos clústeres virtuales ambos se activan en los dos periodos. Una desventaja de este algoritmo es que esta posibilidad resulta en mayor cantidad de energía consumida. El tiempo activo tiene una duración ja y está dividido en dos partes. La primera parte está reservada para recibir un mensaje SYNC, donde se informa del tiempo para la siguiente ventana de activación. Los tiempos de inactividad resultan en una alta latencia, especialmente en los algoritmos de en- rutado multi-hop.

Timeot MAC (T-MAC) [60] intenta solucionar estos problemas. las ventanas activas no tienen una duración fija, lo que les permite adaptarse a distintas cargas de trabajo. Cada nodo se activa al principio de su ventana, pero se desactiva si ningún evento ocurre en un cierto periodo. La recepción de un mensaje es un evento que prolonga la duración de la ventana activa.

En Dynamic Sensor MAC (DS-MAC) [43], también basado en S-MAC, los nodos comienzan con el mismo período de duración pero se permite doblarlo o dejarlo a la mitad en función de la carga de trabajo. En ambos casos la región activa es constante y la latencia disminuye.

5.2.4. Eficiencia energética de IEEE 802.15.4

Conseguir una tasa de transmisión deseada maximizando la vida útil de los nodos suelen ser metas contradictorias. La primera decisión consiste en utilizar

es utilizar modos con beacon (con ranuras de tiempo) o sin beacon (sin ranuras). Estudios realizados dicen que CSMA-CA sin ranuras es capaz de conseguir más utilización del canal que CSMA-CA con ranuras, permitiendo escalabilidad y auto-organización, pero sufre el conocido problema del terminal oculto en en- tornos multi-salto. El problema del terminal oculto ocurre cuando un receptor está en el rango de dos transmisores que no se pueden ver entre ellos. En este caso ambos transmisores pueden enviar a la vez y producirse una colisión. Este problema puede ser serio en redes grandes. En este modo no se tiene mecanismos de ahorro de energía y tampoco se garantiza el tiempo de envío.

El otro modo, basado en TDMA, adopta un periodo coordinado de de-sactivación, permitiendo mayor eficiencia energética, pero sólo utilizable para topologías estrella o árbol con pocos nodos.

5.3. Enfoque con múltiples capas

La información de las capas más altas de la pila de protocolos puede ser combinada con los enfoques propuestos de ahorro energético en la capa MAC para conseguir un mayor ahorro energético. La capa de red y aplicación tienen mucha mas información sobre los patrones de comunicación, topología de red y tasas de transmisión que puede ser usadas para obtener mejores esquemas de activación de radio.

Zheng and Kravets [68] proponen un protocolo usando la información de la capa de red para llevar a la capa MAC a modos activos o de ahorro de energía. La llegada de un mensaje de la capa de red inicia la transición a estado activo de la capa MAC. Mientras sigan llegando mensajes, el temporizador interno se vuelve a inicializar. Si el temporizador expira, indica que no se espera más trá co y se debe pasar a modo inactivo.

En las aplicaciones más sencillas, los nodos recolectan datos del entorno y lo mandan a un único nodo. En este tipo de comunicación en árbol es posible optimizar el consumo energético. El tiempo se divide en periodos de ranuras de tamaño jo. Un nodo que desee enviar información a su padre debe reservar una ranura de tiempo. Una vez reservado no hay colisiones y los nodos solo necesitan estar activos en las ranuras reservadas. La principal limitación de este esquema es que no es apropiado para comunicaciones peer-to-peer, necesarias en aplicaciones más complejas.

Sichitiu [57] propone otro enfoque que si es válido para comunicaciones peer-to-peer y no se limita a ranuras de tamaño jo. La operación se divide en dos partes, la primera de configuración y otra parte de envío de datos. En la primera parte, se establece una ruta con la ayudad e la capa de red y el intercambio RTS/CTS/configuración de ruta/ACK. El intervalo de tiempo entre enviar el RTS y recibir el ACK debe ser más pequeño que el tiempo que se demore en enviar los datos, sino no es útil. Si los anteriores intercambios se realizaron con éxito, ambos nodos reservan un periodo de tiempo para la transmisión de datos. Si el nodo inicial no recibiera la señal ACK, se reintenta más tarde. Durante la segunda fase se envían los datos sin necesidad de utilizar las señales RTS/CTS. Puede ocurrir que haya una colisión con otro mensaje de configuración para otra ruta de datos, pero es poco probable. El protocolo se adapta a diferentes tasas de transmisión de datos reservando múltiples ranuras.

6

Sistemas inteligentes

A continuación se muestran algunos de los escenarios de donde se han aplicado satisfactoriamente la tecnología de redes de sensores. Los sistemas de monitorización y control en red han mostrado su efectividad con un papel clave de los sistemas intermediarios [27, 29-30]. Se ha realizado una búsqueda en la literatura en distintos ámbitos en el que las redes de sensores tienen una clara utilidad. A continuación, se presentan algunos escenarios.

Aplicaciones a la salud

En el campo de la medicina las redes de sensores pueden usarse para monitorizar parámetros fisiológicos de los pacientes, tales como los latidos del corazón o la presión de la sangre, e informar al hospital si alguno de estos parámetros ha cambiado [22, 48].

También podemos destacar el uso en las unidades de cuidados intensivos. En ellas, el paciente normalmente tiene conectados varios sensores que se unen a través de cables a los equipamientos médicos colocados a un lado de la cama. Utilizar sensores inalámbricos sería mucho más cómodo para el paciente y le permitiría mayor movilidad. Otro escenario es la tele-asistencia médica. Tradicionalmente su objetivo es avisar cuando existe algún tipo de crisis. La tecnología de redes de sensores puede proporcionar nuevos dispositivos más potentes para monitorizar las condiciones de enfermedades crónicas (fallos del corazón, enfermedades del pulmón, diabetes) desde el propio hogar del paciente, e incluso mediante la conexión a Internet, puede enviar las alarmas oportunas o simple- mente recopilar datos. Otros ejemplos de aplicaciones en la salud de las redes de sensores se pueden encontrar en [2, 35].

Aplicaciones en la monitorización de especies animales y medio ambiente

El uso en vastas áreas de bosque o de océano mediante la monitorización control de múltiples variables, como temperatura, humedad, fuego, actividad sísmica así como otras. También ayudan a expertos a diagnosticar o prevenir un problema o urgencia y además minimiza el impacto ambiental de la presencia humana.

Monitorizar el entorno y habitat es un campo de aplicación muy interesante y con enorme potencial. La capacidad de las redes de sensores para introducir

pequeños microsensores en distintos puntos del medio ambiente es muy positiva. Además, uno de los problemas tradicionales en este tipo de escenarios es que la presencia humana en determinados entornos es un grave factor de impacto. El uso de estas tecnologías permitiría una recolección de datos minimizando la presencia humana. Se han llevado a cabo algunas experiencias como las realizadas en Great Duck Island [47], la monitorización de un volcán en Ecuador [63], o un sistema de control de inundaciones [15].

Sistemas en agricultura y ganadería

La agricultura y la ganadería son también áreas en las que la introducción de las redes de sensores puede tener bastante utilidad. Por un lado, en la agricultura [61] sería posible monitorizar las condiciones climáticas de extensas áreas y localizaciones remotas de una manera rápida y e caz. En el campo de la ganadería, por su parte se podría monitorizar las condiciones vitales de los animales, además de permitir su identificación y localización [8].

Aplicaciones militares

Las redes de sensores serían de gran utilidad en el campo militar. La introducción de esta tecnología permitiría desplegar redes en las líneas enemigas para observar movimientos de tropas o recoger información geográfica. Inicialmente fueron las principales precursoras de estas redes. Sin embargo, hay poca información de dominio público. Asegurar zonas, detectar intrusos y monitorizar tráfico de vehículos en terreno abierto han sido objeto del proyecto DARPA SensIT en 2001. Otros sistemas, por ejemplo, permiten la localización de francotiradores con redes de sensores [58].

Sistemas inteligentes aplicados al tráfico

Este campo permite varias aplicaciones con objetivos como una mayor seguridad en la carretera o mejorar el flujo del tráfico. La introducción de las redes de sensores, permitirían soluciones de bajo coste para mejorar las condiciones de la red de carreteras. Se han propuesto, entre otros, la monitorización de la polución, identificar la fuente de situaciones anómalas, monitorizar la densidad del tráfico [24].

Otros proyectos más ambiciosos, como Vehicular Networks (CarTalk 2000) [64], proponen desarrollar un sistema cooperativo entre los coches. Los coches podrían, mediante redes ad-hoc, avisar de atascos o accidentes. Se definiría una interfaz estándar de velocidad, temperatura, condiciones de la carretera, localización y posición. Además, los vehículos tienen una ventaja con respecto a otros entornos, y es que no hay problemas de alimentación energía.

Intelligent Building Monitoring

Es estado de las estructuras, como puentes o edificios, requieren una monitorizacion continua para asegurar fiabilidad y seguridad. Desplegar una red de sensores podría permitiría detectar y localizar daños, y predecir el tiempo de vida. En [52] se propone un ejemplo de monitorización de estructuras mediante el estudio de frecuencias y vibraciones.

7

Conclusiones

En este trabajo se ha revisado la literatura reciente en redes de sensores inalámbricos, intentando dar una visión general en los aspectos de mayor importancia. Las redes de sensores es un área de investigación activa en la que los últimos años se han realizado bastantes aportaciones. Han sido resueltos muchos de los problemas, pero todavía queda bastante trabajo que realizar.

En el campo de las plataformas hardware se debe hacer un esfuerzo aún mayor en la miniaturización. Cuanto más pequeño sea el dispositivo, más posibilidades tendrá de ser integrado en diversos entornos. El éxito de las redes de sensores va a depender en gran medida de esta capacidad.

Los desarrollos software analizados intentan ofrecer un entorno que facilite el desarrollo de aplicaciones en las redes de sensores. Existen algunos sistemas operativos (por ejemplo TinyOS), entendidos como una pequeña abstracción del hardware, así como una funcionalidad básica para la creación de tareas. Otras propuestas proporcionan un nivel mayor de abstracción a través de middleware, haciendo transparente la comunicación entre distintas aplicaciones en distintos nodos. Debido a su carácter específico, su utilidad va a depender del tipo de aplicación al que se destine. Es interesante algunas propuestas que permiten de nir el comportamiento global de la red, dejando las tareas de bajo nivel a herramientas automáticas.

Se ha revisado el estándar ZigBee/IEEE 802.15.4 y la literatura reciente en este tema, resumiendo las soluciones adoptadas en el estándar y las nuevas propuestas surgidas en los últimos años. ZigBee ha sido diseñado para redes pequeñas con topologías de red sencillas. Su protocolo de enrutamiento basado en AODV no es apropiado para redes con un número considerable de nodos. La tendencia en investigación es hacia enrutado geográfico, pero falta estudiar y estandarizar estos protocolos. Por el momento, las redes actuales son todavía de poco tamaño y el estándar ZigBee es suficientemente útil, estable y fiable para estas redes.

El consumo energético sigue siendo uno de los problemas abiertos. La comunicación por red inalámbrica es elemento de mayor consumo energético, por ello es el que más se ha intentado optimizar. El enfoque de ZigBee para mejorar la eficiencia energética se basa únicamente en la capa física y en la capa de control de acceso al medio. Han sido propuestos otros enfoques para la capa de acceso al medio que minimicen el consumo energético, pero en general son bastante inflexibles para diferentes tasas de transmisión. Recientemente se han propuesto

soluciones basadas en la cooperación de diferentes capas en la pila de protocolos. Este tipo de enfoque permite utilizar la información de la capa de red y/o de la capa de aplicación para mejorar la eficiencia. La solución parece ser que los nodos de la red de sensores, en cada momento, puedan adaptar automáticamente los parámetros de red en función de las necesidades de las aplicaciones.

La implantación de las redes de sensores no ha tenido un crecimiento tan rápido como se esperaba en un principio. Hay algunos obstáculos que están retrasando el despliegue total de estas redes. Podemos destacar algunos problemas como la estandarización no analizados. Aún queda resolver los problemas de las redes con muchos nodos, que si bien no es un problema para las redes actuales, la falta de un estándar cerrado crea inseguridad. En algunos entornos, la compatibilidad con los sistemas viejos es un problema, para adoptar esta tecnología se va a tener que cambiar parte de la infraestructura que aún funciona. Los problemas energéticos todavía no han sido solucionados, hace falta una mayor autonomía energética de los nodos. Además, las redes inalámbricas se consideran aún una opción arriesgada para el control de procesos, debido a la que su fiabilidad no ha sido completamente demostrada. Sin embargo, conforme se vayan obteniendo más resultados en entornos reales esta tecnología se integrará en los distintos entornos. Cabe esperar que en los próximos años se convierta en una tecnología ubicua.

Bibliografía

[1] H. Abrach, S. Bhatti, J. Carlson, H. Dai, J. Rose, A. Sheth, B. Shucker, J. Deng, and R. Han. Mantis: system support for multimodal networks of in-situ sensors. In WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications, pages 50-59, New York, NY, USA, 2003. ACM.

[2] G. Amato, S. Chessa, F. Conforti, A. Macerata, and C. Marchesi. Health Care Monitoring of Mobile Patients. Ercim news. in, 60.

[3] P. Baronti, P. Pillai, V.W.C. Chook, S. Chessa, A. Gotta, and Y.F. Hu. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards. Computer Communications, 30(7):1655 1695,2007.

[4] R. Barr, J.C. Bicket, D.S. Dantas, B. Du, T.W.D. Kim, B. Zhou, and E.G. Sirer. On the Need for System-Level Support for Ad hoc and Sensor Networks. Operating Systems, 36(2):1 5, 2002.

[5] J. Beutel, O. Kasten, F. Mattern, K. Romer, F. Siegemund, and L. Thiele. Prototyping Wireless Sensor Network Applications with BTnodes. Wire- less Sensor Networks: First European Workshop, EWSN 2004, Berlin, Ger- many, January 19-21, 2004, Proceedings, 2004.

[6] S. Bhatti, J. Carlson, H. Dai, J. Deng, J. Rose, A. Sheth, B. Shucker, C. Gruenwald, A. Torgerson, and R. Han. MANTIS OS: An Embedded Multithreaded Operating System for Wireless Micro Sensor Platforms. Mobile Networks and Applications, 10(4):563-579, 2005.

[7] P. Bonnet, J. Gehrke, and P. Scshadri. Towards Sensor Database Systems. Mobile Data Management: Second International Conference, MDM 2001, Hong Kong, China, January 2001: Proceedings, 2001.

[8] P. Bonnet, M. Leopold, and K. Madsen. Hogthrob: towards a sensor network infrastructure for sow monitoring (wireless sensor network special day). Design, Automation and Test in Europe, 2006. DATE'06. Proceed- ings, 1, 2006.

[9] BTnodes. http://www.btnode.ethz.ch.

[10] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An Energy- E cient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks. Wireless Networks, 8(5):481-494, 2002.

[11] P. Costa and G.P. Picco. Semi-probabilistic content-based publish-subscribe. Proc. of the 25th Int. Conf. on Distributed Computing Systems, 25:575 585, 2005.

[12] P. Costa, GP Picco, and S. Rossetto. Publish-subscribe on sensor networks: a semi-probabilistic approach. Mobile Adhoc and Sensor Systems Confer- ence, 2005. IEEE International Conference on, pages 323 332, 2005.

[13] Crossbow technology. http://www.xbow.com.

[14] G. Cugola, A.L. Murphy, and G.P. Picco. Content-Based Publish-Subscribe in aMobile Environment. The Handbook of Mobile Middleware, 2006.

[15] D. De Roure, C. Hutton, D. Cruickshank, E.L. Kuan, J. Neal, R. Roddis, A. Stanford-Clark, S. Vivekanandan, and J. Zhou. FloodNet-Improving ood warning times using pervasive and Grid computing. Neal, J, Atkinson, P and Hutton, C, 2006.

[16] I. Demirkol, C. Ersoy, and F. Alagoz. MAC protocols for wireless sensor networks: a survey. Communications Magazine, IEEE, 44(4):115 121, 2006.

[17] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - a lightweight and exible operating system for tiny networked sensors. Local Computer Networks, 2004. 29th Annual IEEE International Conference on, pages 455-462, 2004.

[18] Dust Networks. http://www.dustnetworks.com.

[19] P.T.H. EUGSTER, P.A. FELBER, R. GUERRAOUI, and A.M. KER-MARREC. The Many Faces of Publish/Subscribe. ACM Computing Surveys, 35(2):114 131, 2003.

[20] Q. Fang, J. Gao, L.J. Guibas, V. de Silva, and L. Zhang. GLIDER: Gradient landmark-based distributed routing for sensor networks. Proc. of the 24th Conference of the IEEE Communication Society (INFOCOM), 2005.

[21] R. Fonseca, S. Ratnasamy, J. Zhao, C.T. Ee, D. Culler, S. Shenker, and I. Stoica. Beacon vector routing: Scalable point-to-point routing in wire- less sensornets. Proceedings of the Second USENIX/ACM Symposium on Network Systems Design and Implementation (NSDI 2005), 2005.

[22] T. Gao, D. Greenspan, and M. Welsh. Improving patient monitoring and tracking in emergency response. Proceedings of the International Conference on Information Communication Technologies in Health, July, 2005.

[23] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC language: A holistic approach to networked embedded systems. Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation, pages 1 11, 2003.

[24] G. Giuliano and J. Heidemann. Rapidly deployable sensors for vehicle counting and classi cation. 2004.

[25] R. Gummadi, O. Gnawali, and R. Govindan. Macro-programming Wireless Sensor Networks Using Kairos. Distributed Computing in Sensor Systems: First IEEE International Conference, DCOSS 2005, Marina Del Rey, CA, USA, June 30-July 1, 2005: Proceedings, 2005.

[26] R. E. Haber, J.R. Alique, A. Alique, R. H. Haber. Controlling a complex electromechanical process on the basis of a neurofuzzy approach. Future Generation Computer Systems 21(7), 1083-1095, 2005

[27] R. E. Haber, K. Cantillo, J.E. Jiménez, Networked sensing for high-speed machining processes based on CORBA. Sensors and Actuators, A: Physical, 119(2), 418-426, 2005.

[28] Haber-Guerra, R.; Liang, S. Y.; Alique, J. R.; Haber-Haber, R., Fuzzy control of spindle torque in high-speed milling processes. Journal of Manufacturing Science and Engineering, Transactions of the ASME 128(4), 1014-1018, 2006.

[29] R.M. Del Toro, M.C. Schmittdiel, R.E. Haber-Guerra, R. Haber-Haber, In System identification of the high performance drilling process for network-based control. 2007 Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, DETC2007, pages 827-834, 2007.

[30] K. Cantillo, R.E. Haber, J.E. Jiménez, A. Alique, R. Galán, CORBA-Based open platform for processes monitoring. An application to a complex electromechanical process. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 3036, 523-526, 2004.

[31] R.E. Haber, J.R. Alique, A. Alique, J.E. Jiménez, Embedded fuzzy control system: Application to an electromechanical system. In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) Vol. 2658, 812-821, 2003.

[32] S. Hadim and N. Mohamed. Middleware: middleware challenges and approaches for wireless sensor networks. Distributed Systems Online, IEEE, 7(3), 2006.

[33] J. Heidemann and R. Govindan. An Overview of Embedded Sen- sor Networks. Technical report, Technical Report ISI-TR-2004-594, USC/Information Sciences Institute, 2004, 2005.

[34] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. ACM SIGPLAN Notices, 35(11):93-104, 2000.

[35] F. Hu, M. Jiang, L. Celentano, and Y. Xiao. Robust medical ad hoc sensor networks (MASN) with wavelet-based ECG data mining. Ad Hoc Networks, 2007.

[36] Intrinsyc Cerfcube. http://www.intrinsyc.com/.

[37] B. Karp and HT Kung. GPSR: greedy perimeter stateless routing for wire- less networks. ACM Press New York, NY, USA, 2000.

[38] Y.J. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic routing made practical. Proceedings of the 2nd Symposium on Networked Systems Design and Implementation (NSDI 2005), 2005.

[39] SS Kulkarni. TDMA service for sensor networks. Distributed Computing Systems Workshops, 2004. Proceedings. 24th International Conference on, pages 604 609, 2004.

[40] B. Leong, B. Liskov, and R. Morris. Geographic Routing without Planarization. Proceedings of the 3rd Symposium on Networked Systems Design and Implementation, San Jose, Costa Rica, pages 339-352, 2006.

[41] P. Levis and D. Culler. Maté: a tiny virtual machine for sensor networks. ACM SIGOPS Operating Systems Review, 36(5):85-95, 2002.

[42] P.A. Levis, D.E. Gay, and D.E. Culler. Bridging the Gap: Programming Sensor Networks with Application Speci c Virtual Machines. Computer Science Division, University of California, 2004.

[43] P. Lin, C. Qiao, and X. Wang. Medium access control with a dynamic duty cycle for sensor networks. Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE, 3, 2004.

[44] T. Liu and M. Martonosi. Impala: a middleware system for managing autonomic, parallel sensor systems. Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming, pages 107-118, 2003.

[45] S. Madden, R. Szewczyk, MJ Franklin, and D. Culler. Supporting aggregate queries over ad-hoc wireless sensor networks. Mobile Computing Systems and Applications, 2002. Proceedings Fourth IEEE Workshop on, 49-58, 2002.

[46] S.R. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong. TinyDB: an acquisitional query processing system for sensor networks. ACM Transactions on Database Systems (TODS), 30(1):122-173, 2005.

[47] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, pages 88 97, 2002.

[48] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton. CodeBlue: An Ad Hoc Sensor Network Infrastructure for Emergency Medical Care. International Workshop on Wearable and Implantable Body Sensor Networks, 2004.

[49] Mote iv. http://www.moteiv.com.

[50] A. Murphy and W. Heinzelman. Milan: Middleware linking applications and networks. University of Rochester, Tech. Rep. TR-795, 2002.

[51] M. Ouwerkerk, F. Pasveer, N. Engin, P. Res, and N. Eindhoven. SAND: a modular application development platform for miniature wireless sen- sors. Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on, page 5, 2006.

[52] J. Paek, K. Chintalapudi, R. Govindan, J. Ca rey, and S. Masri. A Wire- less Sensor Network for Structural Health Monitoring: Performance and Experience. Embedded Networked Sensors, 2005. EmNetS-II. The Second IEEE Workshop on, pages 1 10, 2005.

[53] C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, 2:90 100, 1999.

[54] J. Polastre, R. Szewczyk, C. Sharp, and D. Culler. The mote revolution: Low power wireless sensor network devices. Proceedings of Hot Chips 16: A Symposium on High Performance Chips, 2004.

[55] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves. Energy-E cient, Collision-Free Medium Access Control for Wireless Sensor Networks. Wireless Networks, 12(1):63-78, 2006.

[56] Sentilla. http://www.sentilla.com/moteiv-transition.html.

[57] M.L. Sichitiu. Cross-Layer Scheduling for Power E ciency in Wireless Sensor Networks. IEEE Infocom, 2004.

[58] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. Proceedings of the 2nd international conference on Embedded networked sensor systems, pages 1 12, 2004.

[59] E. Souto, G. Guimarães, G. Vasconcelos, M. Vieira, N. Rosa, and C. Ferraz. A message-oriented middleware for sensor networks. Proceedings of the 2nd workshop on Middleware for pervasive and ad-hoc computing, pages 127-134, 2004.

[60] T. van Dam and K. Langendoen. An adaptive energy-e cient MAC protocol for wireless sensor networks. Proceedings of the rst international conference on Embedded networked sensor systems, pages 171 180, 2003.

[61] N. Wang, N. Zhang, and M. Wang. Wireless sensors in agriculture and food industry. Recent development and future perspective. Computers and Electronics in Agriculture, 50(1):1 14, 2006.

[62] WASP project IST-034963. State of the Art. 2007.

[63] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh. Fidelity and yield in a volcano monitoring sensor network. Proceedings of the Sev- enth USENIX Symposium on Operating Systems Design and Implementa- tion (OSDI 06), Seattle, USA, 2006.

[64] Hao Wu, Mahesh Palekar, Richard Fujimoto, Jaesup Lee, Joonho Ko, Randall Guensler, and Michael Hunter. Vehicular networks in urban transportation systems.: Proceedings of the 2005 national conference on Digital government research, pages 9 10. Digital Government Research Center, 2005.

[65] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for Ad Hoc routing. Proceedings of the 7th annual international conference on Mobile computing and networking, pages 70 84, 2001.

[66] W. Ye, J. Heidemann, and D. Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Transactions on Networking (TON), 12(3):493 506, 2004.

[67] R. Zheng, J.C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing, pages 35-45, 2003.

[68] R. Zheng and R. Kravets. On-demand power management for ad hoc networks. Ad Hoc Networks, 3(1):51-68, 2005.

[69] ZigBee Alliance. http://www.zigbee.org.