UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE TEORÍA DE LA SEÑAL Y COMUNICACIONES

INGENIERÍA DE TELECOMUNICACIÓN
ESPECIALIDAD: SISTEMAS Y REDES DE TELECOMUNICACIONES

PROYECTO FINAL DE CARRERA

ANÁLISIS DE PROTOCOLOS DE ENRUTAMIENTO PARA REDES DE SENSORES INALÁMBRICAS

AUTOR: ANTONIO MANUEL PALMA GÓMEZ
TUTOR: ÁNGEL MARÍA BRAVO SANTOS

18 de Diciembre de 2009

Título: *ANÁLISIS DE PROTOCOLOS DE ENRUTAMIENTO PARA REDES DE SENSORES INALÁMBRICAS.*

Autor: *ANTONIO MANUEL PALMA GÓMEZ*

Tutor: *ÁNGEL MARÍA BRAVO SANTOS*

La defensa del presente Proyecto Fin de Carrera se realizó el día 18 de Diciembre de 2009; siendo calificada por el siguiente tribunal:

Presidente: *Antonio Artés Rodríguez*

Secretario *David Luengo García*

Vocal *Ignacio Soto Campos*

Habiendo obtenido la siguiente calificación:

Calificación:

Presidente **Secretario** **Vocal**

Agradecimientos

Quiero dar mi más sincero agradecimiento a Don Ángel María Bravo, por haberme dirigido este proyecto de fin de carrera. Por todas las atenciones, por el tiempo que ha perdido conmigo, y sobre todo por su apoyo.

En general, a todos aquellos profesores y alumnos que a lo largo de la carrera me han ayudado a formarme tanto humanamente como técnicamente, que aunque no les mencione de forma explícita, no les puedo negar un sincero agradecimiento.

No puedo dejar de agradecer la comprensión de mis familiares, en especial de mis padres, y amigos, puesto que ellos han sido los que más me han tenido que aguantar en mis malos momentos, enfados y demás sucesos que me han acaecido a lo largo de la elaboración de este proyecto.

También agradecer de forma muy especial a Irene, por estar ahí, por ser mi cómplice, por entender de mí lo que otros no entienden y darme aliento cuando a veces me ahogaba.

Gracias porque todos a vuestra manera habéis sido importantes, y porque en mí y en todo esto hay un trozo de cada uno de vosotros.

Porque semos asina, semos pardos,
del coló de la tierra,
los nietos de los machos, que otros días
triunfaron en América.
Luis Chamizo Trigueros

Verán, el cable telégrafo, es un tipo de gato muy muy largo. Uno tira de su cola en New York, y su cabeza maullando aparece en Los Ángeles. ¿Entienden esto? Pues la radio trabaja de la misma forma, uno manda señales aquí y las reciben allá. La única diferencia es que no hay gato.
Albert Einstein

Resumen

Los recientes desarrollos en la integración de componentes electrónicos han permitido la introducción de una nueva tendencia en el campo de la telemetría y de las comunicaciones: las redes de sensores inalámbricas. Mediante esta nueva tecnología se obtienen los beneficios que proporcionan las redes de sensores junto a una mayor libertad a la hora de situar los sensores en las proximidades del objetivo.

En las redes de sensores inalámbricas, el encaminamiento es un aspecto crítico y de importancia. Este documento realiza un estudio de este tipo de redes y de los protocolos de encaminamiento que pueden ser usados en ellas. Además, ofrece una maqueta de simulación en la cual se puede caracterizar el funcionamiento de cualquier protocolo de encaminamiento que se utilice en una red de sensores inalámbricos ad-hoc. Dicha maqueta ofrece una serie de ventajas frente a los simuladores existentes en el mercado, siendo la más importante la facilidad de implementar cualquier protocolo en ella.

El presente proyecto trata de caracterizar el protocolo *AODV* haciendo uso del entorno de simulación implementado. Este protocolo es utilizado por el estándar Zigbee, siendo dicho estándar el más usado en el ámbito de las redes de sensores inalámbricas. En las simulaciones del protocolo *AODV* se muestran una serie de métricas que pueden ser usadas para las comparaciones de los distintos protocolos implementados en el entorno de simulación.

El objetivo fundamental de este proyecto será ofrecer un entorno de simulación adecuado para caracterizar el funcionamiento de los protocolos de encaminamiento en redes inalámbricas de sensores ah-hoc, facilitando la implementación de estos protocolos mediante el uso de dicho entorno. En el estudio del estado del arte de los protocolos de encaminamiento se propone un gran número de éstos que pueden ser implementados en la maqueta de simulación diseñada.

Abstract

Recent developments in the integration of electronic components have enabled the introduction of a new trend in the field of telemetry and communications: wireless sensor networks. By means of this new technology it is possible to get the benefits that sensor networks provide and a greater freedom for locating the sensors near to the target.

In wireless sensor networks, routing is critical and important. This paper makes a study of this type of networks and routing protocols that can be used in them. In addition, it offers a simulation environment which can characterize the performance of any routing protocol that is used in a wireless sensor network ad-hoc. This simulation environment offers a number of advantages over existing simulators on the market, the most important being the ease of implementing any protocol in it.

This project try characterize the protocol (AODV) using the simulation environment implemented. This protocol is used by the Zigbee standard, being the most used standard in the field of wireless sensor networks. In the simulations of AODV protocol are shown a series of metrics that can be used to comparisons of the different protocols implemented in the simulation environment.

The primary objective of this project will be provide a simulation environment to characterize the operation of routing protocols in wireless sensor networks ah-hoc, facilitating the implementation of these protocols through the use of that environment. In the study of the state of the art of routing protocols are proposed a great number of protocols that can be implemented in the simulation model designed.

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1

INTRODUCCIÓN

1.1. Motivación del proyecto

En septiembre de 1999, la revista *Business Week* predecía que la tecnología en redes de sensores iba a ser una de las tecnologías más importantes para el siglo XXI [82]. En los años 90, las redes revolucionaron la forma en la que las personas y las organizaciones intercambiaban información y coordinaban sus actividades. En el mundo actual, la visión futurista que Mark Weiser describió en su artículo *The Computer for the 21st Century* [30] comienza a ser una realidad. Weiser describe entornos saturados de elementos con capacidades de cómputo y comunicación, totalmente integrados en las vidas de las personas y que proporcionan información asociada a las necesidades y al entorno en el que se pueden encontrar en cada momento.

Los avances recientes en la electrónica y tecnología inalámbrica han permitido que la fabricación de nodos multifuncionales de sensores, de bajo coste y baja potencia, que son pequeños en tamaño y se comunican en distancias cortas, sea una alternativa viable técnica y económicamente [93]. Estos sensores miden las condiciones ambientales en el medio que les rodea y, a continuación, transforman estas medidas en señales que pueden procesarse para revelar algunas características acerca de los fenómenos ubicados en el área situada a su alrededor. En muchas aplicaciones que requieren operaciones desatendidas, estos sensores pueden ser conectados en red dando lugar a una red de sensores inalámbricos (WSN), esta red inalámbrica facilita la conexión de muchos periféricos y reduce la cantidad de cables haciendo el montaje de equipos más sencillo y el ambiente de trabajo menos limitante [70]. Actualmente los sensores ya se utilizan en el conjunto de actividades que se realizan diariamente, además, en muchos casos de

manera transparente, es decir, los usuarios no asocian su presencia al funcionamiento de estas acciones que se hacen de forma automática. Esta integración transparente al usuario se presenta interesante, pero su interés aumenta si se tiene en cuenta que las continuas mejoras tecnológicas permitirán reducir notablemente su tamaño a la vez que los harán más económicos, facilitando así un incremento de su presencia en las tareas cotidianas[59].

Entre las múltiples aplicaciones de las redes de sensores cabe citar por ejemplo, aplicaciones militares y civiles como detección de intrusos, monitorización del tiempo, seguridad y tácticas de vigilancia, computación distribuida, detección de condiciones ambientales, control de inventario, gestión de desastres, etc. El despliegue de una red de sensores en estas aplicaciones puede ser en forma aleatoria (por ejemplo, descenso de un avión en una aplicación de gestión de desastres) o manual (por ejemplo, sensores de alarma de incendios en una instalación subterránea o sensores colocados bajo tierra para la agricultura de precisión)[6].

Las redes de sensores representan una significante mejora por encima de los sensores tradicionales. El desarrollo de este tipo de redes plantea nuevos problemas en ingeniería de software y hardware, pero a su vez acerca más a la realidad escenarios que antes se pensaron como ciencia ficción [41].

Dentro de las redes inalámbricas se encuentran las redes ad-hoc, que no requieren ningún tipo de infraestructura fija ni administración centralizada, donde los nodos también pueden realizar funciones de encaminamiento retransmitiendo paquetes para comunicarse con otros dispositivos con los cuales no tienen conexión directa. La principal ventaja de estas redes es la flexibilidad, pero hay que tener en cuenta sus características y problemas puesto que en redes inalámbricas existen enlaces de calidades variables, nodos con movilidad... [24] Así pues, los protocolos de enrutamiento usados deberán ser completamente adaptativos para ser adecuados para el entorno inalámbrico [14]. Algunos de los protocolos desarrollados para redes cableadas no tienen cabida en entornos tan dinámicos como los de las redes ad-hoc. Es por eso que el desarrollo de protocolos y arquitecturas para el diseño de redes ad-hoc, y en concreto de redes de sensores, se ha convertido en un campo de investigación muy importante en los últimos años.

Los estándares inalámbricos más extendidos hoy en día son IEEE 802.15.4 [1], Bluetooth o IEEE 802.11. Cada uno de ellos nació con un objetivo distinto y por lo tanto tiene un rango de aplicación y unas características diferentes. En los estándares recogidos bajo la norma 802.11 se busca la interconexión de computadores con unas necesidades de ancho de banda elevadas, de hasta 54Mbps para el 802.11g. El caso de Bluetooth está orientado hacia la implementación de

redes inalámbricas de dispositivos periféricos con un ancho de banda menor, de hasta 1 Mbps. El problema que tienen estos dos estándares para convertirse en una tecnología adecuada a nivel de redes de sensores es la complejidad del protocolo y la rigidez de la topología de la red, así como el alto consumo de potencia de los transceptores. Por su parte el estándar IEEE 802.15.4 está encaminado hacia el desarrollo de redes inalámbricas de baja velocidad, bajo coste y bajo consumo de potencia (LRWPAN) [59]. Este estándar está orientado a aplicaciones donde la velocidad de transferencia no es muy alta, pero permite que los nodos de la red puedan ser alimentados mediante baterías y puedan funcionar sin ser recargados durante años. Es por lo tanto el estándar que actualmente mejor se adapta a los requisitos impuestos para el desarrollo de redes de sensores [66]. Se ha diseñado para su uso en dispositivos de capacidad de proceso y almacenamiento limitadas, como sensores y actuadores. De acuerdo con sus objetivos de diseño y usos potenciales, el estándar constituye una pieza significativa en el camino hacia las redes ubicuas [35]. Las redes ubicuas son un entorno de TIC, donde los usuarios pueden conectarse en cualquier lugar, en cualquier momento y a cualquier red [77].

El IEEE 802.15.4 soporta las topologías en estrella, anillo, bus, árbol, mallado con conexión total y mallado con conexión parcial (se desarrollarán más detenidamente en la Sección 2.3). La topología de conexión total se espera que sea una elección común de despliegue de red, principalmente debido a la flexibilidad intrínseca, fiabilidad adicional a causa de la redundancia de caminos y bajo consumo de batería dado que un destino puede ser alcanzado con la ayuda de nodos intermedios [67].

En todas las aplicaciones de las WSNs se necesita transmitir información entre nodos. Estas transmisiones pueden efectuarse directamente si los nodos participantes se encuentran suficientemente cerca o utilizando comunicaciones multisalto cuando la distancia es mayor que el radio de cobertura. Los algoritmos o protocolos de encaminamiento en WSNs se encargan de tomar decisiones acerca de qué hacer con los mensajes recibidos cuando el nodo receptor no es el destinatario del mensaje [30]. A estas decisiones se las conoce como decisiones de encaminamiento y son utilizadas para enviar mensajes entre sensores. Un nodo que recibe un mensaje cuyo destinatario no es él mismo se encarga de reenviarlo para que el mensaje llegue a su destinatario original tras uno o más saltos intermedios. La mayor dificultad del proceso reside en decidir qué nodo de entre los que pueden ser alcanzados directamente debe encargarse del mensaje, es decir, qué nodo debe ser el siguiente salto en el camino hacia el destino final del mensaje. Además, se debe tener en cuenta la función limitada de los dispositivos, así como cambios de

topología frecuentes a causa de la incertidumbre en la conectividad radio o el consumo total de la batería [49]. Adicionalmente, mientras que algunos dispositivos pueden permanecer en un lugar fijo, otros pueden cambiar de posición (por ejemplo, en el caso de sensores asociados a personas), incrementando la dinámica de la topología de la red [35]. Los nodos participantes en la comunicación multisalto se conocen por su nombre en inglés como *forwarders* o *relays*. Las comunicaciones entre nodos de una WSN se pueden clasificar en unicast, broadcast y multicast [30].

Dada la posibilidad de que los dispositivos sensores actúen como reencaminadores, esto permite una reducción en la potencia de transmisión y un ahorro de batería en los nodos de la red. El enrutamiento en WSNs es muy difícil debido a las características propias que distinguen a estas redes de otras redes inalámbricas como las redes celulares [6]. En primer lugar, debido al relativo gran número de nodos sensores, no es posible construir un esquema de direccionamiento global para el despliegue de éstos, ya que los gastos de mantenimiento para la identificación de cada uno de los nodos son altos. Además, los nodos sensores que se despliegan de manera ad-hoc necesitan auto-organización, ya que el despliegue ad-hoc de estos nodos requiere que el sistema forme conexiones y haga frente a la consiguiente distribución nodal, especialmente cuando el funcionamiento de las redes de sensores es desatendido. En WSNs, obtener los datos a veces es más importante que conocer los números de identificación de los nodos que enviaron los datos. En segundo lugar, en contraste con las redes de comunicación típicas, casi todas las aplicaciones de redes de sensores requieren un flujo de datos detectados de múltiples fuentes a una estación base particular. Esto, sin embargo, no impide que el flujo de datos sea de otras formas (por ejemplo, multicast o punto a punto). En tercer lugar, los nodos sensores están muy limitados en términos de energía, procesamiento y capacidad de almacenamiento. Por lo tanto, requieren una cuidadosa gestión de los recursos. En cuarto lugar, en la mayoría de escenarios de aplicación, los nodos en WSNs son generalmente estacionarios después del despliegue a excepción de tal vez unos pocos nodos móviles. Los nodos en las redes inalámbricas tradicionales son libres de moverse, lo que da lugar a cambios topológicos imprevisibles y frecuentes. Sin embargo, en algunas aplicaciones, algunos nodos sensores pueden moverse y cambiar su ubicación (aunque con muy baja movilidad). En quinto lugar, las redes de sensores son aplicaciones específicas (por ejemplo, los requisitos de diseño de una red de sensores cambia con la aplicación). Por ejemplo, el problema difícil de la baja latencia de transmisión en vigilancia táctica es diferente del de una tarea periódica de monitorización del tiempo. En sexto lugar, la posición de los nodos sensores

es importante porque la recogida de datos se basa normalmente en la localización de los mismos.

El encaminamiento geográfico [30] resuelve muchos problemas en WSNs, pero al abordar el problema del encaminamiento multicast aún quedan muchos otros pendientes. Es muy fácil, por ejemplo, decidir qué vecino es el que se encuentra más cerca del destino. Sin embargo, si se tienen múltiples destinos la decisión no es tan simple. Al encaminar en modo multicast, cada nodo que participa en el encaminamiento, es decir, en la creación del árbol multicast, tiene que decidir si crea una ramificación o continúa la rama actual. Es necesario incluir en la cabecera del mensaje la lista de destinos a cubrir. Cuando a un nodo le llega un mensaje debe decidir si hay un vecino idóneo para ser el siguiente *relay* de todos los destinos o si algunos destinos deben ser cubiertos por un *relay* mientras que otros necesitan ser cubiertos por otro *relay* distinto, produciéndose una ramificación. Esta decisión es posiblemente la más difícil de tomar ya que de lo buena o mala que sea depende totalmente la eficacia del árbol creado. Un árbol en el que la ramificación se produce muy cerca de la raíz ahorra muy pocos recursos, ya que los caminos hacia los distintos destinos se comparten por muy pocos nodos. Por otro lado, un árbol en el que las bifurcaciones se produzcan muy cerca de las hojas, genera caminos excesivamente largos. Además hay que tener en cuenta que la creación de este árbol se hace de manera distribuida, es decir, en principio, los distintos nodos participantes del encaminamiento no saben qué decisiones están tomando los demás nodos. Cada uno se encarga de tomar la mejor decisión posible para el conjunto de destinos al que va dirigido el mensaje que está procesando.

Tradicionalmente, los protocolos de encaminamiento se han centrado en encontrar caminos basados en un coste mínimo de saltos, incluso en el caso de cambios en la topología, con un impacto mínimo en la latencia de entrega de datos, ancho de banda disponible, consumo de batería y memoria para cualquier patrón de tráfico. Pero dadas las características del medio radio, un camino integrado por un mínimo número de saltos no tiene por qué ser el ideal en estos entornos [14]. Parece lógico pensar que otras variables, como la calidad del enlace, podrían optimizar y mejorar los protocolos de encaminamiento en estas situaciones. Además, el compromiso entre funcionalidad del protocolo y complejidad debe ser tenido en cuenta [35]. Siguiendo estas directrices, se han propuesto nuevos algoritmos para el problema de enrutamiento en WSNs. Estos mecanismos de encaminamiento han tenido en cuenta las características propias de las WSNs, junto con los requisitos de aplicación y arquitectura. La tarea de encontrar y mantener las rutas en WSNs no es trivial ya que las restricciones de energía y los cambios repentinos del estado de los nodos (por ejemplo, fallos) causan cambios topológicos frecuentes e imprevisibles. Para reducir

al mínimo el consumo de energía, las técnicas de enrutamiento propuestas en la literatura para WSNs emplean algunas tácticas de enrutamiento conocidas, así como tácticas especiales para WSNs, como la agregación de datos y procesamiento dentro de la red, agrupamiento, diferente asignación de tareas en los nodos y métodos centrados en datos.

Según un estudio realizado por *On World* [37], el mercado de redes de sensores inalámbricos experimenta un alto crecimiento en la demanda, porque cada vez más empresas quieren instalar este tipo de red. El estudio de On World llamado *Wireless Sensor Networks: Growing Markets, Accelerating Demand* [37] está basado en los resultados de una encuesta realizada con 147 empresas. De las empresas sondeadas para el estudio, 29 por ciento utilizaban WSN mientras que más de 40 por ciento consideraban muy probable la posibilidad de que realizasen pruebas piloto con redes de sensores inalámbricos en un futuro cercano.

Debido a que en la actualidad los simuladores existentes de redes de sensores inalámbricos son bastante ineficientes en cuanto al tiempo de ejecución, y los lenguajes de programación usados en dichos simuladores tienen una complejidad alta para el diseño de la red, se propone implementar un entorno de simulación en el cual se puede determinar el comportamiento de un algoritmo de encaminamiento concreto sobre una red de sensores inalámbricos ah-hoc dispuesta de manera aleatoria o manual en el emplazamiento. Dicho entorno de simulación permitirá, mediante la implementación de un número reducido de funciones, obtener una serie de parámetros relacionados con el funcionamiento del protocolo en la red.

1.2. Objetivos

El objetivo de este proyecto fin de carrera es la creación de un entorno de simulación apropiado para evaluar y comparar el desempeño de los mecanismos de enrutamiento para redes inalámbricas de sensores. Además, se simulará el algoritmo de encaminamiento *AODV* (*Ad hoc On Demand Distance Vector*) propio de las redes de sensores inalámbricos ad-hoc. En base a la simulación, se obtendrán datos de la tasa efectiva de transporte de información, la batería media y el retardo medio en la red cuando sobre dicha red trabaja el protocolo anteriormente citado.

La red usada será de tipo ad-hoc en la que todos los nodos sensores tienen las mismas funciones, pudiendo encaminar la información para comunicarse con otros nodos con los que no tienen conexión directa. En dicha red se definirá cuál o cuáles de los nodos que la componen

actuarán como nodos sumideros, es decir, nodos encargados de realizar el procesamiento de los datos sensados en dicha red.

Los objetivos específicos que se persiguen con el desarrollo de este sistema son los siguientes:

- Hacer un estudio de los algoritmos de enrutamiento existentes para redes inalámbricas de sensores.
- Caracterizar los problemas de enrutamiento presentados en redes inalámbricas de sensores.
- Implementar un entorno de simulación que permita el análisis de cualquier protocolo de enrutamiento sobre redes de sensores inalámbricos.
- Seleccionar un protocolo de enrutamiento del estudio realizado para poder simular su comportamiento en el entorno implementado.
- Realizar todas las implementaciones en código libre para su posterior uso, estudio y modificación. Además, se ha tratado de cuidar la eficiencia de las simulaciones en cuanto a tiempo de ejecución y memoria utilizada.

1.3. Contenido de la memoria

Todo el trabajo realizado se ha documentado a lo largo de los capítulos y en los anexos.

En el primer capítulo se trata de forma general el gran desarrollo de las redes de sensores, mencionando algunos problemas que existen en este tipo de redes. Además, se explica el por qué usar técnicas de enrutamiento en este tipo de redes. Todo esto se realiza para justificar la realización del proyecto en cuestión. Los objetivos del proyecto y el contenido de la memoria se citan en este capítulo.

El segundo capítulo está dedicado a explicar más detalladamente las redes de sensores. En este capítulo se habla de sus principios, clasificación, problemática y estandarización.

El tercer capítulo se centra en la caracterización del enrutado en las redes de sensores de forma teórica. Se tratará los fundamentos y estrategias de enrutamiento en las redes de sensores y la problemática existente dichas estrategias.

El cuarto capítulo citará la implementación del entorno de simulación que permite caracterizar el comportamiento de cualquier protocolo de encaminamiento para redes inalámbricas de

sensores. Además, se describirá el uso de los ficheros utilizados por dicho entorno y las simulaciones realizadas en dicha maqueta.

El quinto capítulo discute las características principales del algoritmo de encaminamiento *AODV* y cómo implementar un protocolo de encaminamiento, poniendo como ejemplo el protocolo citado anteriormente, haciendo uso de la maqueta de simulación diseñada. Se mostrarán algunas simulaciones realizadas del protocolo implementado.

Tras el capítulo 5 se encuentran las conclusiones del proyecto y las lineas futuras.

En la parte final de la memoria se pueden consultar los anexos. El primer anexo que se encuentra es el presupuesto del proyecto en cuestión.

Capítulo 2

REDES DE SENSORES

2.1. Definición

Una red de sensores está formada por elementos sensores autónomos (llamados nodos, motas o pods) distribuidos en el espacio que se comunican entre sí de manera inalámbrica o por cableado, recogiendo información de sus sensores para controlar diversas condiciones en distintos puntos, entre ellas la temperatura, el sonido, la vibración, la presión y movimiento o los contaminantes [3]. Típicamente envían la información recogida a un elemento sumidero de la red que actúa de pasarela, trasladando la información de la red al usuario final. Normalmente una red de sensores requiere de multitud de nodos sensores para tomar medidas, con lo cual es necesario que el precio de los mismos sea bajo para el éxito de esta tecnología.

En los últimos años, las redes de sensores han estado formadas por un pequeño número de nodos que estaban conectados por cable a una estación central de procesado de datos [82]. Hoy en día, sin embargo, se centran más en redes de sensores distribuidas e inalámbricas, ya que cuando la localización de un fenómeno físico es desconocida, este modelo permite que los sensores estén mucho más cerca del evento de lo que estaría un único sensor. En la Figura 2.1 se puede observar la complejidad de una red de sensores inalámbricos, que generalmente consiste en una red de adquisición de datos y una red de distribución de datos, monitorizada y controlada por un centro de control.

Las redes de sensores sin hilos son un caso particular de redes mesh WMNs multisalto (Wireless Mesh Networks). En las redes WMN los nodos están conectados mediante enlaces sin hilos y actúan como encaminadores de paquetes de otros nodos que no están en el rango de

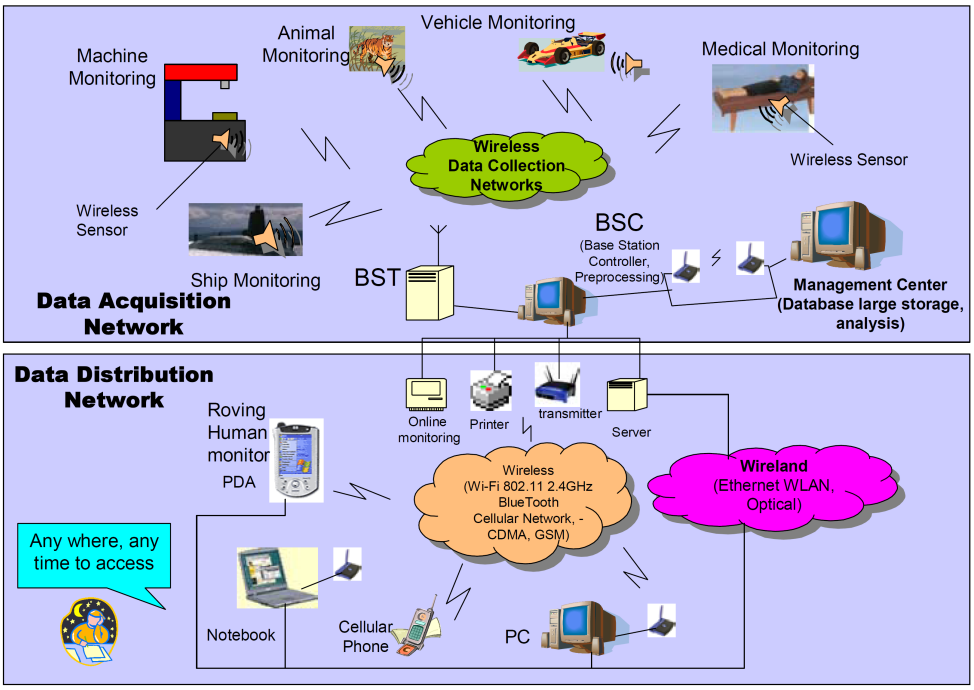


Figura 2.1: *Red de sensores genérica.*

conexión directa con sus destinos [45]. Dentro de las redes mesh se puede diferenciar entre redes con y sin infraestructura. En las redes mesh con infraestructura existen nodos que desempeñan tareas específicas y que se les presupone ciertas prestaciones. Por otro lado, en las redes sin infraestructura los nodos móviles tienen las mismas prestaciones y se conectan entre sí de manera arbitraria cooperando para formar una red. Éstas son las denominadas redes ad-hoc o MANET (Mobile Ad-hoc Network) [14]. Las redes de sensores pueden formar parte tanto de las redes mesh con infraestructura como de las ad-hoc.

Las áreas que hacen de las redes de sensores una categoría distinta dentro de las redes inalámbricas son:

- Movilidad de los nodos: la movilidad no es un mandato que se deba cumplir. Por ejemplo, el despliegue de nodos para monitorizar las propiedades del suelo no lo requiere.

- Tamaño de la red: el número de sensores en la red puede ser mucho más grande que una red inalámbrica típica.

- Densidad de la red: varía de acuerdo al tipo de aplicación. En el caso de aplicaciones de seguridad se requiere que la red siempre esté disponible y con redundancia de información.

- Limitación de energía: en el caso de las redes de sensores, se espera que funcionen en ambientes agresivos en cuanto a condiciones ambientales, con el mínimo o nula supervisión humana posible. Emplear este tipo de redes, donde la única fuente de alimentación para el nodo sensor es la batería, limita la vida útil de la red, exigiendo de esta manera un conjunto de protocolos de red muy eficientes a nivel capa de red, capa de enlace de datos y hasta física para brindar un control óptimo de energía [43]. Las fuentes de alimentación en redes de sensores se pueden clasificar en recargables, no recargables y regenerativas.

- Fusión de los datos e información: las limitaciones de ancho de banda y energía demandan el aumento de bits y de información en los nodos intermedios. Para la fusión de los datos, se necesita la combinación de múltiples paquetes dentro de uno solo antes de su transmisión. Lo que se busca es reducir el ancho de banda utilizado mediante encabezados redundantes en los paquetes y minimizar el retardo al acceder al medio para transmitir los múltiples paquetes.

- Distribución del tráfico: el patrón de tráfico varía en base al tipo de aplicación de la red. El sensado de un factor ambiental, genera de manera periódica pequeños paquetes con datos que indican el estado del parámetro de estudio a una estación central de monitorización. Esto demanda un bajo ancho de banda. Sin embargo, cuando se trata de detectar a un intruso en el ámbito militar, se genera un tráfico de detección en eventos con limitaciones en la transmisión en tiempo real [57].

Habitualmente una red de sensores está formada por varios de los siguientes elementos [63]:

- Nodo sensor: el elemento más común. Es el encargado de recoger la información que se quiere medir y transmitirla. Habitualmente está formado por un subsistema de comunicación (transceptor RF) que es el responsable del intercambio de mensajes con los nodos sensores vecinos, un subsistema de procesado (microcontrolador y memoria) que lleva acabo el cómputo de la información recabada por el sensado, un subsistema del sensor que son un número de sensores que sensan (mide un parámetro) el medio ambiente y una batería.

- Gateway (pasarela): es el dispositivo encargado de enviar la información recopilada al usuario final. Habitualmente está conectado a una red de energía y a Internet mediante conexión de banda ancha.

En algunos protocolos e implementaciones concretas de redes de sensores hacen uso de otros elementos adicionales o modificaciones en los ya comentados.

2.2. Descripción del hardware de un nodo sensor

Como se ha citado en el apartado anterior un nodo sensor es un elemento computacional con capacidad de procesamiento, memoria, interfaz de comunicación y con un conjunto de sensores, como se muestra en la Figura 2.2. El hardware básico de un nodo sensor se compone de un transceptor (transmisor/receptor), un procesador, uno o más sensores, memoria y batería. Los componentes permiten la comunicación (enviar/recibir información) y ejecutar tareas que requieren procesamiento además de efectuar funciones de sensado.

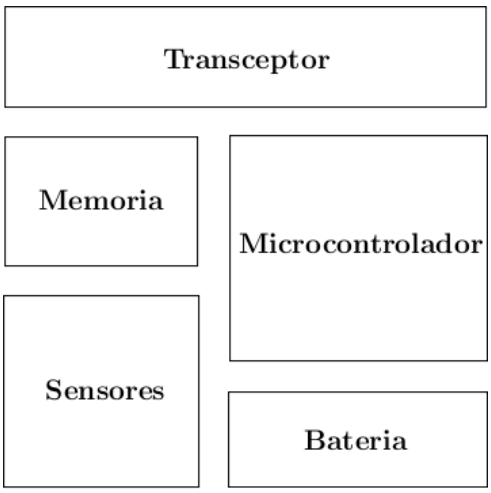


Figura 2.2: *Componentes hardware de un nodo sensor [3].*

La capacidad de procesamiento depende del tipo de microprocesador que se emplee. Así mismo, posee una memoria interna en el microcontrolador. La comunicación se realiza mediante un transceptor (transmisor/receptor). Además, la fuente de alimentación varía dependiendo el tipo de tecnología con la cual la batería esté fabricada. En cuanto al sensor, éste es el responsable de monitorizar el parámetro de interés e informar del mismo. Estas motas o partículas, son pequeños dispositivos inalámbricos basados en tecnología MEMS, que detectan factores físicos [4].

Cada nodo sensor puede ser equipado con dispositivos sensores [4] acústicos, sísmicos, infrarrojos, vídeo cámaras, mecánicos, de calor, temperatura, radiación, entre otros. La Figura

2.3 muestra algunos tipos de sensores que pueden presentarse en un nodo sensor tales como un micrófono (Figura 2.3 (a)) que puede ser usado para captar señales del medio ambiente, un detector de caudal (Figura 2.3 (b)) que es capaz de medir el flujo o corriente de medios líquidos, un diodo detector de luz (Figura 2.3 (c)), un acelerómetro (Figura 2.3 (d)) que es un sensor capaz de medir la aceleración de un objeto, un sensor de humedad (Figura 2.3 (e)), un sensor de radiación (Figura 2.3 (f)) y un sensor de luz ultravioleta (Figura 2.3 (g)) [100].

(a) Micrófono (b) Detector de caudal (c) Detector de luz (d) Acelerómetro

(e) Detector de humedad (f) Detector de radiación (g) Detector de UV

Figura 2.3: *Ejemplos de sensores*

Como se denota en [58], la tendencia es producir sensores a gran escala, a precios bajos, con mejor capacidad de cómputo y de tamaño reducido como se observa en la Figura 2.4.

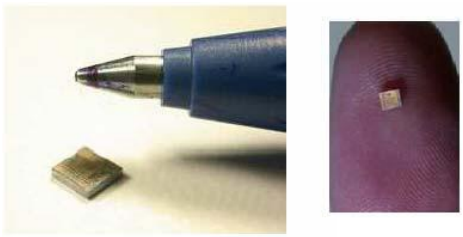


Figura 2.4: *Tamaño de los sensores.*

Es elemental resaltar que estos nodos sensores tienen fuertes restricciones en cuanto a la

capacidad de memoria, de procesamiento y principalmente de energía, siendo deseable poseer dispositivos de bajo consumo de energía [100]. Los problemas de las redes de sensores serán tratados en la Sección 2.4.

Entre los fabricantes de nodos sensores se pueden citar [82]:

- CROSSBOW: 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 se encuentran las plataformas Mica, Mica2, Micaz, Mica2dot, telos, telosb, Iris e Imote2.

- SENTILLA: llamada anteriormente MoteIV. Es la encargada de las motas 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.

- SHOCKFISH: empresa suiza que desarrolla TinyNode. A partir de este tipo de mota en Laussane han implementado una red de sensores en todo el campus de la universidad *Ecole Polytechnique Fédérale de Lausanne*.

- BTnode: los módulos fabricados por esta empresa han sido desarrollados por el ETH Zurich, conjuntamente por el *Computer Engineering and Networks Laboratory* (TIK) y el *Research Group for Distributed Systems*. 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.

- 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. 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.

- SUN: Sun SPOT (*Sun Small Programmable Object Technology*) es una mota para WSN desarrollado por Sun Microsystems. El aparato está construido bajo la normativa estándar IEEE 802.15.4. Al contrario de otros nodos inalámbricos, el Sun SPOT está construido bajo la maquina virtual Java Squawk.

- 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.

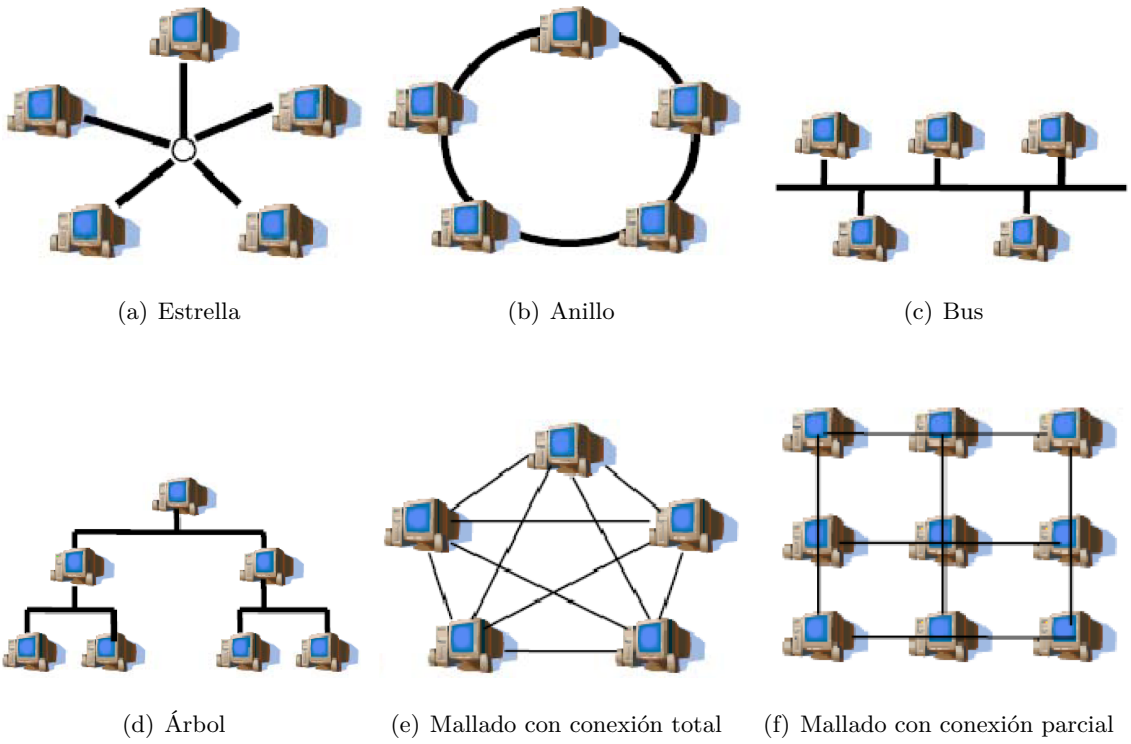
2.3. Topología

Existen diferentes posibilidades a la hora de interconectar los nodos de una red de sensores entre sí como se puede ver en la Figura 2.5. Se puede hablar de las siguientes [63]:

- Estrella: es la topología más básica. Los nodos de los extremos pueden ser muy simples y se conectan directamente al sumidero o a la pasarela.
- Anillo: se trata de una topología típica de redes de fibra óptica, poco práctica en redes de sensores debido a la distribución que tendrán los nodos en el espacio.
- Bus: esta topología tiene sentido cuando se conectan los nodos con un medio guiado, pero no es el caso de las redes de sensores.
- Árbol: una topología interesante en ciertos casos, donde la información se lleva de los nodos sensores más alejados de la pasarela hasta el sumidero. Requiere mayor complejidad que una red en estrella, pero mantiene la simplicidad en los nodos de las hojas.
- Mallado con conexión total: existe enlace directo entre todos los pares de nodos de la red. Una malla completa con n nodos requiere de $n(n-1)/2$ enlaces directos. Debido a esta característica, es una tecnología costosa pero muy confiable.
- Mallado con conexión parcial: algunos nodos están organizados en una malla completa, mientras otros se conectan solamente a uno o dos nodos de la red. Esta topología es menos costosa que la malla completa pero por supuesto, no es tan confiable ya que el número de enlaces redundantes se reduce.

2.4. Problemas característicos

Las características específicas de las redes de sensores traen consigo retos a superar, como son la gestión de la energía, los protocolos utilizados, la fiabilidad de la red, el tamaño de los nodos sensores y la seguridad en las redes de sensores [45].



(a) Estrella (b) Anillo (c) Bus

(d) Árbol (e) Mallado con conexión total (f) Mallado con conexión parcial

Figura 2.5: *Topologías de red*

En este sentido, es muy interesante la encuesta de *OnWorld* [37] (citada en el capítulo anterior) a potenciales clientes (en su mayoría fábricas e industrias) en la cual se les preguntaban por los aspectos más importantes a mejorar en las redes de sensores como se puede apreciar en la Figura 2.6. Es decir, qué aspectos les parecían que impedían su implantación en el mercado.

El principal problema de las redes de sensores es su fiabilidad. Los investigadores, en general, se han centrado en hacer los nodos más pequeños (como el proyecto *Smart Dust*) cuando es lo menos importante. Los estándares son el siguiente problema, como es razonable, debido a la gran variedad y dificultad de elegir uno por encima de los demás [63]. Después viene la facilidad de uso, una crítica posiblemente a ZigBee, que al igual que Bluetooh es engorroso de programar. Y el 50 % hacía hincapié en la gestión del consumo energético. Por supuesto, en aplicaciones más exigentes este aspecto habría tenido más importancia.

2.4.1. Fiabilidad

Es habitual que el lugar donde estén situados los nodos sensores no sea el más adecuado desde el punto de vista de la transmisión de la información. Ya sea en una fábrica, en una

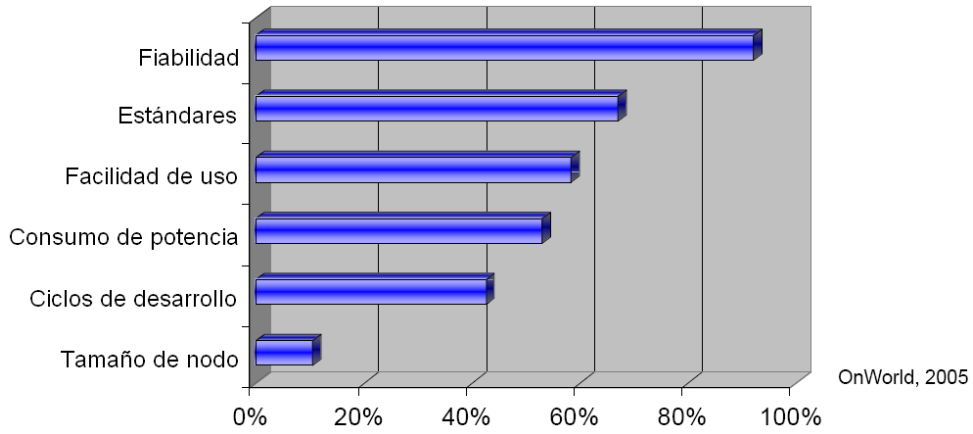


Figura 2.6: *Encuesta OnWorld.*

oficina o en el campo, es seguro que se tienen interferencias en las comunicaciones y que los nodos pueden dejar de funcionar por causas ajenas a su diseño (son robados, se estropean, se introducen obstáculos entre los nodos, etc). Es por ello que la fiabilidad de las comunicaciones es fundamental.

Una manera de mejorar la fiabilidad de la red de sensores es mediante la topología. Se ha comprobado que una topología en estrella o similar es débil porque confía plenamente en que la información va a llegar al nodo central de la estrella. Si ese nodo cae, también se pierden todas sus ramas. Es por eso que la topología que mejor funciona a la hora de la verdad es la mallada, donde todos los nodos tienen varios nodos a los que pueden comunicarse, de manera que la información siempre encontrará una manera de llegar a su destino. Esta topología tiene capacidad de autorrestauración [3], es decir, si se avería un nodo, la red encontrará nuevas vías para encaminar los paquetes de datos. De esta forma, la red sobrevivirá en su conjunto, aunque haya nodos individuales que pierdan potencia o se destruyan. Además, la congestión del tráfico pasa rápidamente a ser un nuevo problema en redes troncales.

El otro punto clave a la hora de la fiabilidad es la utilización de saltos en frecuencia. Los canales que utilizan las redes de sensores son sometidos habitualmente a fuertes interferencias ya que son de libre utilización [28]. En la Figura 2.7 se puede observar los efectos de las interferencias en un caso real, donde se tomaron medidas entre dos nodos sensores a lo largo de 30 días [63].

Como se puede ver, es importante realizar saltos en frecuencia para cambiar de un canal a otro dinámicamente conforme se tienen interferencias si se quiere mantener la fiabilidad de la red.

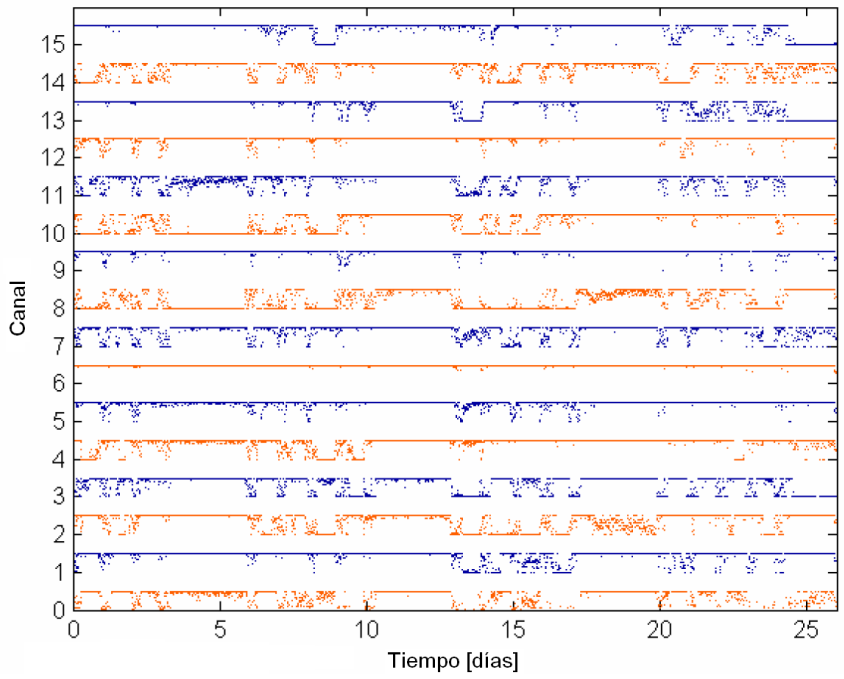


Figura 2.7: *Interferencias en los canales 2.4GHz ISM [28].*

2.4.2. Conservación de la energía

Los nodos sensores han de funcionar con su propia fuente de energía, ya sea una batería, o mediante un método de obtención de energía, como placas solares. Debido a que es necesario que el periodo de vida de la fuente de energía sea largo, típicamente superior al año, es importante hacer un uso muy ajustado de la energía en el dispositivo [3]. Una importante característica de las redes de sensores inalámbricos es reducir al mínimo el consumo de energía de los nodos, proporcionando al mismo tiempo el mayor rendimiento posible a los usuarios del sistema.

Diseñar los nodos para un bajo consumo supone elegir componentes de baja potencia, algo que a primera vista puede parecer trivial, pero que suele ser más complejo de lo que parece. El primer parámetro a considerar es el consumo de energía de la CPU, el sensor, el radiotransceptor y, posiblemente, de otros elementos, como la memoria externa y los periféricos durante el modo normal de operación. La elección de elementos de baja potencia implica normalmente aceptar compromisos sobre el rendimiento medio. Por regla general, una CPU de baja potencia opera en un ciclo reducido de reloj, con menos características en el chip que otras unidades homólogas que consumen más energía. La solución está en elegir elementos con el rendimiento justo para poder hacer el trabajo.

La manera más común de afrontar el problema es el ciclo de trabajo de los componentes. Dado que los nodos no necesitan enviar información continuamente, se establece un ciclo de trabajo para el dispositivo y el resto del tiempo está «dormido», consumiendo muy poca energía, del orden de μA. El ciclo de trabajo se sitúa en torno al 1 %, pudiendo ser tan pequeño hoy en día como un 0,01 % [63]. A menudo se puede incluso desconectar por completo la alimentación del sensor y del transceptor. Sin embargo, la CPU necesitará alguna alimentación en modo dormido para poder reactivarse.

Otro aspecto que también se suele pasar por alto es el tiempo de activación y desactivación de los elementos. Por ejemplo, el transceptor necesitará un cierto tiempo mínimo hasta que se estabilicen sus osciladores. Durante la espera, tanto el transceptor como la CPU consumen energía, consumo que es necesario minimizar. Lo mismo ocurre, como es lógico, al activar la CPU y el sensor.

Además del ciclo de trabajo, hay que tener una serie de protocolos que sean conscientes de este problema y tengan en cuenta la eficiencia energética. Es importante la coordinación de los dispositivos para que estén el mínimo tiempo «despiertos». Algunos protocolos son poco eficientes y ninguna programación integrada inteligente ayudará a reducir el consumo hasta un nivel aceptable. Otros protocolos se diseñan para conseguir un bajo consumo sin comprometer indebidamente el rendimiento de la comunicación. Uno de estos protocolos de baja potencia es la plataforma tecnológica de interconexión inalámbrica para sensores y actuadores (WISA, Wireless Interface to Sensors and Actuators) [34]. En este sentido, es mucho más eficaz un protocolo de acceso al medio basado en la coordinación de los nodos que el habitual CSMA. En la Figura 2.8 se puede observar el consumo de un nodo sensor en el que se utiliza un protocolo de red de bajo consumo.

2.4.3. Tamaño de los nodos sensores

En realidad, es el menor de los problemas hoy en día. Son más importantes lo demás en la mayoría de aplicaciones, pero no en todas. Si se quieren utilizar redes de sensores en aplicaciones como adquisición de señales vitales en pacientes, el tamaño es importante.

En general el tamaño del nodo sensor está limitado por el de su fuente de energía: la batería en la mayor parte de los casos. Se puede observar en la Figura 2.9 algunos ejemplos de baterías. Se tienen tres tipos de fuente de alimentación [63]:

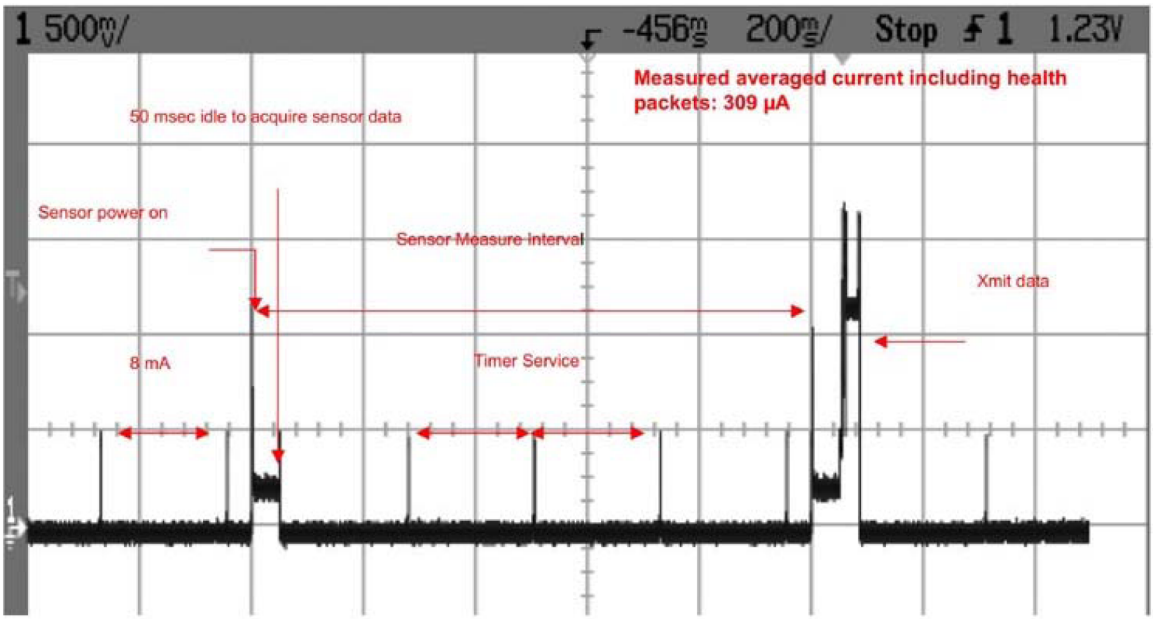
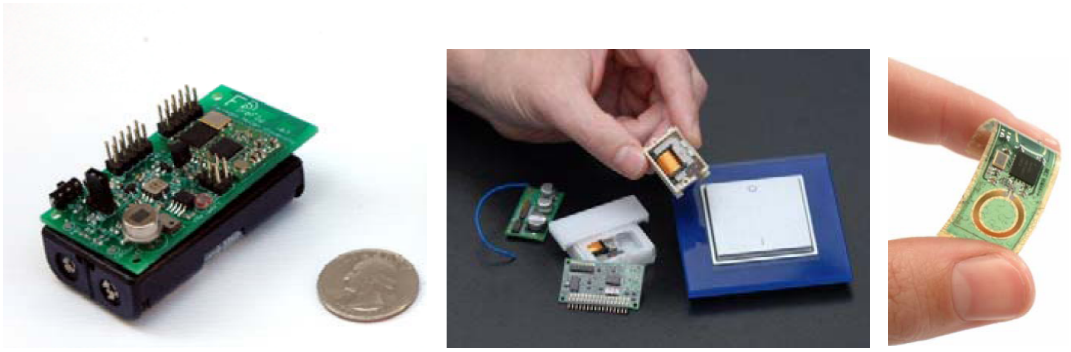


Figura 2.8: *Consumo de un nodo sensor de Crossbow utilizando un protocolo de red de bajo consumo.*

- Baterías convencionales: las más comunes y utilizadas en monitorización de campo, de fábricas, etc. Son las más baratas.
- Baterías especiales: como puedan ser las baterías *Thin Film*, que todavía no han llegado de forma masiva al mercado. Precisamente por ello no están generalizados sino en proceso de investigación.
- Obtención propia de la energía: estos sensores obtienen su propia energía del ambiente. Existen aquellos que la obtienen de la radiación de la luz, de las vibraciones, de la presión o de los cambios de temperatura, por ejemplo. Debido a que la energía que obtienen por este método es muy escasa tienen fuertes limitaciones de diseño.

2.4.4. Seguridad en redes de sensores

La seguridad en las redes de sensores ha sido estudiada de forma extensiva por la comunidad científica, y aunque aún existen problemas de seguridad que deben ser resueltos (p. ej. manejo de nodos móviles, delegación de privilegios, privacidad, agentes seguros, actualización del código [94]), actualmente es posible crear una red de sensores que cumpla un conjunto básico de propiedades de seguridad. Para cumplir con ese conjunto mínimo de propiedades seguras en



(a) Nodo alimentado con baterías AA

(b) Nodo alimentado por presión

(c) Nodo alimentado con baterías Thin Film

Figura 2.9: *Tamaño de las baterías*

sus operaciones internas, las redes de sensores deben utilizar primitivas criptográficas, utilizar sistemas de gestión de claves, y proporcionar soporte para el conocimiento del entorno y la autoconfiguración [8]. Respecto a las primitivas criptográficas, el hardware actual de las redes de sensores es perfectamente capaz de soportar criptografía simétrica, criptografía de clave pública, y funciones hash. El estándar IEEE 802.15.4 proporciona soporte HW para ejecutar la primitiva AES-128, aunque ésta y otras primitivas pueden ejecutarse por SW [50]. Además, existen otros algoritmos de cifrado en bloque y cifrado en flujo como Skipjack [50] y RC4 [21] que, aunque más débiles, tienen unos requerimientos de memoria más asequibles. Los nodos sensores han sido normalmente considerados como dispositivos demasiado restringidos para soportar criptografía de clave pública, pero esta suposición ha cambiado. Utilizando criptografía de curvas elípticas (ECC), es posible tener soporte para cifrar datos (ECIES), firmar y verificar (ECDSA), y negociar claves (ECDH) en un nodo sensor. De todas formas, los requerimientos computacionales y de memoria de estos algoritmos siguen siendo altos [55]. Finalmente, los nodos sensores pueden implementar funciones hash como SHA-1.

Al implementar ECDH en los nodos sensores, es posible resolver el problema de la distribución de claves en una red de sensores. Sin embargo, pueden existir escenarios en los que la funcionalidad de la aplicación sea tan compleja que los nodos sensores no tengan capacidad para implementar las primitivas de clave pública, o incluso que los requerimientos de la aplicación no necesiten de la complejidad de ECDH. La gestión de claves sigue siendo una línea de investigación abierta, aunque con el estado del arte actual es posible satisfacer los requeri-

mientos de redes de sensores pequeñas [7]. La criptografía puede utilizarse como base para crear servicios de seguridad esenciales (confidencialidad, integridad, autenticación), pero estos servicios no son suficientes para cumplir una de las propiedades inherentes a las redes de sensores: la autoconfiguración. Para ser completamente autónomos y autosuficientes, los nodos sensores deben ser capaces de reconocer los eventos que ocurren en su entorno, y que pueden afectar al funcionamiento de la red. Actualmente, existen mecanismos de conocimiento del entorno que pueden detectar eventos anómalos dentro de una red de sensores [79]. Estos mecanismos pueden utilizarse tanto para controlar el estado de la red como para ofrecer soporte a los principales protocolos de una red de sensores: enrutado, agregación y sincronización temporal.

2.5. Protocolo de acceso al medio

Un protocolo de acceso al medio tiene que servir de base para protocolos de más alto nivel. En las redes de sensores por encima del protocolo de acceso al medio está el protocolo de enrutamiento, que usará las funciones implementadas en la MAC (*Media Access Control*) para enviar y recibir paquetes, sincronizar sus operaciones, etc.

Las características esenciales que debe cubrir un protocolo MAC son [82]:

- La flexibilidad, porque el entorno inalámbrico es totalmente cambiante debido a interferencias en el aire de otras ondas, propiedades y formas de los materiales del entorno y un largo etcétera. Además, los nodos pueden fallar en cualquier momento, teniendo que buscar nuevos caminos, reconfigurando la red y recalibrando los parámetros. El tráfico puede incrementarse, ya que la información requerida también puede crecer.

- Eficiencia, un protocolo de MAC debe ser eficiente para poder trabajar en tiempo real, fiable y robusto ante las interferencias y tolerante a los ruidos. Además, debe estar profundamente integrado con el medio donde va a trabajar, siendo a su vez un software barato y con funciones que cubran las necesidades más amplias del mercado.

Estos protocolos afectan directamente a la disipación de la energía, ya que son la capa más próxima al nivel físico, y determinan, en parte, el coste del sistema [66]. También serán clave a la hora de especificar la latencia y el nivel de seguridad del sistema.

En las redes de sensores estos protocolos determinan los canales de radio a utilizar, implementan las transmisiones y recepciones a bajo nivel, además de controlar los errores [75].

Las funciones de un protocolo de MAC son controlar el acceso al medio compartido, que en este caso será un canal de radio (a través del aire). El protocolo debe evitar las interferencias entre transmisiones, mitigando el efecto de las colisiones, mediante retransmisiones, etc.

Se tienen varias aproximaciones:

- Basadas en contención, sin coordinación.
- Basadas en planificación, con un nodo central o punto de acceso, encargado de sincronizar al resto.
- División en frecuencia y división en código, pero no se utilizan prácticamente en las redes inalámbricas de sensores.

Los protocolos de control de acceso al medio han sido estudiados durante décadas, y los diseños varían mucho según el objetivo de la aplicación. Pueden ser clasificados en categorías basadas en diferentes principios; algunos son centralizados, con una estación central como líder del grupo haciendo el control de acceso, otras son distribuidas [75]. Algunas usan un único canal, otras varios. Algunas usan diferentes versiones de acceso aleatorio; otras usan reserva de canal y planificación. Los protocolos están optimizados para diferentes causas: energía, retardo, tasa de transferencia, equidad, calidad de servicio (QoS) o soportar múltiples servicios.

Cada tipo de red necesitará un protocolo diferente. Por ejemplo, las redes donde los eventos se producen de forma periódica necesitan protocolos que usen reserva y planificación del tiempo. Estas redes tendrán una mejor utilización del canal y un mayor tiempo de vida. Por el contrario, para las redes de sensores con eventos asíncronos la MAC ha de ser distribuida y optimizada para la energía. Un método distribuido usando múltiples canales y acceso aleatorio sería lo más indicado para estas redes, ya que se evitaría tener un único punto de fallo. La utilización de múltiples canales reducen las colisiones y retransmisiones, además del retraso, incrementando la tasa de transferencia. El acceso aleatorio evita al nodo conocer la red, en este caso no es necesaria la sincronización.

Existen gran variedad de protocolos, que fueron desarrollados con anterioridad para otros tipos de redes, y que ahora han sido implementados sobre redes inalámbricas de sensores. Además, diversos grupos de trabajo han creado otros protocolos ad-hoc para este tipo específico de redes, lo que ofrece una amplia gama de protocolos, que se enumeran a continuación según su tipo.

Protocolos de Acceso Múltiple por Detección de Portadora (CSMA. *Carrier Sense Multiple Access*) o ranurados:

- Sensor MAC (S-MAC).
- 802.15.4.
- Timeout MAC (T-MAC).
- Berkeley-MAC (B-MAC).
- Power-Aware Multi-access protocol with signaling (PAMAS).

Protocolos de Acceso multiplexado por división de tiempo (TDMA. *Time Division Multiplex Access*):

- Traffic-Adaptive MAC (TRAMA).
- Low-Energy Adaptive Clustering Heirarchy (LEACH).
- Power Aware Clustered TDMA (PACT).
- Bit-Map Assisted (BMA).
- Proposed Gateway MAC (G-MAC).
- SPRIME (SupSlot Period Reservation & Inter-Master Estimation).
- SMACS (Self-Organized MAC for Sensor networks).

Otros:

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

2.6. Sistemas operativos para nodos sensores

Para la programación de nodos sensores existen diversos sistemas operativos, enumerándose los más conocidos a continuación [60]:

- Bertha (pushpin computing platform): plataforma de software diseñada e implementada para modelar, testear y desplegar una red de sensores distribuida de muchos nodos idénticos. Sus principales funciones se dividen en los siguientes subsistemas:
  - Administración de procesos.
  - Manejo de las estructuras de datos.
  - Organización de los vecinos.
  - Interfaz de red.
- Nut/OS: pequeño sistema operativo para aplicaciones en tiempo real, que trabaja con CPUs de 8 bits. Tiene las siguientes funciones:
  - Multihilo.
  - Mecanismos de sincronización.
  - Administración de memoria dinámica.
  - Temporizadores asíncronos.
  - Puertos serie de Entrada/Salida.
- Contiki: sistema operativo de libre distribución para usar en un limitado tipo de computadoras, desde 8 bits a sistemas embebidos en microcontroladores, incluidas motas de redes inalámbricas.
- CORMOS (*Communication Oriented Runtime System for Sensor Networks*): sistema operativo específico para redes de sensores inalámbricas.
- eCos (*embedded Configurable operating system*): sistema operativo gratuito, en tiempo real, diseñado para aplicaciones y sistemas embebidos que sólo necesitan un proceso. Se pueden configurar muchas opciones y puede ser personalizado para cumplir cualquier requisito, ofreciendo la mejor ejecución en tiempo real y minimizando las necesidades hardware.

- EYESOS: se define como un entorno para escritorio basado en Web, permite monitorizar y acceder a un sistema remoto mediante un sencillo buscador.

- MagnetOS: sistema operativo distribuido para redes de sensores en infraestructura o ad-hoc, cuyo objetivo es ejecutar aplicaciones de red que requieran bajo consumo de energía, adaptativas y fáciles de implementar.

- MANTIS (*MultimodAl NeTworks In-situ Sensors*)

- TinyOS: sistema operativo utilizado para *TMote Sky*. Es *event-driven*, es decir, que funciona a partir de eventos producidos que llaman a funciones. Ha sido desarrollado para redes de sensores con recursos limitados. El entorno de desarrollo de TinyOS soporta directamente la programación de diferentes microprocesadores y permite programar cada tipo con un único identificador para diferenciarlo, o lo que es lo mismo se puede compilar en diferentes plataformas cambiando el atributo.

- t-Kernel: sistema operativo que acepta las aplicaciones como imágenes de ejecutables en instrucciones básicas. Por ello, no importará si está escrito en C++ o lenguaje ensamblador.

- LiteOS: sistema operativo desarrollado en principio para calculadoras, pero que ha sido también utilizado para redes de sensores.

2.7. Lenguajes de programación para nodos sensores

La programación de sensores es relativamente compleja, entre otras dificultades está la limitada capacidad de cálculo y la cantidad de recursos. Así como en los sistemas informáticos tradicionales se encuentran entornos de programación prácticos y eficientes para generar y depurar código, incluso en tiempo de ejecución, en estos microcontroladores todavía no hay herramientas comparables.

Se pueden encontrar lenguajes como [60]:

- NesC: lenguaje que se utiliza para motas, y que está directamente relacionado con TinyOS.

- Protothreads: específicamente diseñado para la programación concurrente, provee hilos de dos bytes como base de funcionamiento.

- SNACK: facilita el diseño de componentes para redes de sensores inalámbricas, sobre todo cuando la información o el cálculo a manejar es muy voluminoso, complicado con nesc, este lenguaje hace su programación más eficiente. Por lo tanto es un buen sustituto de nesc para crear librerías de alto nivel a combinar con las aplicaciones más eficientes.

- DCL: lenguaje de composición distribuido (*Distributed Compositional Language*).

- GalsC: diseñado para ser usado en TinyGALS, es un lenguaje programado mediante el modelo orientado a tarea, fácil de depurar, permite concurrencia y es compatible con los módulos nesc de TinyOS.

- SQTL (*Sensor Query and Tasking Language*): como su nombre indica es una herramienta interesante para realizar consultas sobre redes de sensores.

2.8. Software de simulación

El software de simulación existente para redes de sensores inalámbricos es bastante complejo de utilizar e ineficiente en cuanto a tiempo de ejecución. Para implementar un protocolo de encaminamiento en el simulador NS-2, es necesario implementar un módulo con la estrategia de enrutamiento y compilar el código fuente del simulador. Posteriormente, la estrategia es fácil de utilizar, pero la implementación es costosa y complicada. Debido a la complejidad de crear un protocolo en estos simuladores, se ha optado por implementar un entorno de simulación en el cual sea sencillo la caracterización de cualquier protocolo de enrutamiento en redes de sensores ad-hoc.

A continuación se citan algunos simuladores en los que se pueden ver el funcionamiento de redes de sensores:

- NS-2: un simulador de eventos discretos dirigidos a la creación de redes de investigación. Proporciona un apoyo sustancial para la simulación de TCP, enrutamiento y los protocolos multicast en redes cableadas e inalámbricas (locales y por satélite).

- OPNET: con los módulos apropiados, esta plataforma se utiliza para aplicaciones de empresariales de gestión de rendimiento, análisis de configuración de red y capacidad de modelado y planificación.

- OMNeT++: software de código libre, basado en componentes, modular y con un entorno de simulación de arquitectura abierta. Cuenta con una interfaz gráfica robusta y un núcleo de simulación embebido. Su principal área de aplicación es la simulación de redes de comunicación y por su arquitectura flexible y genérica, se ha utilizado con éxito en otras áreas como la simulación de sistemas de tecnología de la información, redes de colas, arquitecturas hardware y procesos de negocio. OMNeT++ se está convirtiendo rápidamente en una plataforma de simulación popular en la comunidad científica, así como en entornos industriales. Varios modelos de simulación de código abierto han sido publicados en el campo de las simulaciones de Internet (IP, IPv6, MPLS, etc), movilidad, ad-hoc y otras áreas.

- CodeBlue: infraestructura de software escalable para los dispositivos médicos inalámbricos. Está diseñado para proporcionar encaminamiento, descubrimiento y seguridad para redes inalámbricas médicas, PDA, PCs y otros dispositivos que pueden utilizarse para vigilar y tratar a los pacientes en el rango médico. También debe operar en una amplia gama de dispositivos inalámbricos, desde sistemas con recursos limitados hasta los más potentes.

2.9. Situación actual. Estándares

Desde 1999 las redes de sensores han sufrido un desarrollo importante. Prácticamente todas las universidades tecnológicas del mundo tienen proyectos dedicados a esta tecnología. Las grandes compañías, como Intel, IBM, Microsoft, Google y Motorola están invirtiendo en su desarrollo [63].

Dado que está comenzando, son muchos los aspectos a mejorar e investigar en el funcionamiento de las redes de sensores. Continuamente salen artículos con nuevas alternativas a soluciones ya ofrecidas anteriormente.

El valor de las redes inalámbricas de sensores se basa en su bajo coste y su distribución en grandes cantidades. Para lograr las economías de escala necesarias para alcanzar un mercado de bajo coste, algunos elementos de las redes inalámbricas de sensores deben ser estandarizados, de manera que los productos de muchos fabricantes puedan interoperar. Esta sinergia añadirá utilidad a las redes inalámbricas de sensores y, por tanto, fomentará su uso.

Con este fin, se están realizando esfuerzos para estandarizar las distintas capas de los protocolos de comunicación de redes de sensores inalámbricos, incluyendo la carga útil de los datos

enviados por sus sensores. El éxito de las redes inalámbricas de sensores como una tecnología se basa en el éxito de estos esfuerzos de estandarización para unificar el mercado, dando lugar a un gran número de dispositivos de bajo coste, interoperables, y evitando la proliferación de la propiedad y de protocolos incompatibles que, aunque tal vez óptimos en sus nichos de mercado, limitan el tamaño del mercado global de sensores inalámbricos.

Además, hoy en día no existe un estándar de facto, ni en protocolos, ni en hardware, ni en representación de datos. Esto es debido por una parte, a que las aplicaciones de las redes de sensores son tan amplias que es difícil englobarlas todas. Y por otra parte, a que las tecnologías están en constante cambio y desarrollo. De entre las soluciones existentes, sin embargo, sí hay quienes tienen más fuerza y quiénes tienen menos. A continuación se presentan las principales iniciativas en curso [63].

2.9.1. IEEE 802.15.4 LOW-RATE WPAN

El ámbito de aplicación del grupo de tareas IEEE 802.15.4, tal como se define en su primera solicitud de autorización del proyecto, es «definir las especificaciones de PHY y MAC para conectividad inalámbrica de baja tasa de datos con dispositivos fijos, portátiles y móviles sin requisitos de consumo de batería que normalmente operan en el Espacio de Operación Personal (POS) de 10 metros». Además, el objetivo del proyecto es «proporcionar una norma de baja complejidad, bajo coste, pequeño consumo de energía y conectividad inalámbrica de baja velocidad de datos entre dispositivos. La tasa de datos en bruto será lo suficientemente alta (un máximo de 200 kbps) para satisfacer un conjunto de necesidades simples, pero adaptable a las necesidades de los sensores y de automatización (10 kbps o menos) para comunicaciones inalámbricas» [64]. Las tasas de datos máxima y mínima se plantearon más tarde a 250 y 20 kb/s, respectivamente.

Este conjunto de objetivos requiere que el estándar IEEE 802.15.4 sea extremadamente flexible. A diferencia de los protocolos que han sido diseñados para una sola aplicación, tales como IEEE 802.11, el estándar IEEE 802.15.4 soporta una variedad casi infinita de posibles aplicaciones en el POS. Estas aplicaciones varían desde requerimiento de alta tasa de transferencia de datos y relativamente baja latencia de mensaje, tales como teclados inalámbricos, ratones, y joysticks, a los que requieren muy baja tasa de transferencia y son capaz de tolerar significante latencia de mensajes, tales como la agricultura inteligente y aplicaciones de detección del entorno. El estándar IEEE 802.15.4 soporta conexiones tanto en estrella como punto a punto,

por lo que es capaz de soportar una amplia variedad de topologías de red y de algoritmos de enrutamiento. Cuando se utiliza seguridad, el paquete de seguridad AES-128 [92] es obligatorio.

El estándar emplea balizas, aunque su uso es opcional. El período de baliza es variable, de modo que se puede hacer para cada aplicación un equilibrio óptimo entre la latencia del mensaje y el consumo de energía de cada nodo de la red. Las balizas pueden ser omitidas para aplicaciones que tienen limitaciones de ciclo de servicio, como puede ocurrir en las redes en la banda de 868 MHz o aplicaciones que requieren nodos de red con recepción constante. El acceso al canal está basado en la contención a través de mecanismo de acceso múltiple por detección de portadora con prevención de colisión (CSMA-CA); la baliza es seguida por un período de acceso de contención (CAP) para los dispositivos que intentan acceder al canal. La duración del CAP es ajustable a una fracción del período entre balizas. Para atender las necesidades de las aplicaciones que requieren mensajes de baja latencia, el estándar admite el uso de ranuras de tiempo garantizadas (GTSs), que reservan el tiempo del canal para dispositivos individuales sin la necesidad de seguir el mecanismo de acceso CSMA-CA.

El estándar tiene un campo de direcciones de 16 bits, sin embargo, la norma incluye también la capacidad de enviar mensajes con direcciones extendidas de 64 bits, lo que permite un número casi ilimitado de dispositivos que se pueden colocar en una única red. La transmisión de mensajes puede ser asentida mediante reconocimientos positivos, tras cada trama transmitida (con la excepción de las balizas y los reconocimientos a sí mismos) se puede recibir un reconocimiento explícito dando lugar a un protocolo fiable. La sobrecarga asociada con los reconocimientos explícitos es aceptable dada la baja tasa de datos típica de redes inalámbricas de sensores. El uso de los reconocimientos es opcional con cada trama transmitida, sin embargo, se soportan el uso de técnicas de reconocimiento pasivas.

El estándar IEEE 802.15.4 incorpora muchas características diseñadas para minimizar el consumo de energía de los nodos de la red. Además de la utilización de períodos largos de baliza y un modo de extensión de vida de la batería, el período activo de un nodo con baliza se puede reducir, permitiendo que el nodo se duerma entre balizas.

La coexistencia de otros servicios que usan bandas sin licencia con los dispositivos IEEE 802.15.4 fue también un factor importante en el diseño del protocolo, y es evidente en muchas de sus características. Por ejemplo, es necesaria la selección dinámica de canales; cuando se puede interferir a otros servicios que se encuentran trabajando en un canal que está siendo utilizado por una red IEEE 802.15.4, el nodo de red que controla la misma (coordinador de red de área

personal (PAN)) explora otros canales disponibles para encontrar un canal más adecuado. En esta exploración, se obtiene una medida de la energía de cada canal alternativo y, a continuación, utiliza esta información para seleccionar un canal adecuado. Este tipo de escaneo también se puede usar antes del establecimiento de una nueva red. Antes de cada una de las transmisiones (excepto tramas baliza o de reconocimiento), cada nodo de red IEEE 802.15.4 debe realizar dos evaluaciones de canal (CCAs) como parte del mecanismo CSMA-CA para garantizar que el canal está desocupado antes de la transmisión.

Un byte de indicación de la calidad del enlace (LQI) se adjunta a cada trama recibida por la capa física antes de que sea enviada a la capa de control de acceso al medio. El nodo receptor utiliza esta información para diversos fines, dependiendo del diseño de la red.

Para maximizar la utilidad de la norma, el grupo de tarea IEEE 802.15.4 tenía que equilibrar el deseo de permitir nodos de red pequeños, de bajo coste y baja potencia con el deseo de producir una norma que se pudiera utilizar por una amplia variedad de aplicaciones en el mercado. El estándar resultante incluye tres tipos de funcionalidades de nodo de red:

- Coordinador PAN: es el nodo que inicia la red y es el principal controlador de la misma. El coordinador PAN puede transmitir balizas y se puede comunicar directamente con cualquier dispositivo en su rango. Dependiendo del diseño de red, es posible que tenga suficiente memoria para almacenar información sobre todos los dispositivos en la red, y debe tener memoria suficiente para almacenar la información de enrutamiento según lo exija el algoritmo empleado por la red.

- Coordinador: puede transmitir balizas y se puede comunicar directamente con cualquier dispositivo a su alcance. Un coordinador puede convertirse en un coordinador PAN en el caso de empezar una nueva red.

- Dispositivo: no genera balizas y sólo puede comunicarse directamente con un coordinador o coordinador PAN.

Estas tres funciones son incorporadas en dos tipos de dispositivos diferentes:

- Dispositivo de función completa (FFD): puede operar en cualquiera de las tres funciones de red (coordinador PAN, coordinador, o dispositivo). Debe tener memoria suficiente para almacenar la información de enrutamiento que imponga el algoritmo empleado por la red.

- Dispositivo de función reducida (RFD): es un dispositivo de muy bajo coste, con requisitos de memoria mínimos. Sólo puede funcionar como un dispositivo de red.

Similar a todos los estándares inalámbricos IEEE 802, el estándar IEEE 802.15.4 normaliza sólo las capas física y de control de acceso al medio (MAC). El estándar IEEE 802.15.4 incorpora dos capas físicas:

- La banda inferior: 868,0-868,6 MHz (para Europa), además de los 902-928MHz (para la mayor parte de América y la Cuenca del Pacífico).
- La banda superior: 2,400-2,485 GHz (en todo el mundo).

El mayor problema de este estándar es que está definido para WPAN y no es específico para redes de sensores [65]. Por lo tanto, aunque abarca una gran cantidad de posibilidades y métodos de comunicación, no resulta eficiente en las aplicaciones más restrictivas y exigentes. Esta dificultad se intenta solventar con el estándar 802.15.4a-2007 [2], que añade nuevas posibilidades a la capa física, como UWB, CSS y DSSS. En la Figura 2.10 se puede observar la pila de protocolos que propone dicho estándar, sobre el que está montado Zigbee.

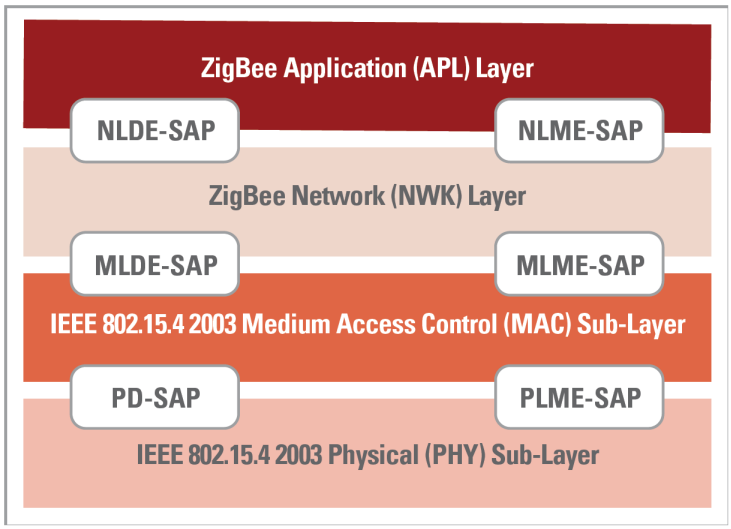


Figura 2.10: *Pila de protocolos 802.15.4 y ZigBee [68].*

2.9.2. Alianza ZigBee

El estándar IEEE 802.15.4 no estandariza las capas más altas del protocolo de comunicación, incluyendo las capas de red y la aplicación. Para asegurar la interoperabilidad entre los disposi-

tivos que funcionen con el estándar IEEE 802.15.4, el comportamiento de estas capas debe estar especificado. La creación de tal especificación se ha incorporado por *ZigBee Alliance* (Alianza ZigBee), un consorcio de la industria de fabricantes de chips, fabricantes OEM, proveedores de servicios y usuarios en el mercado de las redes de sensores inalámbricas, muchos de los cuales trabajaban para desarrollar la norma IEEE 802.15.4. Además de la creación de especificaciones de la capa superior mediante perfiles de aplicación (ver Figura 2.11), ZigBee Alliance es también el apoyo de comercialización y cumplimiento del IEEE 802.15.4, de manera análoga a la relación entre Wi-Fi Alliance, Mountain View, California y el estándar WLAN IEEE 802.11 (comercializado como "Wi-Fi").

Esta especificación inalámbrica es de baja potencia, bajo coste y baja velocidad de transferencia de datos, destinada a electrodomésticos, juguetes, aplicaciones industriales y otras similares [3]. Al igual que ocurre con el estándar 802.15.4-2006, está orientado a WPAN y sufre los mismos defectos.

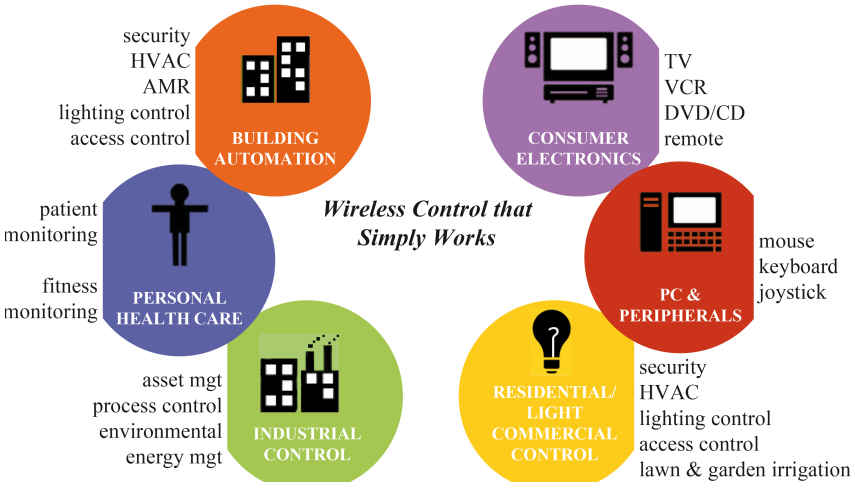


Figura 2.11: *Perfiles de ZigBee.*

Tanto Zigbee como IEEE 802.15.4 han hecho un esfuerzo para adaptarse a las redes de sensores, pero sólo en los últimos años. Es por eso que no han logrado la difusión que esperaban.

Los mayores problemas de ZigBee son su protocolo de acceso al medio, debido a que es poco eficiente en términos de energía, y su topología de red, formada por nodos simples y nodos de enrutamiento [63]. Las redes suelen ser interconexiones de redes en estrella, reduciendo el alcance, la fiabilidad y aumentando notablemente el consumo de energía de los nodos enrutadores al estar siempre despiertos y comprobando si existen nuevos mensajes.

Un ejemplo de esto puede observarse en los cambios que se han hecho en el estándar ZigBee. Comenzó con la especificación ZigBee 1.0 en 2004 y en 2006 sacó una nueva especificación que no era compatible con la anterior. Debería haber sido un problema para sus clientes, pero en realidad no fue así porque apenas había productos desarrollados con ZigBee.

2.9.3. WirelessHART

WirelessHART surge como una adaptación de la especificación HART ya existente, que se creó en EE.UU. para la monitorización de sensores en entornos industriales. Por lo tanto, es lógico que tuviera en cuenta desde el principio las características de este tipo de redes [3]. Esta iniciativa especifica perfiles y casos prácticos en los que se puede aplicar directamente el control inalámbrico. La pila de protocolos de HART se puede observar en la Figura 2.12.

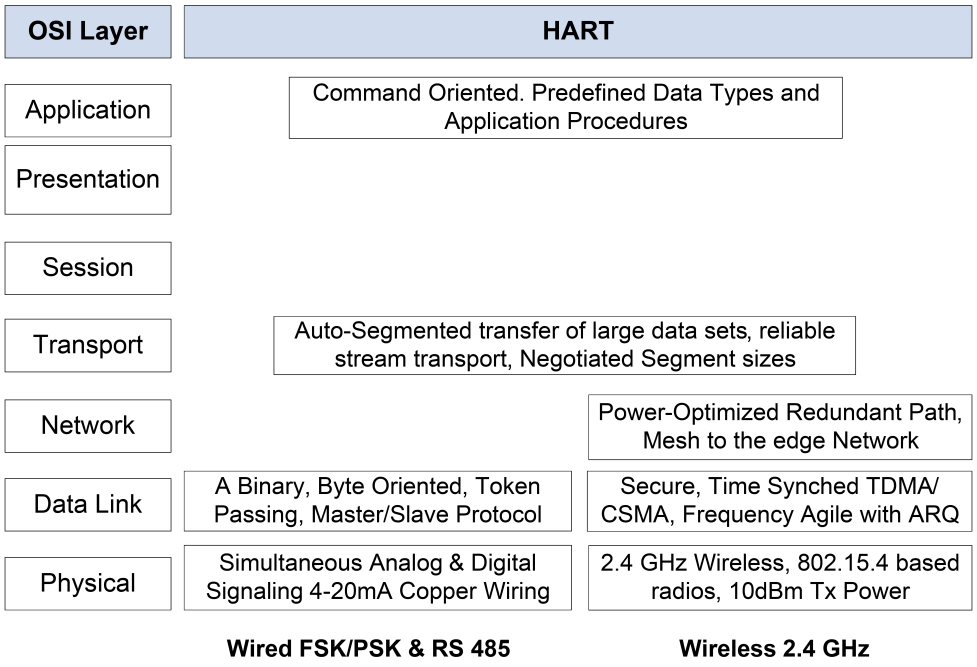


Figura 2.12: *Pila de protocolos de HART [88].*

La mayor diferencia con respecto a ZigBee es el acceso al medio, realizado mediante TDMA, ahorrando tiempo y energía en los nodos sensores. La topología también es diferente, ya que en esta especificación la red es mallada. De esta manera consigue una red fiable y eficiente energéticamente. Además, todos los nodos pueden encaminar y son iguales, haciendo mucho más sencilla su instalación y consiguiendo mayor flexibilidad. Para la generación de mensajes de información, debido a la naturaleza de la aplicación, tiene su propio protocolo (Smart Data)

que permite enviar información sólo cuando se registra un cambio en la medición del sensor.

No sólo eso, esta pila de protocolos es compatible con IEEE 802.15.4 y forma parte de un nuevo estándar que se está desarrollando, el SP100.11 [3], creado por ISA (Instrumentation, System and Automation Society) para el entorno industrial. En vez de normalizar todos los elementos del sistema, SP100.11 especifica sólo los niveles superiores de la pila, con varias implementaciones posibles a nivel inferior.

2.9.4. DUST Networks

Es una empresa líder en las redes de sensores, surgida a partir del proyecto de Berkeley y creada por su director, Kris Pister.

Esta empresa ha desarrollado su propio protocolo para redes de sensores, TSMP (*Time Synchronized Mesh Protocol*) que tiene las siguientes características [63]:

- Capa física: utiliza DSSS y FHSS para mayor fiabilidad en la comunicación entre nodos y para evitar interferencias con otras redes u obstáculos. Cuando un nodo entra a formar parte de la red, recibe la secuencia pseudo-aleatoria de frecuencias de sus vecinos y genera la suya propia.

- Capa MAC: en lugar de CSMA utiliza TDMA, sincronizando los nodos para la transmisión de información en slots de tiempo. Para la información de sincronización de los nodos utiliza los paquetes ACK. Esta manera de actuar consigue evitar colisiones y hace que el sistema sea fácilmente escalable.

- Auto-organización: todos los nodos son capaces de descubrir vecinos, ajustar la potencia de la señal RF y adquirir información de sincronización y saltos de frecuencia.

- Diversidad de enrutamiento: un nodo tiene siempre dos o más posibles rutas. Utilizará una de las dos normalmente y, si después de enviar un mensaje no recibe ACK, utilizará la otra. Si recibe un NACK, entonces repetirá la misma ruta.

- Consumo de energía: gracias al uso de un acceso al medio sincronizado, el consumo de energía se reduce drásticamente. La causa está en el ciclo de trabajo, al enviar y recibir en periodos de tiempo predeterminados, los nodos sólo necesitan estar «despiertos» en los slots que tienen asignados. Incluso en los nodos que no realizan enrutamiento se reduce

el consumo, ya que de toda la circuitería de un nodo sensor, la antena RF es la que más consume (alrededor del 95 %).

2.10. Nuevas tendencias

2.10.1. TCP/IP

De entre todos los protocolos que se generan continuamente para redes de sensores, llama la atención 6LoWPAN [25]. Este protocolo trata de adaptar a las redes de sensores el protocolo IPv6 para conseguir compatibilidad entre Internet y las redes de sensores, comunicando entre sí miles de sistemas, aprovechando las ventajas que trae consigo IPv6. Actualmente está en desarrollo y no ha visto la luz.

2.10.2. Formatos

Dada la cantidad de protocolos y formatos de datos que existen hoy en día, buena parte de ellos propietarios, no es de extrañar que surjan empresas y soluciones que intenten solventar este problema.

Si se tiene una red de sensores de hace un par de años y se quiere ampliarla, es fácil que ya haya mejores soluciones que la que se tiene instalada, pero es probable que los formatos de la información sean diferentes. Si se quiere conectar esta red a otra, se necesita de nuevo traducir la información de un formato a otro. Se han formulado dos maneras de solucionar esto [31]. La primera es utilizar un estándar ya conocido y eficiente para manejar la información. El estándar elegido es XML, por su versatilidad al utilizar etiquetas y por su presencia en Internet. De esta manera, manejar los datos y representarlos sería mucho más sencillo y permitiría la interoperabilidad entre redes de una manera fácil. Sin embargo, para ello es necesario que los fabricantes y desarrolladores acepten utilizar XML. La otra manera de evitar estos problemas es realizar un procesado en la propia red con dispositivos traductores de información y de arquitectura abierta y flexible. De este modo, con un solo dispositivo se puede conectar dos redes diferentes. Un ejemplo de traducción de formatos lo tenemos en la Figura 2.13.

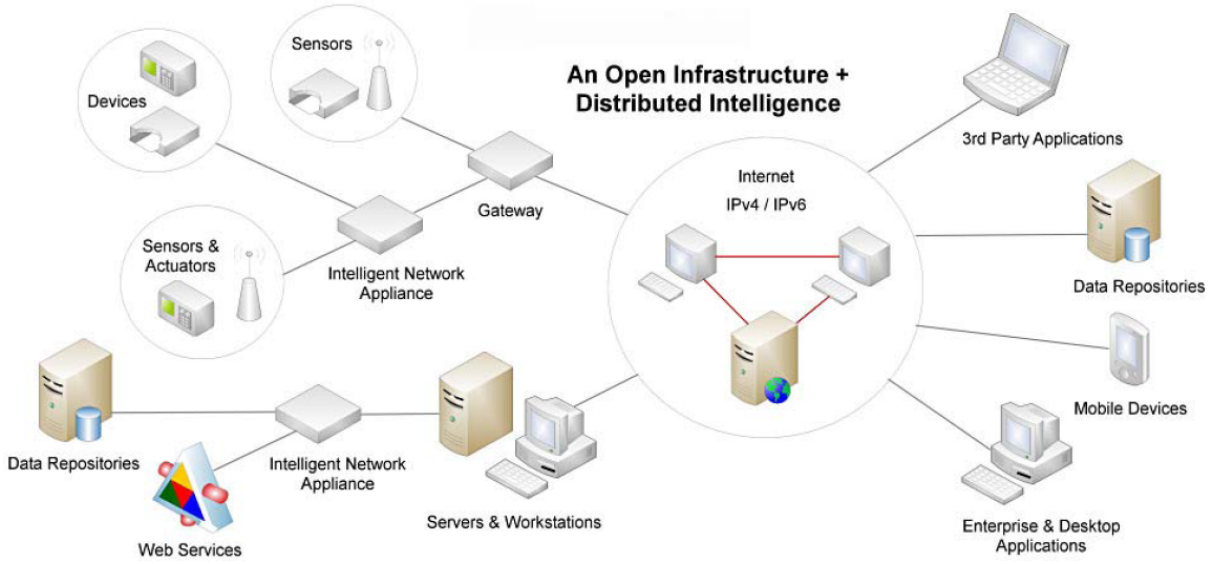


Figura 2.13: *Ejemplo de traducción de formatos en una red.*

2.10.3. Representación de datos

Una vez que se tienen los datos obtenidos por los nodos de la red de sensores, es necesario trabajar con ellos y representarlos de forma comprensible. Existe software preparado para este fin, la mayoría distribuido por los propios fabricantes junto con el hardware de la red.

Pero ésta no es la única posibilidad. Microsoft y Google [63] ofrecen un servicio de publicación de datos y representación de los mismos, de forma que dicho servicio es accesible fácilmente desde cualquier lugar del mundo. Así, de nuevo, se observa la convergencia de estas nuevas redes con Internet.

2.10.4. Procesado ubicuo

A pesar de que las redes de sensores son muy recientes, ya se empiezan a observar nuevas líneas de expansión que sobrepasan el concepto actual de red de sensores. El siguiente salto en la evolución de esta tecnología parece ser la toma de decisiones «in situ» por la propia red de sensores. Es lo que se denomina *pervasive computing*, que podría traducirse como procesado ubicuo [35].

En la evolución de las tecnologías inalámbricas y los sensores ha ido aumentando la capacidad de procesado y de autonomía de los elementos de la red. De los códigos de barras se dio el salto

a RFID sin alimentación. Cuando se conectaron entre sí módulos inalámbricos formando redes ad-hoc, se llegó a las redes de sensores. El siguiente paso es el procesado ubicuo.

En esta nueva tecnología, los nodos de la red no sólo recopilan información y la envían al usuario, sino que toman decisiones por sí mismos en función de la información obtenida. De esta manera, la rapidez de actuación y la flexibilidad de estos sistemas darían un salto cualitativo importante. La coordinación de grandes cantidades de información, como gestión de tráfico, por ejemplo, sería resuelta de una manera mucho más eficaz.

2.10.5. Web de sensores

Quizás la evolución más interesante y prometedora de las redes de sensores sea la web de sensores [91]. Es muy similar al procesado ubicuo, sólo cambia un poco el enfoque.

Al igual que las redes de sensores, las webs de sensores se definen como una red amorfa de sensores distribuidos en el espacio que se comunican inalámbricamente entre ellos. La diferencia está en que la web de sensores actúa como un único ente sensor: la información que capta un nodo es compartida con todos los nodos de la red, permitiendo la actuación de toda la web de forma conjunta y autónoma. No necesita de agentes exteriores para funcionar.

De esta manera, el alcance de actuación se incrementa notablemente. Una web de sensores es capaz de tener en cuenta la región en lugar de sus puntos y, por ejemplo, permite conocer la dirección y velocidad de un frente en movimiento.

A día de hoy ya existen webs de sensores, pero están lejos de alcanzar el potencial que tienen. La NASA es el principal promotor de esta tecnología y por el momento ya ha llevado a cabo proyectos en detección de inundaciones, de actividad volcánica y de condiciones de vida en la Antártida [91].

Sin embargo, no se aprovecha realmente las posibilidades del concepto. En realidad, la web de sensores, a día de hoy, actúa como un sensor que transmite la información a un centro de datos, en lugar de ser a la vez un sistema que reacciona según esa misma información. Eso sí, realiza un procesado de esa información para decidir cuándo es importante transmitirla y cuándo no. El avance es significativo respecto a las redes de sensores normales, en las cuales no hay procesado de la información y se trata de un sistema pasivo que traslada la información al centro de procesado.

2.11. Aplicaciones

Las redes de sensores tienen una amplia variedad de aplicaciones [32]:

- Monitorización de un hábitat (para determinar la población y comportamiento de animales y plantas).
- Monitorización del medio ambiente, observación del suelo o agua.
- Mantenimiento de ciertas condiciones físicas (temperatura, luz).
- Control de parámetros en la agricultura.
- Detección de incendios, terremotos o inundaciones.
- Estudio de los movimientos de objetos o estructuras.
- Control de tráfico.
- Mejora visual de un paciente con deficiencia óptica
- Asistencia militar o civil.
- Control de inventario.
- Control médico.
- Detección acústica.
- Cadenas de montaje, etc.

Los sensores pueden ser aplicados para realizar una monitorización continua, detección de eventos, localización y control de actuadores. Las áreas de aplicación son: Militar, Medio Ambiente, Salud, Residencias, Comercio, etc.

De hecho las redes de sensores inalámbricas (WSN) tienen el potencial de revolucionar los complejos sistemas de control u observación, tal y como hoy se entienden.

A continuación se citan varias aplicaciones con redes de sensores inalámbricas en el mundo comercial [82]:

- XSILOGY Solutions es una compañía que provee WSN para las siguientes aplicaciones comerciales: organización de inventario de tanques, sistemas de distribución de flujos, edificios comerciales, monitorización medioambiental, defensa del hogar, etc.
  (www.xsilogy.com/home/main/index.html)

- ENSCO investiga con WSN para aplicaciones meteorológicas.
  (www.in-q-tel.com/tech/dd.html)

- EMBER provee soluciones con WSN para automatización industrial, defensa y edificios inteligentes.
  (www.ember.com)

- H900 Wireless SensorNet System (TM), el primer sistema de enrutamiento para redes inalámbricas de sensores con topología mallada, desarrollado por la compañía Snsicast Systems. Sus aplicaciones van desde la electricidad a la seguridad del hogar.
  (www.sensicast.com)

- SOFTLINX desarrolla productos de seguridad perimetral basados en sensores.
  (www.soflinx.com)

- XYZ integra redes de sensores inalámbricas para el control de entornos en el interior de edificios.
  (www.cbe.berkeley.edu/research/briefswirelessxyz.htm)

- Crossbow desarrolla redes de sensores para monitorizar entornos y otras aplicaciones industriales como puentes, estructuras, calidad del aire/comida, automatización industrial...
  (www.xbow.com)

- Japan's Omron Corp ha elaborado una red de sensores para naves de carga que provee un sistema de seguridad en los puertos.
  (www.omron.com)

- Otras páginas web son: www.millennial.net, www.dust-inc.com, www.melexis.com, www.chipcon.com, www.zigbee.com.

Capítulo 3

ENCAMINAMIENTO EN REDES DE SENSORES INALÁMBRICOS

3.1. Problemática del enrutamiento en redes de sensores inalámbricos

A pesar de las innumerables aplicaciones de las redes de sensores inalámbricos, estas redes tienen varias restricciones, como la limitación del suministro de energía, la limitación de la capacidad de procesamiento y la limitación del ancho de banda de los enlaces inalámbricos entre nodos sensores. Uno de los principales objetivos de diseño es llevar a cabo la comunicación de datos mientras se intenta prolongar la vida útil de la red y evitar la degradación de la conectividad mediante el empleo de técnicas agresivas de gestión de energía. El diseño de protocolos de enrutamiento en estas redes está influido por factores difíciles. Estos factores deben superarse para que se pueda lograr una comunicación eficiente. A continuación, se resumen algunos de los problemas y cuestiones de diseño que afectan al proceso de enrutamiento en redes de sensores inalámbricos.

Despliegue del nodo: el despliegue de los nodos es dependiente de la aplicación y puede ser de tipo manual (determinista) o aleatorio. En el despliegue manual, los nodos son colocados manualmente y los datos se distribuyen a través de rutas predeterminadas. Sin embargo, en el despliegue aleatorio, los nodos sensores están dispersos al azar, creando una infraestructura de enrutamiento ad-hoc. Si la distribución resultante de los nodos no es uniforme, se necesita una

agrupación óptima para permitir que la conectividad y las funciones de red sean eficientes en energía. La comunicación entre sensores normalmente se produce en rangos cortos de transmisión debido a las limitaciones de energía y ancho de banda. Por lo tanto, lo más probable es que una ruta esté formada por múltiples saltos inalámbricos.

El consumo de energía sin perder precisión: los nodos sensores pueden agotar su suministro limitado de energía realizando el procesamiento y la transmisión de la información en un entorno inalámbrico. Como tal, son esenciales las formas de conservación de energía en la comunicación y en el procesamiento. El tiempo de vida de un nodo sensor muestra una fuerte dependencia con la vida de la batería [39]. En una red de sensores inalámbricos multisalto, cada nodo desempeña un doble papel, como transmisor de datos y como enrutador de datos. El mal funcionamiento de algunos nodos sensores debido a la falta de energía puede causar importantes cambios en la topología y, puede requerir cambios de ruta de los paquetes y la reorganización de la red.

Método de presentación de datos: la representación de los datos en las redes de sensores inalámbricos depende de la aplicación y del tiempo de entrega de los datos. Puede clasificarse en dirigido por tiempo (*time-driven*), dirigido por eventos (*event-driven*), dirigido por consultas (*query-driven*) o un híbrido de todos estos métodos. El método de entrega dirigido por tiempo es adecuado para aplicaciones que requieren monitorización de datos periódicos. Como tal, los nodos sensores encienden periódicamente sus sensores y transmisores, detectan el medio, y transmiten los datos de interés en intervalos de tiempo periódicos y constantes. En los métodos dirigido por eventos y dirigido por consultas, los nodos sensores reaccionan inmediatamente a los cambios repentinos y drásticos en el valor del atributo detectado debido a la ocurrencia de un determinado evento, o responden a una consulta generada en la red por otro nodo. Como tales, se adaptan bien a las aplicaciones de tiempo crítico. También cabe la posibilidad de una combinación de los métodos anteriores. El protocolo de enrutamiento está influenciado por el método de presentación de los datos en términos de consumo de energía y cálculos de ruta.

Heterogeneidad nodo/enlace: en muchos estudios, todos los nodos sensores se suponen que son homogéneos (es decir, tienen la misma capacidad en términos de computación, comunicación y potencia). Sin embargo, dependiendo de la aplicación, un nodo sensor puede tener una función o capacidad diferente. La existencia de un conjunto heterogéneo de sensores plantea muchas cuestiones técnicas relacionadas con los datos de enrutamiento. Por ejemplo, algunas aplicaciones pueden requerir una mezcla de diversos sensores para el control de temperatura,

presión y humedad del entorno, detección de movimiento a través de sintonías acústicas y captura de imágenes o seguimiento por vídeo de objetos en movimiento. Cualquiera de estos sensores especiales se puede desplegar de forma independiente, pudiéndose incluir en ellos las diferentes funcionalidades. Incluso estos sensores pueden generar la lectura y presentación de los datos a diferentes tasas, sujeto a diversas restricciones de calidad de servicio (QoS), y pueden seguir múltiples modelos de presentación de datos. Por ejemplo, los protocolos jerárquicos designan un nodo organizador de clúster diferente de los sensores normales. Estos organizadores de grupos pueden ser más poderosos que otros nodos sensores en términos de energía, ancho de banda y memoria.

Tolerancia a fallos: algunos nodos sensores pueden fallar o ser bloqueados debido a la falta de potencia, daños físicos o interferencias en el medio. El fallo de los nodos sensores no debe afectar a la tarea global de la red. Si muchos nodos fallan, el control de acceso al medio (MAC) y los protocolos de enrutamiento deben adaptarse a la formación de nuevas conexiones y rutas para la recogida de datos. Esto puede requerir que los nodos tengan que adaptarse al consumo de potencias de transmisión o a cambios en las rutas de los paquetes a regiones donde hay más energía disponible. Por lo tanto, pueden ser necesarios múltiples niveles de redundancia en la tolerancia a fallos de la red de sensores.

Escalabilidad: el número de nodos sensores desplegados en la zona de detección puede ser del orden de cientos o miles, o más. Cualquier sistema de enrutamiento debe ser capaz de trabajar con este enorme número de nodos sensores. Además, los protocolos de enrutamiento de la red de sensores deben ser lo suficientemente escalables para responder a los acontecimientos en el medio. Hasta que ocurra un suceso, la mayoría de sensores pueden permanecer en estado de suspensión, con los datos de los pocos sensores restantes proporcionando el grueso de la calidad.

Red dinámica: en muchos estudios, los nodos sensores se suponen fijos. Sin embargo, en numerosas aplicaciones los nodos sensores pueden ser móviles [98]. Como tal, el enrutamiento de mensajes desde o hacia los nodos móviles es más difícil ya que la estabilidad de la ruta y la topología empieza a ser una cuestión importante, además de la energía y el ancho de banda. Por otra parte, el fenómeno puede ser móvil (por ejemplo, una aplicación de detección o seguimiento de un objetivo). La detección de eventos fijos le permite a la red trabajar en un modo reactivo (es decir, generando tráfico cuando es necesario informar), mientras que los eventos dinámicos requieren el envío de información periódica en la mayoría de aplicaciones.

Medios de transmisión: en una red de sensores multisalto, la comunicación entre los

nodos se realiza por un medio inalámbrico. Los problemas asociados a un canal inalámbrico (por ejemplo, desvanecimiento, alta tasa de error) también pueden afectar al funcionamiento de la red de sensores. En general, se requiere un ancho de banda bajo para los datos del sensor, del orden de 1-100 kb/s. El diseño de la MAC está relacionado con el medio de transmisión. Un enfoque de diseño de la MAC para redes de sensores es el uso de protocolos basados en Acceso Múltiple por División en el Tiempo (TDMA) que conservan más energía que los protocolos basados en contienda como Acceso Múltiple por Detección de Portadora (CSMA) (por ejemplo, IEEE 802.11).

Conectividad: la alta densidad de nodos en las redes de sensores excluye totalmente que estén aislados unos de otros. Por lo tanto, se espera que los nodos sensores estén muy conectados. Esto, sin embargo, no puede impedir que la topología de red sea variable y el tamaño de la red se reduzca debido a fallos de los nodos. Además, la conectividad depende posiblemente de la distribución aleatoria de nodos.

Cobertura: en estas redes, cada nodo sensor obtiene una cierta visión del medio. Una visión determinada del medio está limitada en rango y precisión, ya que sólo puede cubrir un espacio físico reducido del medio. Por lo tanto, el área de cobertura también es un parámetro importante de diseño y un problema.

Agregación de datos: dado que los nodos sensores pueden generar un gran número de datos redundantes, los paquetes similares de múltiples nodos pueden ser agregados para reducir el número de transmisiones. La agregación de datos es la combinación de datos de diferentes fuentes según cierta función de agregación. Esta técnica se ha utilizado para conseguir la optimización de energía y de transferencia de datos en una serie de protocolos de enrutamiento. Los métodos de procesado de señal también pueden ser utilizados para la agregación de datos. En este caso, se refiere a la fusión de datos cuando un nodo es capaz de producir una señal de salida de mayor precisión mediante el uso de algunas técnicas, como conformación de haz (beamforming), para combinar las señales de entrada y reducir el ruido en estas señales.

Calidad del servicio: en algunas aplicaciones, los datos deben ser entregados dentro de un cierto período de tiempo desde el momento en que se detectan o no tendrán validez. Por lo tanto, la latencia para la entrega de datos es otra condición para aplicaciones de tiempo limitado. Sin embargo, en muchas aplicaciones, la conservación de energía, que está directamente relacionada con el tiempo de vida de la red, se considera relativamente más importante que la calidad de los datos enviados. Como la energía se agota, se le puede requerir a la red que reduzca la calidad de

los resultados con el fin de reducir la disipación de la energía en los nodos y, por tanto, alargar el tiempo de vida total de la red. Por lo tanto, para cumplir este requisito se utilizan los protocolos de enrutamiento con ahorro de energía.

3.2. Protocolos de encaminamiento en redes de sensores inalámbricos

Los protocolos de encaminamiento en redes inalámbricas de sensores se pueden clasificar generalmente en dos grupos, de acuerdo a la estructura de la red o al criterio de encaminamiento utilizado.

Dependiendo de la estructura de la red, se puede tener encaminamiento plano en el que todos los nodos desempeñan el mismo papel, encaminamiento jerárquico, que tiene por objeto agrupar los nodos para que los organizadores de la agrupación puedan hacer la agregación y reducción de datos con el fin de ahorrar energía, y encaminamiento basado en localización, que utilizan la información de posición para transmitir los datos hasta las regiones deseadas de la red.

Según el criterio de encaminamiento, los protocolos pueden clasificarse en basados en múltiples rutas, consultas, negociación, calidad de servicio (QoS) y basados en coherencia.

Estas estrategias de encaminamiento se consideran adaptativas si ciertos parámetros pueden ser controlados con el fin de amoldarse a las condiciones actuales de la red y a los niveles de energía disponibles.

La clasificación citada anteriormente se puede ver en la Figura 3.1.

Además de lo anterior, los protocolos de enrutamiento se pueden clasificar en tres categorías, proactiva, reactiva, e híbrida, en función de cómo encuentra la fuente una ruta hasta el destino. En los protocolos proactivos, todas las rutas se calculan antes de que sean realmente necesarias, mientras que en los protocolos reactivos, las rutas se calculan cuando se demandan. Los protocolos híbridos utilizan una combinación de estas dos ideas.

Otra clase de protocolos de encaminamiento son los llamados cooperativos. En este tipo de protocolos, los nodos envían datos a un nodo central donde se pueden agregar datos y pueden ser objeto de procesamiento posterior, por lo tanto se reducen los costes en términos de utilización de energía.

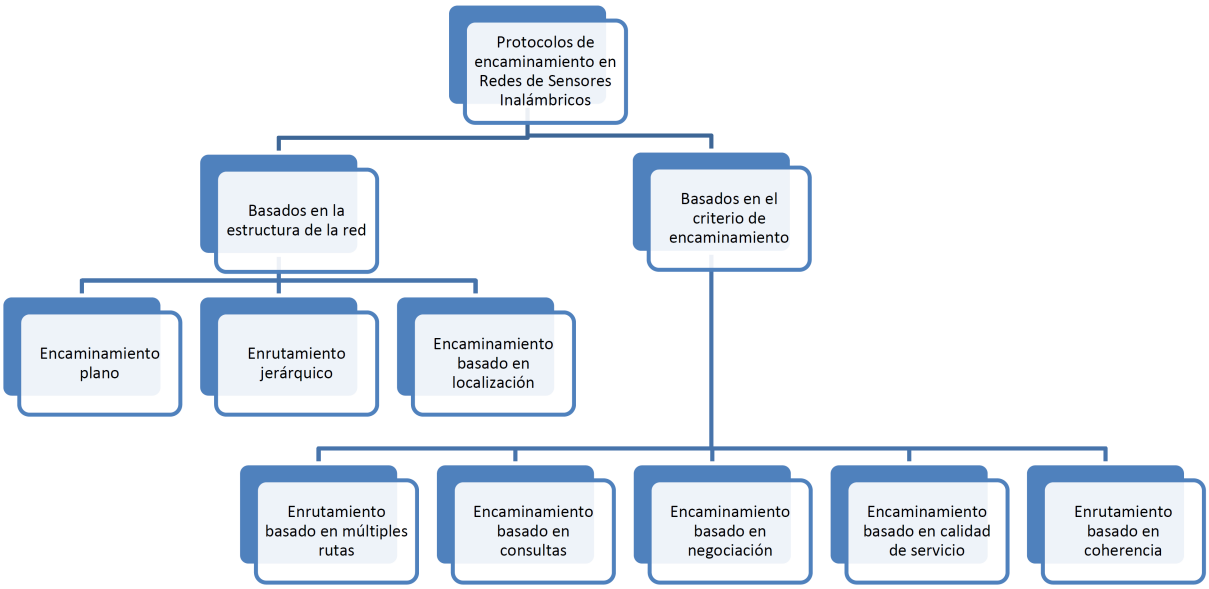


Figura 3.1: *Protocolos de encaminamiento en redes de sensores inalámbricos.*

3.2.1. Protocolos de basados en la estructura de la red

La estructura de la red puede desempeñar un papel importante en el funcionamiento del protocolo de enrutamiento en redes de sensores inalámbricos. En esta sección se estudia más en detalle los protocolos que entran en esta categoría.

3.2.1.1. Encaminamiento plano

La primera categoría de protocolos de enrutamiento son los protocolos de encaminamiento plano. En las redes planas, los nodos normalmente desempeñan el mismo papel y colaboran entre sí para llevar a cabo la tarea de sensado. Debido al gran número de nodos, no es posible asignar un identificador global para cada nodo. Esta consideración ha llevado al encaminamiento centrado en datos, donde la estación base envía consultas a determinadas regiones y espera los datos de los sensores ubicados en las regiones seleccionadas. Puesto que los datos se solicitan a través de consultas, se necesita un nombrado basado en atributo para especificar las propiedades de los datos. Los primeros trabajos en encaminamiento centrado en datos (por ejemplo, SPIN y difusión dirigida [97]) se dedicaron a ahorrar energía a través de la negociación de datos y la eliminación de datos redundantes. Estos dos protocolos motivaron el diseño de otros muchos protocolos que persiguen un concepto similar.

Protocolos de sensores para información vía negociación: Heinzelman en [40, 48] propuso una familia de protocolos adaptativos llamados Protocolos de Sensores para Información vía Negociación (*Sensor Protocols for Information via Negotiation* (SPIN)) que distribuyen toda la información de cada nodo a todos los nodos de la red, suponiendo que todos los nodos en la red son posibles estaciones base. Esto permite a un usuario consultar cualquier nodo y obtener inmediatamente la información solicitada. Estos protocolos hacen uso de la propiedad de que los nodos cercanos tienen datos similares, y por lo tanto sólo existe la necesidad de distribuir los datos a otros nodos. La familia de protocolos SPIN utiliza la negociación de datos y algoritmos adaptativos de recursos. Los nodos con SPIN funcionando asignan un nombre de alto nivel para describir completamente sus datos recopilados (llamados metadatos) y realizan las negociaciones de metadatos antes de que los datos sean transmitidos. Esto asegura que no se envían datos redundantes a través de la red. La semántica del formato de los meta-datos no se especifica en SPIN, por tanto es tarea de las aplicaciones definir dicha semántica. Además, SPIN tiene acceso al nivel de energía del nodo y adapta el protocolo que se está ejecutando dependiendo de la cantidad de energía restante. Estos protocolos trabajan dirigidos por tiempo y distribuyen la información sobre toda la red, incluso cuando el usuario no solicita ningún dato.

La familia SPIN está diseñada para abordar las deficiencias de la inundación por negociación. La familia de protocolos SPIN se diseña basándose en dos ideas básicas:

1. Los nodos sensores operan de forma más eficiente y conservan la energía mediante el envío de datos que describen la información del sensor, en lugar de enviar todos los datos.

2. Los protocolos convencionales, como la inundación, desperdician energía y ancho de banda cuando envían, de forman adicional e innecesaria, copias de datos de sensores que cubren áreas solapadas. Los inconvenientes de la inundación incluyen la implosión, que es causada por duplicar los mensajes enviados al mismo nodo. Esto ocurre cuando dos nodos están sensando la misma región y envían paquetes similares a los mismos vecinos, dando lugar a un consumo de grandes cantidades de energía sin considerar las limitaciones de la misma.

Los meta-datos de la negociación SPIN resuelven los problemas clásicos de las inundaciones, logrando así una gran eficiencia energética. SPIN es un protocolo de tres fases, y por tanto, los nodos sensores usan tres tipos de mensajes, ADV, REQ, y DATA, para comunicarse. ADV se utiliza para anunciar nuevos datos, REQ para solicitar datos, y DATA para enviar datos. El

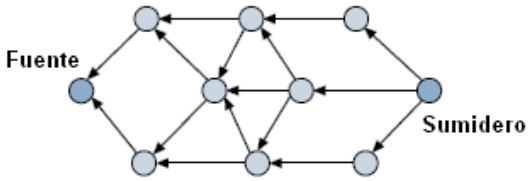
protocolo se inicia cuando un nodo SPIN obtiene nuevos datos que está dispuesto a compartir. Lo hace mediante radiodifusión y un mensaje ADV que contiene metadatos. Si un vecino está interesado en los datos, envía un mensaje REQ para los datos y los datos se envían a este nodo vecino. El nodo sensor vecino repite este proceso con sus vecinos. Como resultado, toda la zona del sensor recibirá una copia de los datos.

Una de las ventajas de SPIN es que los cambios topológicos están localizados, ya que cada nodo sólo necesita saber sus vecinos de un salto. SPIN proporciona más ahorro de energía que la inundación, y una negociación con metadatos con casi la mitad de los datos redundantes. Sin embargo, el mecanismo que anuncia los datos SPIN no puede garantizar la entrega de los mismos. Para ver esto, se puede considerar una aplicación de detección de intrusos, donde los datos deben ser entregados de manera fiable a intervalos periódicos. Asumiendo que los nodos interesados en los datos se encuentran lejos del nodo fuente, y los nodos entre el origen y destino no están interesados en esos datos, estos datos no serán entregados al destino.

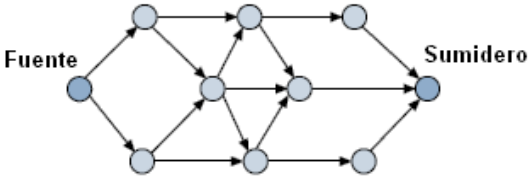
Difusión dirigida: en [42], C. Intanagonwiwat propuso un popular paradigma de agregación de datos para redes de sensores inalámbricos llamado difusión dirigida. La difusión dirigida es un paradigma centrado en datos (DC) en el sentido de que todos los datos generados por los nodos sensores se denominan por pares atributo-valor. La idea principal del paradigma DC es combinar los datos procedentes de distintas fuentes en la ruta (agregación de red) eliminando la redundancia, reduciendo al mínimo el número de transmisiones, ahorrando energía y prolongando el tiempo de vida. A diferencia del encaminamiento punto a punto, se buscan rutas desde múltiples fuentes a un único destinatario, que es el encargado de realizar la agregación.

En la difusión dirigida, los sensores miden eventos y crean gradientes de información en sus respectivos vecinos. La estación base pide datos mediante la transmisión broadcasting de «intereses». Un interés describe una tarea que debe hacerse por la red, dicho interés se difunde a través de la red salto a salto, y es emitido broadcast por cada nodo a sus vecinos. Cada sensor que recibe el interés, establece un gradiente hacia los nodos sensores de los que recibe el interés. Este proceso continúa hasta que se crean gradientes de las fuentes a la estación base. En general, un gradiente especifica un valor de atributo y una dirección. La fuerza del gradiente puede ser diferente hacia diferentes vecinos, dando lugar a diferentes cantidades de flujo de información. Cuando los intereses ajustan los gradientes, las rutas de flujo de información se forman a partir de múltiples caminos, y los mejores caminos se refuerzan para evitar una mayor inundación de acuerdo con una regla local. Con el fin de reducir costes de comunicación, los datos se agregan

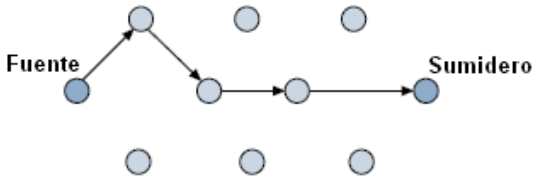
en el camino. El objetivo es encontrar un buen árbol de agregación que permita el envío de datos desde el nodo fuente a la estación base. La estación base se actualiza periódicamente y reenvía el interés cuando comienza a recibir datos de la/s fuente/s. Esto es necesario porque los intereses no son transmitidos de forma fiable a través de la red. En la Figura 3.2 se pueden observar las fases de este algoritmo.



(a) Propagación de intereses



(b) Establecimiento de gradientes



(c) Envío de datos y refuerzo del camino

Figura 3.2: *Difusión dirigida*

Todos los nodos en una red basada en difusión dirigida permiten la difusión para lograr ahorro de energía mediante la selección empírica de buenos caminos, el almacenamiento en caché y el procesamiento de datos en la red. El cacheado puede aumentar la eficiencia, robustez y escalabilidad de la coordinación entre los nodos sensores, que es la esencia del paradigma de difusión de datos. Otros usos de la difusión dirigida es la propagación espontánea de eventos importantes para algunos sectores de la red de sensores. Ese tipo de recuperación de información sólo se adapta bien a peticiones persistentes, donde los nodos solicitantes no esperan, durante período de tiempo, datos que satisfacen una consulta. Esto hace que sea inadecuado para consultas que se realizan una sola vez, ya que no merece la pena la creación de gradientes para consultas que utilizan la ruta una vez.

Enrutamiento por rumor: es una variación de la difusión dirigida y está previsto para aplicaciones donde no es factible el enrutamiento geográfico. En general, la difusión dirigida utiliza inundación para inyectar la consulta a toda la red cuando no hay un criterio geográfico para difundir tareas. Sin embargo, en algunos casos sólo hay una pequeña cantidad de datos

que se solicitan a partir de los nodos, por lo que el uso de las inundaciones es innecesario. Un enfoque alternativo es inundar los eventos si el número de eventos es pequeño y el número de consultas es muy grande. La idea clave es encaminar las consultas a los nodos que han observado un evento en particular, en lugar de las inundaciones en toda la red, para obtener información acerca de los eventos ocurridos. Para las inundaciones de eventos a través de la red, el algoritmo de encaminamiento por rumor emplea paquetes de larga vida llamados agentes. Cuando un nodo detecta un evento, agrega el evento a su tabla local, llamada tabla de eventos, y genera un agente [10]. Los agentes viajan por la red para difundir información acerca de eventos locales a nodos distantes. Cuando un nodo genera una consulta para un evento, los nodos que conocen la ruta pueden responder a la consulta mirando su tabla de eventos.

Por lo tanto, no hay necesidad de inundación en toda la red, lo que reduce el coste de comunicación. Por otra parte, el encaminamiento por rumor mantiene sólo un camino entre el origen y el destino, al contrario que la difusión dirigida donde los datos pueden ser enrutados a través de múltiples vías. Además, puede ahorrar energía en comparación con las inundaciones de eventos y también puede manejar fallos en los nodos. Sin embargo, funciona bien sólo cuando el número de eventos es pequeño. Para un gran número de eventos, el coste del mantenimiento de agentes y tablas de eventos en cada nodo se convierte en inviable si no hay suficiente interés en estos eventos. Además, la sobrecarga asociada se controla mediante diferentes parámetros utilizados en el algoritmo, como el tiempo de vida (TTL) perteneciente a las consultas y agentes. Dado que los nodos empiezan a conocer los eventos a través de los agentes de eventos, la heurística para definir la ruta de un agente afecta el rendimiento de la selección del próximo salto.

Algoritmo de reenvío de coste mínimo: este protocolo aprovecha el hecho de que la dirección de enrutamiento siempre es conocida (es decir, hacia la estación base externa fijada). Por lo tanto, un nodo sensor no necesita tener un identificador exclusivo ni mantener una tabla de enrutamiento. En lugar de ello, cada nodo mantiene la estimación del menor coste desde él a la estación base [97]. Cada nodo envía los mensajes de forma broadcast a sus vecinos. Cuando un nodo recibe el mensaje, comprueba si está en la ruta de menor coste entre el nodo sensor fuente y la estación base. Si éste es el caso, se retransmite el mensaje a sus vecinos. Este proceso se repite hasta que se llegue a la estación base.

Además, cada nodo debe conocer el menor coste estimado desde sí mismo a la estación base. Éste se obtiene de la siguiente manera. La estación base transmite de forma broadcast un mensaje con el coste a cero, mientras que cada nodo inicializa a infinito el menor coste a

la estación base. Cada nodo que recibe el mensaje broadcast originado por la estación base, comprueba si la estimación en el mensaje más el coste del enlace en el que se recibe es inferior a la estimación actual. En caso afirmativo, la estimación actual y la estimación en el mensaje de difusión se actualizan. Si el mensaje de difusión recibido es actualizado, se reenvía, de lo contrario, se elimina. Sin embargo, el procedimiento anterior puede dar lugar a que algunos nodos tengan varias actualizaciones, y además, los nodos lejos de la estación base obtengan más actualizaciones que los cercanos a ella. Para evitar esto, el algoritmo de reenvío de coste mínimo fue modificado para ejecutar un algoritmo de backoff en la fase de instalación. El algoritmo de backoff determina que un nodo no enviará el mensaje de actualización hasta que han transcurrido $a * l_c$ unidades de tiempo desde el momento en que el mensaje fue actualizado, donde a es una constante y l_c es el coste del enlace en el cual el mensaje fue recibido.

Encaminamiento basado en gradiente: Schurgers [84] propuso una variante de la difusión dirigida. La idea clave es memorizar el número de saltos cuando el interés se difunde a través de toda la red. Como tal, cada nodo puede calcular un parámetro llamado altura del nodo, que es el número mínimo de saltos para alcanzar la estación base. La diferencia entre la altura de un nodo y la de su vecino se considera el gradiente del enlace. Un paquete se envía a través del enlace con mayor gradiente. Se utilizan algunas técnicas auxiliares como la agregación de datos y la difusión de tráfico para dividir uniformemente el tráfico a través de la red. Cuando varias rutas pasan a través de un nodo que actúa como nodo retardo, este nodo retardo puede combinar datos de acuerdo con una función determinada. En este encaminamiento, han sido objeto de debate tres técnicas de difusión de datos:

- Un esquema estocástico, si un nodo recoge un gradiente al azar cuando hay dos o más saltos tienen el mismo gradiente.
- Un esquema basado en energía, donde un nodo incrementa su altura cuando la energía cae por debajo de un umbral determinado, de manera que otros sensores son descartados para enviar datos a ese nodo.
- Un esquema basado en corrientes, donde las nuevas corrientes no se enrutan a través de nodos que forman parte actualmente de la ruta de acceso de otras corrientes. El objetivo principal de estos sistemas es obtener una distribución equilibrada del tráfico en la red, aumentando así el tiempo de vida de la misma. Este esquema supera a la difusión dirigida en términos de energía total en la comunicación.

Encaminamiento de sensores dirigidos por información y encaminamiento de difusión anisótropa restringida: dos técnicas de enrutamiento, *information-driven sensor querying* (IDSQ) y *constrained anisotropic diffusion routing* (CADR), se proponen en [22]. CADR pretende ser una forma general de la difusión dirigida. La idea clave es consultar sensores y rutas de datos en la red, de tal manera que la ganancia de información se maximiza mientras se minimizan la latencia y el ancho de banda. CADR difunde consultas por medio de un conjunto de criterios de información para seleccionar qué sensores pueden obtener los datos. Esto se logra mediante la activación de los sensores que están cerca de un evento particular y el ajuste dinámico de las rutas de datos. En CADR, cada nodo evalúa un objetivo de información/coste, las rutas de datos basadas en el gradiente de información/coste local y las necesidades de los usuarios. En IDSQ, el nodo consultado puede determinar qué nodo puede proporcionar la información más útil, con la ventaja adicional de equilibrar el coste de energía. Sin embargo, IDSQ no define específicamente la consulta y la información que se enruta entre los sensores y la estación base. Por lo tanto, IDSQ puede considerarse como un procedimiento complementario de optimización. Estos enfoques son más eficientes energéticamente que la difusión dirigida, donde las consultas se difunden de manera isotrópica y alcanzan primero los vecinos más cercanos.

COUGAR: otro protocolo centrado en datos es el llamado COUGAR [96], que considera la red como un sistema enorme de base de datos distribuida. La idea es utilizar consultas declarativas mediante el procesamiento de consultas abstractas de las funciones de la capa de red, como puede ser la selección de los sensores relevantes. COUGAR utiliza la agregación de datos en la red para conseguir más ahorro de energía. La abstracción es soportada a través de una capa adicional de consulta que se encuentra entre las capas de red y aplicación. COUGAR incorpora una arquitectura para el sistema de base de datos del sensor, donde los nodos sensores seleccionan un nodo líder para realizar la agregación y la transmisión de los datos a la estación base. La estación base es responsable de generar un plan de consulta que especifica la información necesaria sobre el flujo de datos y la computación de la red para la próxima consulta, y lo envía a los nodos. El plan de consulta también describe cómo seleccionar un líder para la consulta. La arquitectura proporciona capacidad de computación a la red, dando lugar a una eficiencia de energía en situaciones donde los datos generados son enormes. COUGAR proporciona una capa de red independiente del método de consulta de datos. Sin embargo, tiene algunos inconvenientes. En primer lugar, la adición de una capa de consulta en cada uno de los nodos sensores puede añadir sobrecarga extra en términos de consumo de energía y de almacenamiento de memoria. En

segundo lugar, se necesita la sincronización entre los nodos para obtener éxito en el procesamiento de datos en la red (no todos los datos se reciben al mismo tiempo de las fuentes) antes de enviar los datos al nodo líder. En tercer lugar, los nodos líderes se deben mantener dinámicamente para evitar que éstos sean propensos a fallos.

ACQUIRE: en [81], Sadagopan propuso una técnica para consultas en redes de sensores llamada Reenvío de Consultas Activas en Redes de Sensores (*Active Qwery Forwarding in Sensor Networks* (ACQUIRE)). Similar a COUGAR, ACQUIRE ve la red como una base de datos distribuida donde las consultas complejas pueden dividirse en varias subconsultas. El nodo estación base envía una consulta, que es retransmitida por cada nodo que la recibe. Durante esto, cada nodo intenta responder a la consulta utilizando su información precacheada y luego la envía a otro nodo sensor. Si la información no está precacheada hasta la fecha, los nodos reúnen información de sus vecinos que se encuentran como máximo a d saltos. Una vez que la consulta se resuelva por completo, se envía de vuelta a través del camino más corto a la estación base. Por lo tanto, ACQUIRE puede hacer frente a consultas complejas permitiendo a muchos nodos enviar respuestas. La difusión dirigida no se puede utilizar para consultas complejas porque utiliza un mecanismo de consultas basado en inundaciones para consultas continuas y agregadas, y daría lugar a un consumo excesivo de energía. Por otro lado, ACQUIRE puede proporcionar consultas eficientes ajustando el valor del parámetro d. Cuando d es igual al diámetro de la red, ACQUIRE se comporta de manera similar a las inundaciones. Sin embargo, la consulta tiene que pasar más saltos si d es demasiado pequeño. Para seleccionar el siguiente nodo para la transmisión de la consulta, ACQUIRE escoge al azar o basa la selección en la máxima satisfacción posible de la consulta.

Encaminamiento basado en ahorro de energía: este protocolo [86] reactivo iniciado por el destino tiene como objetivo aumentar el tiempo de vida de la red. Aunque este protocolo es similar a la difusión dirigida, difiere en el sentido de que mantiene un conjunto de rutas en lugar de mantener o forzar una ruta óptima con la tasa más alta. Estos caminos se mantienen y eligen por medio de una cierta probabilidad. El valor de esta probabilidad depende de cómo se logra el consumo de energía en cada ruta. Debido a que los caminos se eligen en diferentes momentos, la energía de un solo camino no se agotará rápidamente. Esto puede lograr un mayor tiempo de vida de la red además de disipar la energía más equitativamente entre todos los nodos. El protocolo inicia una conexión a través de inundaciones localizadas que se utilizan para descubrir todas las rutas y los costes entre un par fuente/destino, de esta manera se crean las tablas de rutas. Los

caminos de coste alto se descartan, y se construye una tabla de encaminamiento mediante la elección de los nodos vecinos de una manera que sea proporcional a su coste. Posteriormente, las tablas de enrutamiento se utilizan para enviar datos al destino con una probabilidad inversamente proporcional al coste del nodo. Las inundaciones localizadas son realizadas por el nodo destino para mantener las rutas activas. En comparación con la difusión dirigida, este protocolo establece una mejora general en ahorro de energía y aumento en el tiempo de vida de la red. Sin embargo, el enfoque requiere reunir información sobre la ubicación y un mecanismo de establecimiento de direcciones de los nodos, lo que complica el establecimiento de la ruta en comparación con la difusión dirigida.

Encaminamiento con pasos aleatorios: el objetivo de esta técnica es lograr un equilibrio de carga haciendo uso del enrutamiento multitrayecto en redes de sensores inalámbricos [85]. Esta técnica considera sólo redes a gran escala donde los nodos tienen movilidad muy limitada. En este protocolo, se supone que los nodos sensores se pueden activar o desactivar de forma aleatoria. Para encontrar una ruta desde un origen a su destino, la información de localización se obtiene calculando la distancia entre nodos mediante el uso de la versión asíncrona del algoritmo de *Bellman-Ford*. Un nodo intermedio se selecciona como el siguiente salto al nodo vecino que está más cerca del destino de acuerdo con una probabilidad calculada. El algoritmo de enrutamiento es simple, ya que los nodos están obligados a mantener poca información de estado. Además, las distintas rutas se eligen en diferentes momentos, incluso para el mismo par de nodos origen y destino. Sin embargo, la principal preocupación de este protocolo es que a veces la topología de la red no es práctica.

3.2.1.2. Encaminamiento jerárquico

Propuesto originalmente en redes inalámbricas, son técnicas con ventajas especiales relacionadas con la escalabilidad y una comunicación eficiente. Como tal, el concepto de enrutamiento jerárquico es utilizado también para realizar un encaminamiento eficiente en términos de energía en redes inalámbricas de sensores. En una arquitectura jerárquica, los nodos de mayor energía pueden ser utilizados para procesar y enviar la información, mientras que los nodos de baja energía pueden ser usados para realizar la detección en la proximidad del objetivo. La creación de grupos y la asignación de tareas especiales a los jefes de grupos, puede contribuir enormemente a la escalabilidad del sistema global, el tiempo de vida y la eficiencia energética. El enrutamiento

jerárquico es una forma eficaz para reducir el consumo energético en un grupo, realizando la agregación de datos con el fin de disminuir el número de mensajes transmitidos a la estación base. Está compuesto principalmente de dos capas, donde una capa se utiliza para seleccionar el líder del grupo y la otra para el encaminamiento. Sin embargo, la mayoría de las técnicas en esta categoría no tienen que ver con rutas, sino con «quién y cuándo enviar o procesar/agregar» la información y distribución de canal.

Protocolo LEACH: Heinzelman [39] introdujo un algoritmo de agrupamiento jerárquico para redes de sensores, denominado Jerarquía de Clustering Adaptativa de Baja Energía (*Low Energy Adaptive Clustering Hierarchy* (LEACH)). Es un protocolo basado en agrupamiento que incluye la formación de grupos distribuidos. Selecciona aleatoriamente unos pocos nodos como los jefes de clúster (CHs) y rota esta función para distribuir uniformemente la carga de energía entre los nodos de la red. En LEACH, los nodos CH comprimen los datos que les llegan de los nodos que pertenecen a sus respectivos grupos, y envían un paquete agregado a la estación base para reducir la cantidad de información que debe ser transmitida a dicha estación. La recogida de datos es centralizada y se lleva a cabo periódicamente. Por lo tanto, este protocolo es apropiado cuando se necesita una monitorización constante por parte de la red de sensores. Un usuario podría no necesitar todos los datos inmediatamente, en este caso, las transmisiones de datos periódicos son innecesarias, y pueden agotar la energía limitada de los nodos sensores. Después de un determinado intervalo de tiempo, se lleva a cabo la rotación aleatoria del rol de CH dando lugar a una disipación de energía uniforme en la red. En los estudios realizados de este protocolo las conclusiones reflejan que sólo el 5 por ciento de los nodos deben actuar como CHs.

El funcionamiento de LEACH está separado en dos fases, la fase de configuración y la fase de estado estacionario. En la fase de configuración los grupos están organizados y son seleccionados los CHs. En la fase de estado estacionario tiene lugar la transferencia de datos a la estación base. La duración de la fase de estado estacionario es más larga que la duración de la fase de configuración con el fin de minimizar los gastos generales. Durante la fase de configuración se eligen una determinada fracción de nodos como CHs. Todos los CHs elegidos anuncian mediante un mensaje broadcast al resto de los nodos de la red que son los nuevos CHs. Todos los nodos no-CH, después de recibir este anuncio, deciden el grupo al que desean pertenecer. Esta decisión está basada en la intensidad de la señal de aviso. Los nodos no-CH informan a los CHs apropiados que serán un miembro de la agrupación. Después de recibir todos los mensajes de los nodos que

deseen ser incluidos en el clúster y en función del número de nodos del clúster, el nodo CH crea un programa TDMA y asigna a cada nodo una ranura de tiempo para transmitir. Este programa se transmite broadcast a todos los nodos del clúster.

Durante la fase de estado estacionario, los nodos sensores pueden comenzar la detección y transmisión de datos a los CHs. El nodo CH, después de recibir todos los datos, los agrega antes de enviarlos a la estación base. Después de un cierto tiempo, que es determinado a priori, la red pasa a la fase de configuración de nuevo y entra en otra ronda de selección de nuevos CHs. Cada grupo se comunica utilizando diferentes códigos CDMA para reducir la interferencia con los nodos que pertenecen a otros grupos.

Aunque LEACH es capaz de aumentar el tiempo de vida de la red, todavía hay una serie de cuestiones acerca de las hipótesis utilizadas en este protocolo. LEACH asume que todos los nodos pueden transmitir con la suficiente potencia para llegar a la estación base y que cada nodo tiene la potencia computacional para dar soporte a diferentes protocolos MAC. Por lo tanto, no es aplicable a redes desplegadas en grandes regiones. También asume que los nodos siempre tienen datos que enviar, y los nodos que están situados cerca unos de otros tienen datos correlacionados. No está claro cómo se distribuye de manera uniforme el número predeterminado de CHs través de la red. Por lo tanto, existe la posibilidad de que los CHs elegidos estén concentrados en una parte de la red, por lo que algunos nodos no tendrán ningún CH en sus alrededores. Además, la idea de agrupamiento dinámico da lugar a una sobrecarga extra (cambios de cabeza de grupo, avisos, etc), que puede disminuir la ganancia en el consumo de energía. Por último, el protocolo supone que todos los nodos comienzan con la misma cantidad de energía en cada ronda de elección, suponiendo que un CH consume aproximadamente la misma cantidad de energía para cada nodo.

Concurrencia eficiente en potencia en sistemas de información de sensores: en [54], se propuso el protocolo denominado *Power-Efficient Gathering in Sensor Information Systems* (PEGASIS). La idea básica del protocolo es que, para ampliar el tiempo de vida de la red, los nodos se organizan en cadenas en las que un nodo sólo puede transmitir su información al siguiente nodo de la cadena (después de haber agregado sus datos a los ya recibidos), hasta que el último de ellos se comunica directamente con la estación base. De la misma forma que en LEACH, PEGASIS también funciona en base a rondas, de modo que se va alternano el nodo que tiene comunicación directa con la estación base. Cuando termina la ronda de comunicación de todos los nodos con la estación base, comienza una nueva ronda, y así sucesivamente. Esto reduce

la potencia necesaria para transmitir datos por ronda así como extiende la potencia consumida de manera uniforme en todos los nodos. Por lo tanto, PEGASIS tiene dos objetivos principales. En primer lugar, aumentar la vida útil de cada uno de los nodos mediante el uso de técnicas de colaboración. En segundo lugar, permitir sólo la coordinación local entre los nodos que están cerca, de modo que el ancho de banda consumido en la comunicación se reduce.

Para encontrar el nodo vecino más cercano, cada nodo utiliza la intensidad de la señal para medir la distancia a todos los nodos vecinos y, a continuación, ajusta la intensidad de la señal de modo que sólo un nodo pueda ser escuchado. La cadena hasta la estación base constará de los nodos que están más cerca entre sí y forman una ruta hasta dicha estación. PEGASIS es capaz de aumentar la vida útil de la red al doble que bajo el protocolo LEACH. Esa ganancia se logra mediante la eliminación de la sobrecarga causada por la formación de grupos dinámicos en LEACH, y disminuyendo el número de transmisiones y recepciones de datos mediante el uso de agregación de datos. Aunque la sobrecarga de la agrupación se evita, PEGASIS requiere un ajuste dinámico de la topología, ya que un nodo sensor necesita conocer el estado de energía de sus vecinos, con el fin de conocer la ruta de los datos. Ese ajuste de topología puede introducir importantes sobrecargas, especialmente para redes altamente utilizadas. Además, PEGASIS considera que todos los nodos mantienen una base de datos completa de la ubicación de los demás nodos de la red.

Protocolos eficientes en energía sensibles a umbral: dos protocolos de enrutamiento jerárquico llamados *Threshold-Sensitive Energy Efficient Sensor Network Protocol* (TEEN) y *Adaptive Periodic TEEN* (APTEEN) se proponen en [62, 61].

Estos protocolos se utilizan para aplicaciones de tiempo crítico. En TEEN, los nodos sensores detectan continuamente el medio, pero la transmisión de datos se realiza con menos frecuencia. Un sensor CH envía a sus miembros un umbral duro, que es el valor umbral del atributo detectado, y un umbral blando, que es un pequeño cambio en el valor del atributo detectado que provoca que el nodo encienda su transmisor y transmita. Por lo tanto, el umbral duro intenta reducir el número de transmisiones permitiendo a los nodos transmitir solamente cuando el atributo detectado está en el rango de interés. El umbral blando reduce aún más el número de transmisiones que pueden producirse cuando no hay cambio o éste es muy bajo en el atributo sensado. Un valor más pequeño del umbral blando da una imagen más exacta de la red, a expensas de incrementar el consumo de energía. Así, el usuario puede controlar el compromiso entre eficiencia energética y precisión de los datos. Cuando los CHs se cambian, los nuevos valores

de los parámetros anteriores se emiten broadcast. El principal inconveniente de este sistema es que si los umbrales no se reciben, los nodos no se comunicarán nunca, y el usuario no obtendrá ningún dato de la red.

Los nodos detectan su entorno continuamente. La primera vez que un parámetro de un conjunto de atributos llega al valor de umbral duro, el nodo conmuta su transmisor y envía los datos sensados. El valor se almacena en una variable interna llamada valor sensado (SV). Los nodos transmitirán datos en el período de grupo actual sólo cuando se cumplen las siguientes condiciones:

- El valor actual del atributo detectado es mayor que el umbral duro.
- El valor actual del atributo sensado difiere de SV una cantidad igual o superior al umbral blando.

En el protocolo TEEN la transmisión de mensajes consume más energía que la detección de datos, por lo que el consumo de energía en este sistema es menor que en las redes proactivas. Además, si es necesario, el usuario puede modificar el umbral blando y emitir broadcast los parámetros nuevos.

APTEEN, por otra parte, es un protocolo híbrido que modifica la periodicidad o los valores umbrales utilizados en el protocolo TEEN de acuerdo a las necesidades del usuario y al tipo de aplicación. En APTEEN, el nodo detecta continuamente el entorno, y los nodos que detectan un valor del atributo por encima del umbral duro transmiten los datos cuando dicho valor cambia en una cantidad igual o superior a al umbral blando. Si un nodo no envía datos durante un período de tiempo igual al contador de tiempo (máximo período de tiempo entre dos informes sucesivos enviados por un nodo), se ve obligado a detectar y retransmitir los datos. Se utiliza un programa TDMA, y a cada uno de los nodos del clúster se le asigna una ranura de transmisión. Por lo tanto, APTEEN utiliza un programa modificado TDMA para implementar la red híbrida. Las principales características del esquema APTEEN incluyen lo siguiente. Combina ambas políticas proactivas y reactivas. Ofrece una gran flexibilidad al permitir al usuario ajustar el intervalo contador de tiempo y los valores del umbral para el consumo de energía. La principal desventaja del sistema es la complejidad adicional para implementar las funciones del umbral y del contador de tiempo. La simulación de TEEN y APTEEN ha demostrado que estos dos protocolos superan a LEACH. El rendimiento de APTEEN está entre LEACH y TEEN en cuanto a la disipación de energía y tiempo de vida de la red. TEEN da los mejores rendimientos, ya que

disminuye el número de transmisiones. Los principales inconvenientes de los dos enfoques son la sobrecarga y la complejidad asociada con la formación de grupos en varios niveles, el método de implementación de funciones basadas en umbral, y cómo tratar las consultas de nombres basados en atributos.

Red de comunicación de mínima energía: en [78] se propone un protocolo que calcula la eficiencia energética de una subred, la red de comunicación de mínima energía (*Minimum Energy Communication Network*(MECN)), para una determinada red de sensores utilizando GPS de baja potencia. MECN identifica una región de retransmisión para cada nodo. La región de retransmisión consta de nodos en un área circundante donde la transmisión a través de estos nodos es más eficiente energéticamente que la transmisión directa. El recinto de un nodo es creado mediante la unión de todas las regiones de retransmisión que dicho nodo puede alcanzar. La idea principal de MECN es encontrar una subred que tendrá menos nodos y requerirá menos energía para la transmisión entre dos nodos particulares. De esta manera, las rutas de potencia mínima global se encuentran sin tener en cuenta todos los nodos de la red. Esto se realiza utilizando una búsqueda localizada para cada nodo considerando su región de retransmisión. MECN es autoreconfigurable y, por tanto, puede adaptarse dinámicamente a un fallo de un nodo o al despliegue de nuevos sensores. SMECN (*Small Minimum Energy Communication Network*) [51] es una extensión de MECN. En MECN, se supone que cada nodo puede transmitir a cualquier otro nodo, cosa que no es siempre posible. SMECN tiene en cuenta los posibles obstáculos que pueden aparecer entre cualquier par de nodos. Además, también es posible reducir el número de dispositivos incluidos en la subred de transmisión respecto al protocolo MECN.

Protocolo de auto-organización: Subramanian [90] describe un protocolo de auto-organización que es usado para construir una arquitectura para dar soporte a los sensores heterogéneos. Además, estos sensores pueden ser móviles o estacionarios. Algunos sensores exploran el medio ambiente y transmiten los datos a un conjunto de nodos designados que actúan como enrutadores. Los nodos encaminadores son estacionarios y forman la red troncal de comunicación. Los datos recogidos son enviados a través de los enrutadores a los nodos con más potencia. Cada nodo de detección debe ser capaz de llegar a un encaminador para ser parte de la red. Los nodos sensores se identifican a través de la dirección del nodo enrutador al que están conectados. La arquitectura de enrutamiento es jerárquica donde los grupos de nodos se forman y agrupan cuando sea necesario. El algoritmo de bucles locales de Markov (LML), que realiza un recorrido aleatorio en árboles de expansión, se utilizó para apoyar la tolerancia a fallos y como un medio

de difusión. En este enfoque, los nodos sensores pueden ser direccionados individualmente en la arquitectura de enrutamiento, por lo que es adecuado para aplicaciones donde la comunicación con un nodo es necesaria. Además, este algoritmo incurre en un pequeño coste para el mantenimiento de las tablas de enrutamiento y de una jerarquía de enrutamiento equilibrada. También se constató que la energía consumida para el envío broadcasting de un mensaje es inferior a la que se consume en el protocolo SPIN. Este protocolo, sin embargo, no es un protocolo bajo demanda, especialmente en la fase de organización del algoritmo, y por tanto introduce sobrecarga extra. Otra cuestión está relacionada con la formación de la jerarquía. Podría ocurrir que haya muchos cortes en la red y, por tanto, la probabilidad de aplicar la fase de reorganización incrementase, siendo ésta una operación costosa.

Encaminamiento de agregación de sensores: en [33] se propone un conjunto de algoritmos para la construcción y el mantenimiento de la agregación de sensores. El objetivo es controlar colectivamente una cierta actividad en el entorno (aplicaciones de seguimientos de objetivos). Una agregación de sensores comprende aquellos nodos en una red que satisfacen un predicado de agrupación para una tarea de procesamiento colaborativo. Los parámetros del predicado dependen de la tarea y de los requisitos de recursos. Los sensores en un campo de sensores se dividen en grupos de acuerdo a la intensidad de la señal sensada, por lo tanto sólo hay un máximo por grupo. A continuación, se eligen los líderes de las agrupaciones locales. Para elegir un líder, es necesario el intercambio de información entre sensores vecinos. Si un sensor, tras el intercambio de paquetes con todos sus vecinos de un salto, considera que es superior a todos ellos, se declara como líder. Este algoritmo de seguimiento basado en líder asume que el líder conoce la región geográfica de la colaboración.

Tres algoritmos fueron propuestos en [33]. Primero fue un protocolo ligero, Gestión de Agregados Distribuidos (*Distributed Aggregate Management* (DAM)), utilizado en la formación de agregados de sensores para una tarea de monitorización de objetivos. El protocolo incluye un predicado de decisión para que cada nodo pueda decidir si debe participar en una agregación. Un nodo determina si pertenece a una agregación basándose en el resultado de aplicar el predicado a los datos del nodo así como a la información de otros nodos. Los agregados se forman cuando el proceso converge finalmente. En segundo lugar, Monitorización de la Actividad Basada en Energía (*Energy-Based Activity Monitoring* (EBAM)), que estima el nivel de energía en cada uno de los nodos computando el área de impacto de la señal, combinando de forma ponderada la energía del objetivo detectado en cada sensor, asumiendo que cada objetivo tiene un nivel

de energía constante. El tercer algoritmo, Monitorización de la Actividad con Maximización de Expectativa (*Expectation-Maximization Like Activity Monitoring* (EMLAM)), elimina la suposición del nivel de energía constante del objetivo. EMLAM estima las posiciones del objetivo y la energía de la señal utilizando las señales recibidas, y utiliza las estimaciones resultantes para predecir cómo las señales de los objetivos pueden ser mezcladas en cada uno de los sensores. Este proceso se itera hasta que la estimación es suficientemente buena.

Encaminamiento en una arquitectura grid virtual: un paradigma de enrutamiento eficiente en energía se propone en [5], dicho protocolo utiliza la agregación de datos y el procesamiento de la red para maximizar el tiempo de vida de la red. Debido a que en muchas aplicaciones en redes de sensores inalámbricos los nodos son estacionarios o de baja movilidad, un enfoque razonable es organizar los nodos en una topología fija [95]. Un enfoque GPS-libre [83] se utiliza para crear grupos con formas simétricas que están fijados, iguales, adyacentes y no solapados. En [5], las agrupaciones cuadradas fueron utilizadas para obtener una topología virtual rectilínea. Dentro de cada zona, un nodo es seleccionado de forma óptima para actuar como CH. La agregación de datos se realiza a dos niveles: local y luego global. El conjunto de CHs, también llamados agregadores locales (LAs), realizan la agregación local, mientras que un subconjunto de estos LAs se utilizan para realizar la agregación global. Sin embargo, la determinación de una selección óptima de los puntos de agregación globales, llamados maestros agregadores (MAs), es NP-duro. La arquitectura de grid virtual se puede apreciar en la Figura 3.3.

Encaminamiento jerárquico de ahorro de energía: el protocolo divide la red en grupos de sensores [52]. Cada grupo de sensores de proximidad geográfica se agrupan juntos como una zona, y cada zona es tratada como una entidad. Para realizar el encaminamiento, a cada zona se le permite decidir cómo se enrutará un mensaje a través de las otras zonas, de tal manera que se maximiza la vida de la batería de los nodos del sistema. Los mensajes son encaminados a lo largo de la ruta que tiene el máximo sobre la totalidad de los mínimos de la potencia restante, llamado ruta max-min. La motivación de esta idea se base en que el uso de nodos de alta potencia residual puede ser más caro que el camino con el mínimo consumo de energía. Un algoritmo de aproximación, llamado algoritmo máx-min zPmin, se ha propuesto en [52]. El quid del algoritmo se basa en la relación entre minimizar el consumo de potencia total y maximizar la mínima potencia residual de la red. Por lo tanto, el algoritmo intenta mejorar una ruta max-min limitando su consumo de potencia. En primer lugar, el algoritmo encuentra la

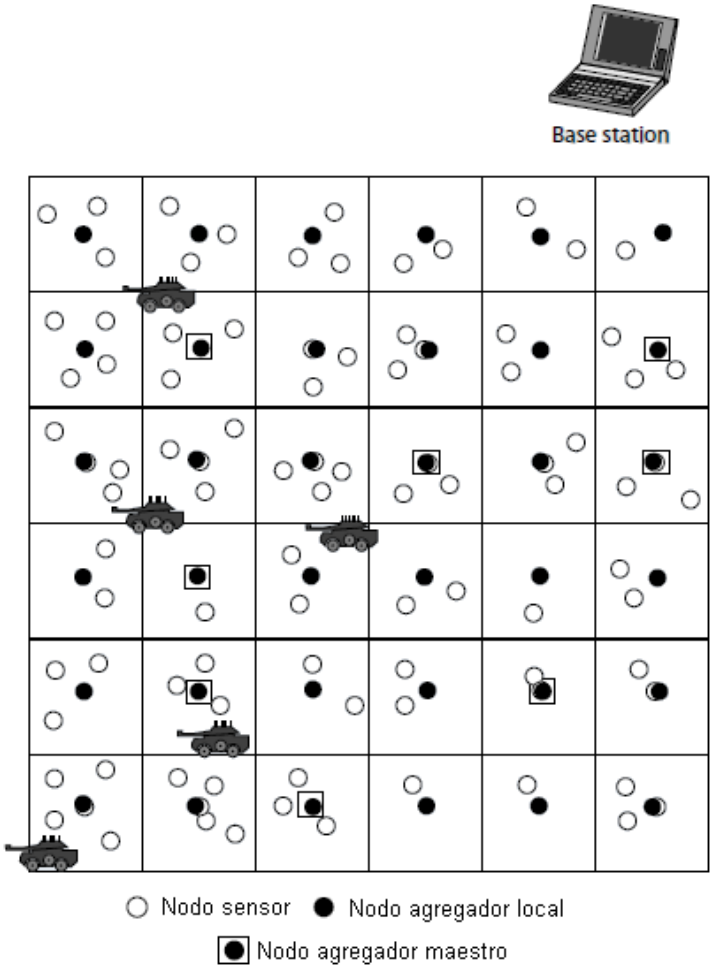


Figura 3.3: *Arquitectura de grid virtual.*

ruta con el menor consumo de potencia (Pmin) usando el algoritmo de Dijkstra. En segundo lugar, el algoritmo encuentra un camino que maximiza la potencia mínima residual en la red. El algoritmo propuesto trata de optimizar la solución de los dos criterios.

Diseminación de datos en dos niveles: un enfoque en [98] denominado Diseminación de Datos de Dos Niveles (*Two-Tier Data Dissemination* (TTDD)) proporciona la entrega de datos a varias estaciones base móviles. Cada fuente de datos construye de manera proactiva una estructura de red que se utiliza para difundir los datos a los sumideros móviles, suponiendo que los nodos sensores son estacionarios. Una vez que ocurra un suceso, los sensores que lo rodean procesan la señal, y uno de ellos se convierte en la fuente para generar informes de datos. Para construir la estructura grid, una fuente de datos se elige a sí misma como punto de cruce del inicio de la red, y envía un mensaje de anuncio de datos a cada uno de sus cuatro puntos de

cruce adyacentes usando transmisión geográfica por gradiente simple.

El mensaje se detiene cuando llega al nodo más cercano al punto de cruce (especificado en el mensaje). Durante este proceso, cada nodo intermedio almacena la información de la fuente y envía el mensaje a sus puntos de cruce adyacentes excepto desde el que viene el mensaje. Este proceso continúa hasta que el mensaje se detiene en el borde de la red. Los nodos que almacenan la información de la fuente son elegidos como puntos de difusión. Después de este proceso, se obtiene la estructura grid. Usando el grid, un estación base puede inundar una consulta, que será enviada al punto de difusión más cercano para recibir datos. A continuación, la consulta se remite a lo largo de otros puntos de difusión hacia la fuente. Los datos solicitados fluyen en el camino inverso al sumidero. Aunque TTDD es un enfoque de encaminamiento eficiente, hay algunas preocupaciones acerca de cómo obtiene la información de localización, dicha información es necesaria para establecer la estructura grid. La longitud de una ruta de transmisión en TTDD es mayor que la longitud del camino más corto. Los autores de TTDD creen que la longitud subóptima de la ruta merece la pena debido a la ganancia en escalabilidad. La comparación de resultados entre TTDD y difusión dirigida muestran que TTDD puede lograr aumentar el tiempo de vida de la red y disminuir el retardo en la entrega de datos. Sin embargo, la sobrecarga asociada con el mantenimiento y los cálculos del grid, como los cambios de topología de red, pueden ser altos. Además, TTDD supone que se tiene la disponibilidad de un sistema de posicionamiento muy preciso que todavía no está disponible para redes de sensores inalámbricos.

3.2.1.3. Encaminamiento basado en localización

En este tipo de encaminamiento, los nodos sensores son identificados por medio de sus localizaciones. La distancia entre nodos vecinos puede estimarse en base a los niveles de la señal de entrada. Las coordenadas relativas de los nodos vecinos se pueden obtener mediante el intercambio de información entre los vecinos [13, 83, 15]. Alternativamente, la ubicación de los nodos puede estar disponible directamente mediante la comunicación con un satélite utilizando GPS si los nodos están equipados con un receptor GPS de baja potencia [95]. Para ahorrar energía, algunos sistemas basados en ubicación mantienen los nodos dormidos si no hay actividad. El problema estará a la hora de diseñar los planificadores de periodos de dormido de manera localizada [20, 95].

Fidelidad Adaptativa Geográfica: este algoritmo de enrutamiento denominado Fidelidad

Adaptativa Geográfica (*Geographic Adaptive Fidelity* (GAF)) se basa en localización y está diseñado principalmente para redes ad hoc móviles, pero también puede ser aplicable a redes de sensores [95]. El área de red está dividida en zonas fijas formando un grid virtual. Dentro de cada zona, los nodos colaboran entre sí para desempeñar diferentes funciones. Por ejemplo, los nodos elegirán un nodo sensor que permanecerá despierto durante un determinado período de tiempo y, a continuación, el resto se duermen. Este nodo es el responsable de monitorizar y notificar los datos a la estación base en nombre de los nodos de la zona. Por lo tanto, GAF conserva la energía apagando los nodos innecesarios en la red sin afectar el nivel de fidelidad del enrutamiento. Cada nodo utiliza su localización GPS para asociarse a un punto en el grid virtual. Los nodos asociados con el mismo punto del grid son considerados equivalentes en términos del coste de enrutado de paquetes. Dicha equivalencia se explota mediante el uso de períodos de dormido en algunos nodos del mismo punto del grid con el fin de ahorrar energía. Por lo tanto, GAF puede aumentar sustancialmente el tiempo de vida de la red así como el número de nodos. Hay tres estados definidos en el GAF: descubrimiento, para la determinar los vecinos en el grid, activo, lo que refleja la participación en el enrutamiento, y dormido, cuando la radio está apagada. Para gestionar la movilidad, cada nodo en la red estima el tiempo de salida de la red y lo envía a sus vecinos. Los vecinos dormidos ajustan el tiempo de dormido para mantener la fidelidad del enrutamiento. Antes de que el tiempo de salida del nodo activo expire, los nodos dormidos se despiertan y uno de ellos se convierte en activo. GAF se aplica tanto para nodos estacionarios (GAF-básico) como móviles (GAF-adaptado a movilidad). El rendimiento de GAF es similar al de un protocolo de enrutamiento ad-hoc en términos de latencia y paquetes perdidos, pero incrementa el tiempo de vida de la red ahorrando energía. Aunque GAF es un protocolo basado en localización, también puede ser considerado como un protocolo jerárquico, donde los grupos se basan en la localización geográfica. Para cada área grid particular, un nodo perteneciente a ella actúa como el líder para transmitir los datos a otros nodos.

Encaminamiento geográfico basado en ahorro de energía: el protocolo *Geographic and Energy Aware Routing* (GEAR) [99] examina el uso de información geográfica mientras se reparten consultas a las regiones apropiadas, ya que las consultas de datos suelen incluir atributos geográficos. Este protocolo usa selección heurística de vecinos, basándose en ahorro de energía y localización geográfica, para encaminar un paquete hacia la región de destino. La idea clave es limitar el número de intereses de la difusión dirigida, considerando solo una región determinada en lugar de enviar los intereses a toda la red. De esta manera, GEAR puede conservar más

energía que la difusión dirigida.

Cada nodo en GEAR mantiene un coste estimado y un coste de aprendizaje para llegar al destino a través de sus vecinos. El coste estimado es una combinación de la energía residual y la distancia al destino. El coste aprendido es un ajuste del coste estimado, que explica el encaminamiento alrededor de los agujeros en la red. Un agujero se produce cuando un nodo no tiene ningún vecino cercano en la región ocupada por él. Si no hay agujeros, el coste estimado es igual al coste aprendido. El coste aprendido se propaga un salto hacia atrás cada vez que un paquete llega al destino, de forma se ajusta la configuración de la ruta para el próximo paquete. Hay dos fases en el algoritmo:

- Reenvío de paquetes hacia la región objetivo: al recibir un paquete, un nodo comprueba si alguno de sus vecinos está más cerca de la región objetivo que él mismo. Si hay más de uno, se selecciona para el próximo salto al vecino más cercano a la región objetivo. Si están todos lejos del propio nodo, significa que hay un agujero. En este caso, se selecciona uno de los vecinos basándose en la función de coste aprendido. Esta elección puede ser actualizada de acuerdo con la convergencia del coste aprendido durante la entrega de paquetes.

- Reenvío de los paquetes dentro de la región: si el paquete ha llegado a la región, puede ser difundido en dicha región, ya sea por transmisión geográfica recursiva o inundaciones restringidas. La inundación restringida es buena cuando los sensores no están desplegados densamente. En redes de alta densidad, la transmisión geográfica recursiva es más eficiente energéticamente que la inundación restringida. En ese caso, la región se divide en cuatro subregiones y se crean cuatro copias del paquete.

Most Forward within Radious (MFR), Compass Routing Method (DIR) y Geographic Distance Routing (GEDIR): Stojmenovic y Lin [89] describen y discuten los algoritmos de enrutamiento localizados básicos. Estos protocolos implementan métodos básicos basados en distancias, progreso y dirección. Un nodo fuente o nodo intermedio seleccionará uno de sus vecinos de acuerdo a un cierto criterio. Los métodos de enrutamiento que pertenecen a esta categoría son *Most Forward within Radius* (MFR), *Geographic Distance Routing* (GEDIR) que es una variante de los algoritmos de gradiente, método de gradiente de dos saltos y método de gradiente alternativo, y DIR (método de enrutamiento brújula). GEDIR es un algoritmo de gradiente que siempre mueve el paquete al vecino cuya distancia al destino sea mínima. El

algoritmo falla cuando el paquete atraviesa el mismo borde dos veces sucesivas. En la mayoría de los casos, los métodos MFR y de gradiente tienen la misma ruta hasta el destino. Para DIR, el mejor vecino es aquel que se encuentra en la dirección más próxima al destino [89].

Un estudio comparativo [89] entre estos algoritmos muestra que los tres algoritmos son comparables en términos de tasa de entrega y retardo promedio. Además, los nodos en los métodos MFR y de gradiente seleccionan los mismos vecinos para transmitir en más de 99 por ciento de los casos, y todos los caminos seleccionados son idénticos en la mayoría de los casos.

Greedy Other Adaptive Face Routing (GOAFR): algoritmo de enrutamiento geométrico combinando encaminamiento por gradiente y por caras [47]. El algoritmo de gradiente GOAFR siempre escoge para el enrutamiento al vecino más cercano al nodo siguiente. Sin embargo, puede quedar fácilmente atrapado en algún mínimo local (es decir, ningún vecino está más cerca de un nodo que el nodo actual). Other Face Routing (OFR) es una variante de Face Routing (FR). El algoritmo FR [89] es el primero que garantiza el éxito si la fuente y el destino están conectados. Sin embargo, el peor caso de coste de FR es proporcional al tamaño de la red en términos de número de nodos. OFR utiliza la estructura de cara de gráficos planos de tal manera que los mensajes se encaminan del nodo s al nodo t atravesando una serie de fronteras de cara. El objetivo es encontrar el mejor nodo de la frontera (es decir, el nodo más cercano al destino t) mediante el uso de planos geométricos. Cuando haya terminado, el algoritmo devuelve a s el mejor nodo de la frontera. El algoritmo de gradiente simple se comporta bien en redes densas, pero falla para configuraciones muy sencillas [47].

SPAN: algoritmo basado en posición [20] que selecciona algunos nodos como coordinadores dependiendo de sus posiciones. Los coordinadores forman una red troncal utilizada para reenviar los mensajes. Un nodo debe convertirse en un coordinador si dos vecinos, que no sean nodos coordinadores, no pueden llegar directamente entre sí o a través de uno o dos coordinadores (acceso mediante tres saltos). Los coordinadores nuevos y los ya existentes no tienen porque ser necesariamente vecinos, lo que hace que el diseño sea menos eficiente en cuanto a energía debido a la necesidad de mantener la distancia de dos o tres saltos a los vecinos.

3.2.2. Protocolos basados en el criterio de encaminamiento

En esta sección se revisan los protocolos de enrutamiento con diferentes funcionalidades. Cabe señalar que algunos de estos protocolos pueden encontrarse dentro de una o varias de las

categorías anteriores de enrutamiento.

Protocolos de enrutamiento multicamino: estos protocolos utilizan varias rutas en lugar de una sola ruta con el fin de mejorar el rendimiento de la red. La tolerancia a fallos de un protocolo se mide por la probabilidad de que exista un camino alternativo entre una fuente y un destino, cuando la principal ruta falla. Esto puede ser incrementado por el mantenimiento de varias rutas entre el origen y el destino, a expensas de un mayor consumo de energía y generación de tráfico. Estas rutas alternativas se mantienen vivas mediante el envío periódico de mensajes. Por lo tanto, la fiabilidad de la red se puede aumentar a cambio de un aumento de la sobrecarga debido al mantenimiento de la rutas alternativas.

El algoritmo [19] enruta datos a través de un camino cuyos nodos tienen la mayor energía residual. El camino se cambia cada vez que se encuentra un camino mejor. La ruta principal será utilizada hasta que su energía esté por debajo de la energía de la ruta de *backup*, en ese momento se utiliza dicha ruta. Usando este enfoque, los nodos de la ruta principal no agotan sus recursos de energía a través del uso continuo de la misma ruta, por lo tanto, se logra una vida más larga.

Los autores de [76] propusieron el uso de un conjunto de caminos subóptimos que incrementan el tiempo de vida de la red. Estos caminos son elegidos por medio de una probabilidad que depende de cuánto baja el consumo de energía de cada ruta.

El camino con la mayor energía residual puede ser demasiado caro energéticamente cuando se utiliza para encaminar los datos en una red, así que hay un compromiso entre minimizar la potencia total consumida y la energía residual de la red. Los autores en [52] proponen un algoritmo en el que la energía residual de la ruta se reduce un poco con el fin de seleccionar una ruta más eficiente energéticamente.

En [29], el enrutamiento multiruta se utiliza para mejorar la fiabilidad de las redes de sensores inalámbricos. Una forma de incrementar la fiabilidad de la red puede ser mediante la creación de varios caminos desde el origen al destino y el envío de los mismos paquetes en cada ruta. Sin embargo, con el uso de esta técnica el tráfico se incrementará significativamente. Por lo tanto, existe un compromiso entre la cantidad de tráfico y la fiabilidad de la red. Este compromiso se estudia [29] usando una función de redundancia que depende del grado de multicamino y las probabilidades de fallo de los caminos disponibles. La idea es dividir el paquete de datos original en subpaquetes y, a continuación, enviar cada subpaquete a través de uno de los caminos disponibles. Se ha comprobado que, incluso si algunos de estos subpaquetes se pierden, el mensaje

original todavía puede ser reconstruido.

La difusión dirigida [42] es un buen candidato para la robustez del encaminamiento multicamino. Se ha encontrado que el uso de encaminamiento multitrayecto proporciona una alternativa eficiente en energía para la recuperación ante fallos. La motivación para el uso de estos caminos es mantener bajo el coste del mantenimiento de multicaminos. Los costes de las rutas alternativas son comparables a los de la ruta principal, ya que tienden a estar muy cerca de la misma.

Encaminamiento basado en petición: en este tipo de encaminamiento, los nodos destinos propagan una consulta de datos (tarea de detección) a un nodo a través de la red, y el nodo con estos datos los envía al nodo que inició la consulta. La difusión dirigida [42] descrita anteriormente es un ejemplo de este tipo de enrutamiento. En difusión dirigida, el nodo estación base envía mensajes de interés a los sensores. Como el interés se propaga en toda la red de sensores, se establecen los gradientes de la fuente a la estación base. Cuando la fuente tiene datos para el interés, los envía a lo largo del camino de gradiente de interés. Para bajar el consumo de energía se realiza la agregación de datos en el camino.

El protocolo de encaminamiento por rumor [11] utiliza un conjunto de agentes de larga vida para crear caminos que se dirigen a los eventos encontrados. Cuando un agente cruza un camino que conduce a un evento que no se ha encontrado aún, crea un camino que conduce a dicho evento. Cuando los agentes van a través de caminos más cortos o más eficientes, optimizan las rutas en las tablas de enrutamiento. Cada nodo mantiene una lista de sus vecinos y una tabla de eventos que se actualiza cada vez que se encuentran nuevos eventos. Cada agente contiene una tabla de eventos que se sincroniza con cada nodo que visita. El agente tiene un tiempo de vida de un determinado número de saltos, después del cual muere. Un nodo no generará una consulta a menos que aprenda una ruta hacia el evento requerido. Si no hay una ruta disponible, el nodo transmite una consulta en una dirección aleatoria. A continuación, el nodo espera una cierta cantidad de tiempo para saber si la consulta alcanzó el destino. En caso de no recibir una respuesta del destino, los nodos inundan la red.

Enrutamiento basado en negociación: estos protocolos utilizan descriptores de datos de alto nivel con el fin de eliminar, mediante un negociación, las transmisiones de datos redundantes. Las decisiones de comunicación también se basan en los recursos disponibles para ellas. Los protocolos de la familia SPIN [40] y los protocolos en [48] son ejemplos de protocolos de enrutamiento basados en negociación. La motivación es que el uso de las inundaciones para difundir datos producirá la implosión y la superposición de los datos enviados, por lo que los

nodos recibirán copias duplicadas de los mismos datos. Esta operación consume más energía y procesado debido al envío de los mismos datos por diferentes sensores. Los protocolos SPIN están diseñados para difundir los datos de un sensor a todos los demás sensores, asumiendo que estos sensores son posibles estaciones base. Por lo tanto, la idea principal del enrutamiento basado en negociación es suprimir la información duplicada e impedir que se envíen datos redundantes al siguiente sensor mediante la realización de una serie de mensajes de negociación antes de que se inicie la verdadera transmisión de datos.

Encaminamiento basado en calidad de servicio: usando estos protocolos la red tiene un equilibrio entre el consumo de energía y la calidad de los datos. En particular, la red ha de satisfacer determinadas métricas de *QoS* (retardo, energía, ancho de banda, etc) al entregar los datos a la estación base.

El Enrutamiento por Asignación Secuencial (*Sequential Assignment Routing* (SAR)) [87] es uno de los primeros protocolos de enrutamiento para redes de sensores inalámbricos que introduce la noción de calidad de servicio (*QoS*) en las decisiones de enrutamiento. Una decisión de enrutamiento en SAR depende de tres factores: los recursos energéticos, *QoS* en cada ruta y el nivel de prioridad de cada paquete. Para evitar el fallo en una ruta, se utiliza un enfoque multicamino y sistemas de restauración de caminos localizados. Para crear varias rutas desde un nodo fuente, se construye un árbol que cuelga del nodo fuente al nodo destino. Los caminos del árbol son construidos mientras se evitan nodos de baja energía o baja garantia de *QoS*. Al final de este proceso, cada nodo sensor será parte de un árbol multitrayecto. Como tal, SAR es un protocolo multitrayecto dirigido por tablas, que tiene como objetivo lograr la eficiencia energética y la tolerancia a fallos. En esencia, SAR calcula una métrica *QoS* ponderada como el producto de métricas de *QoS* adicionales y un coeficiente asociado con el nivel de prioridad del paquete. El objetivo de SAR es reducir al mínimo el promedio ponderado de la métrica de *QoS* a través del tiempo de vida de la red. Si la topología cambia debidos a fallos de nodo, se recalcula el camino. Como medida preventiva, la estación base obliga a un recálculo periódico de caminos por si ha habido cambios en la topología. La recuperación de un fallo se realiza imponiendo consistencia en la tabla de rutas entre los nodos de cada ruta. SAR ofrece menos consumo de potencia que el algoritmo métrico de mínima de energía, que se centra sólo en el consumo de energía de cada paquete sin tener en cuenta su prioridad. SAR mantiene varias rutas de los nodos a la estación base. Si bien, esto garantiza la tolerancia a fallos y facilidad de recuperación, el protocolo se ve afectado por la sobrecarga de mantenimiento de las tablas y los

estados en cada nodo sensor, sobre todo cuando el número de nodos es enorme.

Otro protocolo de enrutamiento, basado en *QoS*, llamado SPEED se introdujo en [38]. Dicho protocolo proporciona garantía extremo a extremo a las transmisiones en tiempo real. El protocolo exige que cada uno de los nodos mantenga la información acerca de sus vecinos y utiliza el reenvío geográfico para encontrar los caminos. Además, se esfuerza para asegurar una cierta velocidad para los paquetes que circulan por la red, de forma que las aplicaciones pueden estimar el retardo extremo a extremo de los paquetes dividiendo la distancia al nodo destino entre la velocidad del paquete antes de llevar a cabo la admisión de dicha transmisión. Como novedad sobre otros sistemas de encaminamiento, SPEED también consta de métodos para combatir la congestión de la red. Mirando los valores de retardo, se selecciona el nodo que cumpla con los requisitos de velocidad. Debido a la simplicidad del algoritmo de enrutamiento, se minimiza la energía de la transmisión total y la sobrecarga de paquetes de control.

Enrutamiento basado en procesamiento de datos coherentes y no coherentes: el procesamiento de datos es un componente importante en el funcionamiento de redes inalámbricas de sensores. Por lo tanto, las técnicas de enrutamiento emplean diferentes técnicas de procesamiento de datos. En general, los nodos sensores cooperan entre sí en el procesamiento de datos. Dos ejemplos de técnicas de procesamiento de datos son el enrutamiento basado en procesamiento de datos coherente y no coherente [87]. En el encaminamiento procesando datos no coherentes, los nodos procesan localmente los datos antes de que sean enviados a otros nodos para su posterior procesamiento. Los nodos que realizan el procesado se denominan agregadores. En el enrutamiento coherente, los datos se enviarán a los agregadores después de un procesamiento mínimo. El procesamiento mínimo típicamente incluye tareas como creación de marcas de tiempo y supresión de duplicados. El procesamiento coherente se selecciona normalmente para realizar un enrutamiento eficiente energéticamente.

Las funciones no coherentes tienen baja carga de tráfico de datos. Por otro lado, el procesamiento coherente genera grandes flujos de datos y la eficiencia energética se alcanza por la elección de una ruta óptima.

3.3. Protocolos de encaminamiento en redes ad-hoc

En estas redes los protocolos se clasifican atendiendo al descubrimiento de la ruta hacia el nodo destino. En el punto anterior se dio una breve noción sobre los protocolos proactivos,

reactivos e híbridos, ahora se va a extender la explicación:

- Protocolos proactivos: todas las rutas son calculadas antes de que se necesiten. 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 respuesta rápida ante solicitudes de ruta y ofrece un buen comportamiento en situaciones donde la tasa de movilidad es baja. Sin embargo, la sobrecarga que se introduce en la red con información de control es alta.

- Protocolos reactivos: las rutas se descubren y se establecen bajo peticiones, con el consumo de energía que lleva asociado. A diferencia de los protocolos proactivos, los reactivos buscan cómo llegar al nodo destino cuando quieren iniciar una comunicación (esto es lo que les ha llevado a conocerse como protocolos bajo demanda). Para cada comunicación entre un nodo fuente y un nodo destino van descubriendo la ruta necesitada. Estos algoritmos optimizan los recursos evitando el envío de paquetes de forma innecesaria. Como contrapartida, sufren una pérdida de tiempo cada vez que realizan el descubrimiento de la ruta.

- Protocolos híbridos: usan una combinación de los dos tipos anteriores. Se mantiene una filosofía proactiva en un ámbito local y reactiva a nivel más global.

Dentro de los protocolos proactivos se pueden destacar los siguientes: **DSDV** (*Destination Sequence Distance Vector*): protocolo diseñado por Charles E. Perkins y Pravin Bhagwat [17]. Cada nodo de la red mantiene una tabla de encaminamiento que contiene todos los posibles destinos y el número de saltos que daría un paquete que viajara hacia el destino especificado. Cada entrada posee un número de secuencia asignado por el nodo destino. Los números de secuencia permiten distinguir las rutas antiguas de las rutas modernas. Continuamente se deben enviar mensajes de actualización a través de la red para mantener la consistencia de las tablas. Para ayudar a minimizar la gran cantidad de tráfico que ocasionan estas actualizaciones, se utilizan dos tipos de paquetes. El primero recibe el nombre de *full dump*. Este paquete transporta toda la información disponible sobre el encaminamiento y puede requerir que su envío se divida en varias unidades más pequeñas. Cuando los cambios en la red son pequeños, es raro que se use este tipo de paquete. En cambio, hay un segundo tipo que solo contiene la información que ha variado desde el último *full dump*. Este paquete se llama *incremental*. Los nodos disponen de una tabla adicional donde guardan los datos recibidos por los paquetes *incremental*. Las nuevas

rutas contienen la dirección de destino, el número de saltos requeridos para alcanzar al destino, el número de secuencia asociado al destino y un nuevo número que identifica todo el mensaje. En el caso de que haya dos rutas distintas hacia un destino, se usará la que contenga el número de secuencia más moderno. Además, si ambos números coincidieran, la ruta con menor número de saltos sería la que se usaría. En general, DSDV es un protocolo aceptable en escenarios en los que todos los nodos intervienen en las comunicaciones y en los que la movilidad es media.

WRP (*Wireless Routing Protocol*): creado por S. Murthy y J.J. García-Luna-Aceves [80]. Protocolo basado en tablas cuyo objetivo principal es mantener información actualizada de todos los nodos de la red. Cada nodo es responsable de mantener cuatro tablas: tabla de distancias, tabla de encaminamiento, tabla de coste de ruta y tabla con la lista de mensajes retransmitidos (MRL).

La tabla MRL se utiliza para gestionar el envío de los paquetes de actualización de rutas. Cada entrada de la MRL contiene el número de secuencia que identifica el paquete de actualización de rutas, un contador de retransmisiones, un vector de asentimientos con una entrada por vecino y una lista de las unidades enviadas en el paquete de actualización (en ocasiones el paquete se trocea en unidades más pequeñas). La tabla MRL almacena qué unidades deben ser retransmitidas y qué vecinos deben asentir todavía los envíos. Los nodos se informan entre ellos de los cambios en las rutas a través de los paquetes de actualización. Estos paquetes son enviados entre vecinos y contienen los elementos a actualizar en las rutas. Los nodos envían estos paquetes cuando procesan las actualizaciones recibidas de otros vecinos o cuando ellos mismos detectan un cambio en el enlace con algún vecino. Además, mantienen el enlace activo con los vecinos, siempre y cuando reciban asentimientos u otros mensajes de ellos. Si un nodo no está enviando mensajes, debería enviar un paquete *HELLO* cada cierto tiempo a sus vecinos, para que éstos no creyeran que el nodo se había vuelto inalcanzable. Por consiguiente, la omisión de mensajes por parte de un nodo ocasionará la ruptura de ese enlace. Cuando un nodo recibe un mensaje HELLO de un nuevo nodo, éste nodo será añadido a la tabla de encaminamiento y una copia de esta tabla será enviada al nuevo nodo.

Respecto a los protocolos reactivos se pueden citar los siguientes: **AODV** (Ad Hoc On-Demand Distance Vector): creado por Charles E. Perkins [16] como evolución de su anterior protocolo (DSDV). Mantuvo la idea de mantener números de secuencia y tablas de encaminamiento pero añadió el concepto de encaminamiento bajo demanda, es decir, solo se guarda información de los nodos que intervengan en la transmisión de datos. La optimización primor-

dial que se consiguió en relación a su anterior diseño fue el decremento del tiempo de proceso, disminución del gasto de memoria y reducción del tráfico de control por la red. En el Capítulo 5 se explicará con detalle este protocolo.

DSR (Dynamic Source Routing): creado por David B. Johnson y Josh Broch [26]: algoritmo basado en el concepto de encaminamiento en origen. Los nodos mantienen cachés, cuyas entradas incluyen el destino y la lista de nodos para llegar a él. Las entradas de esta tabla son actualizadas según se aprendan rutas nuevas. El protocolo consta de dos mecanismos principales: descubrimiento de ruta y mantenimiento de ruta. Cuando un nodo quiere enviar un paquete a un destino, primero consulta su caché para determinar si dispone de una ruta hacia el destino. Si tiene una ruta válida, la usará para enviar el paquete. Sin embargo, si el nodo no dispone de dicha ruta iniciará un descubrimiento de ruta enviando un paquete RREQ (Route Request). Este paquete contiene la dirección de destino buscada, la dirección del nodo que origina el envío y un identificador único. Cada nodo que reciba el paquete verificará si posee una ruta hacia el destino. Si no la tiene, añadirá su propia dirección en el registro de rutas del paquete y después reenviará el paquete a través de todos sus enlaces. Para limitar la propagación excesiva de descubrimientos de ruta, un nodo solo reenviará este mensaje si la misma petición no fue recibida con anterioridad. Cuando un RREQ alcanza su destino final, este nodo genera una respuesta de ruta (RREP). También podría contestar con un RREP un nodo intermedio que tuviera en su caché una ruta válida hacia el destino del RREQ. Si el nodo que genera la respuesta es el destino, colocará el registro de rutas contenido en el RREQ dentro del RREP. Si es un nodo intermedio el que responde, extraerá de su caché la ruta para llegar al destino, que unida al registro de rutas contenido en el RREQ compondrá la ruta a introducir en el RREP. El mantenimiento de rutas se completa con el uso de paquetes de error en ruta (RERR) y asentimientos. Los paquetes de error en ruta son iniciados por un nodo cuando encuentra un problema en la transmisión con algún enlace. Cuando un RERR es recibido, el nodo que provocó el error es eliminado de la caché de rutas. También serán borradas todas las rutas en las que intervenga el enlace roto. Además de los mensajes de error, se usan asentimientos para verificar que las operaciones con los enlaces son correctas.

TORA (Temporally Ordered Routing Algorithm): creado por M. Scott Corson y Vincent Park [16]. Es un algoritmo de tipo *Link Reversal Routing*, que se basa en mantener un grafo dirigido y sin ciclos para llegar al destino. El objetivo es minimizar la carga sobre la red. La diferencia con otros algoritmos es la imposibilidad de estimar constantemente la distancia hacia

el destino o de mantener siempre la ruta más corta. Sin embargo, tiene la ventaja de que es un algoritmo muy eficiente pues no satura en exceso la red.

El protocolo híbrido por excelencia es el siguiente: **ZRP** (Zone Routing Protocol): creado por Zygmunt J. Haas y Marc R. Pearlman [16]. Es un protocolo híbrido a medio camino entre los reactivos y los proactivos. Es utilizado en una clase particular de redes ad-hoc llamadas RWNs (*Reconfigurable Wireless Networks*). Estas redes se caracterizan por tener gran cantidad de nodos, mucha movilidad y alto tráfico. Los protocolos anteriores no satisfacían las necesidades específicas de estas redes y los autores se decidieron a crear un nuevo protocolo. ZRP usa zonas similares a clústeres, en las que los nodos que actúan de bordes se van seleccionando dinámicamente. Además, el radio de estas zonas se reajusta sobre la marcha según las condiciones de la red. Se pueden usar protocolos distintos para comunicarse dentro de las zonas y entre zonas distintas.

3.4. Direcciones futuras en el encaminamiento en redes de sensores inalámbricos

La visión de futuro para redes de sensores inalámbricos es el uso de numerosos dispositivos distribuidos para controlar e interactuar con los fenómenos del mundo físico, y para aprovechar la densidad espacial y temporal y la capacidad de actuación de los dispositivos de detección. Estos nodos se coordinan entre sí para crear una red que realiza tareas de alto nivel.

A pesar de los grandes esfuerzos que se han ejercido hasta ahora sobre el problema de enrutamiento en redes de sensores inalámbricos, todavía existe la búsqueda de soluciones eficaces para los problemas de enrutamiento. En primer lugar, hay un estrecho vínculo entre los nodos sensores y el mundo físico. Los sensores se insertan en lugares o sistemas desatendidos. En segundo lugar, los sensores presentan limitaciones estrictas de energía, ya que están equipados con pequeñas fuentes de energía finita. En tercer lugar, las comunicaciones son las principales consumidoras de energía en este entorno en el que el envío a poco más de 10 o 100 m consume tanta energía como miles de millones de operaciones [36].

Aunque estos protocolos son prometedores en términos de eficiencia energética, se necesitan más investigaciones para abordar cuestiones tales como *QoS* por vídeo, sensores de imágenes y aplicaciones en tiempo real. El encaminamiento basado en *QoS* con ahorro de energía en redes de sensores asegura ancho de banda (o retardo) proporcionando el uso de la ruta de mayor eficiencia

energética. Otra cuestión interesante para los protocolos de enrutamiento es la consideración de la movilidad del nodo. La mayoría de los protocolos actuales suponen que los nodos sensores son estacionarios.

Sin embargo, podría haber situaciones en las que los sensores deben ser móviles. En tales casos, la frecuente actualización de la posición de cada nodo y la propagación de esa información a través de la red puede aumentar excesivamente el consumo de energía de los nodos. Son necesarios nuevos algoritmos de enrutamiento para manejar la sobrecarga de la movilidad y los cambios en la topología. Las tendencias futuras en las técnicas de enrutamiento en estas redes se centran en direcciones diferentes, pero todas comparten el objetivo común de prolongar la vida útil de red. Algunas de estas direcciones son:

- Aprovechar la redundancia: normalmente, un gran número de nodos sensores se implantan dentro o junto al fenómeno. Los nodos sensores son propensos a fallos, y por tanto, las técnicas de tolerancia a fallos son relevantes para mantener el funcionamiento de la red y el desempeño de sus tareas. Las técnicas de enrutamiento que emplean técnicas de tolerancia a fallos de manera eficaz están todavía bajo investigación [29].

- Arquitecturas estructuradas (combinación de factores de forma/energía): el enrutamiento jerárquico es una técnica antigua para mejorar la escalabilidad y la eficiencia del protocolo de enrutamiento. Sin embargo, otro tema de investigación en estas redes son las nuevas técnicas de agrupamiento de red que maximizan el tiempo de vida de la misma [9].

- Aprovechar la diversidad espacial y la densidad de los nodos sensores/actuadores: los nodos abarcan un área de red que podría ser lo suficientemente grande como para proporcionar comunicación espacial entre ellos. El logro de una comunicación eficiente en energía en este entorno de gran densidad de población merece una investigación más a fondo. El despliegue denso de los nodos sensores debería permitir a la red adaptarse a un entorno impredecible.

- Lograr un comportamiento global deseado con algoritmos adaptativos (es decir, no dependen de información o interacción global): sin embargo, en un entorno dinámico, esto es difícil de modelar [42].

- Aprovechar el procesamiento de datos dentro de la red y explotar el procesamiento de las fuentes de datos cercanas para reducir las comunicaciones (es decir, realizar procesamiento distribuido en la red): las redes de sensores inalámbricos se organizan en torno a los datos.

Debido a que se tiene un gran número de elementos distribuidos, se necesitan algoritmos localizados que logren propiedades del sistema en términos de procesado de datos, antes de que éstos sean enviados a un destino. Los nodos en la red almacenan los datos y hacen que estén disponibles para su procesamiento. Existe una gran necesidad de crear puntos de procesamiento eficaces en la red (por ejemplo, suprimir duplicados, la agregación, correlación de datos). Cómo encontrar de manera eficiente y óptima estos puntos sigue siendo un tema de investigación [5].

- La sincronización de tiempo y ubicación: se requieren técnicas de eficiencia energética para asociar las coordenadas temporales y espaciales con los datos y así soportar procesamiento colaborativo [13].

- Localización: los nodos sensores son desplegados de forma aleatoria en una infraestructura no planificada. El problema de la estimación de las coordenadas espaciales del nodo se refiere a la localización. GPS no se puede utilizar en estas redes ya que sólo puede trabajar al aire libre y no en la presencia de cualquier obstrucción. Además, los receptores GPS son caros y no aptos en la construcción de pequeños nodos sensores baratos. Por lo tanto, hay una necesidad de desarrollar otros medios para establecer un sistema de coordenadas sin depender de una infraestructura ya existente. La mayoría de las técnicas de localización propuestas dependen de técnicas recursivas trilateration/multilateration [12], que no proporcionan suficiente precisión.

- La auto-configuración y reconfiguración son esenciales para el tiempo de vida de los sistemas desatendidos en un entorno dinámico y limitado de energía. Esto es importante para mantener la red en marcha. Como los nodos mueren y dejan la red, deben existir mecanismos de actualización y reconfiguración. Una característica que es importante en cada protocolo de enrutamiento es adaptarse muy rápidamente a los cambios de topología y mantener las funciones de red [40].

- Enrutamiento seguro: los protocolos de enrutamiento actuales optimizan la capacidad reducida de los nodos y la aplicación específica de las redes, pero no consideran la seguridad. A pesar de que estos protocolos no han sido diseñados con la seguridad como un objetivo, es importante analizar sus propiedades de seguridad. Uno de los aspectos de redes de sensores que complica el diseño de un protocolo de enrutamiento seguro es la agregación de

red. El procesamiento de la red hace que los mecanismos de seguridad extremo a extremo sean más difíciles de implementar porque los nodos intermedios necesitan acceso directo a los contenidos de los mensajes [74, 46].

Otras posibles investigaciones para protocolos de enrutamiento incluyen la integración de redes de sensores con las redes de cable (es decir, Internet). La mayoría de las aplicaciones en materia de seguridad y monitorización del entorno requieren que los datos recogidos por nodos sensores sean transmitidos a un servidor y se realice un nuevo análisis. Por otro lado, las solicitudes de los usuarios deben hacerse a través de Internet. Los requerimientos del encaminamiento en cada entorno son diferentes, por ello se necesita una mayor investigación para el manejo de este tipo de situaciones.

Capítulo 4

MAQUETA DE SIMULACIÓN

4.1. Visión general de la maqueta de simulación

La maqueta de simulación implementada en este proyecto tiene como objetivo caracterizar el funcionamiento de protocolos de encaminamiento en redes inalámbricas de sensores ad-hoc. Todos los nodos de estas redes tienen la misma funcionalidad excepto los nodos sumideros, que son los encargados de recibir la información de detección y procesarla.

La razón de implementar una maqueta de simulación se debe a que tras realizar un estudio sobre los simuladores existentes para las redes de sensores inalámbricos, se encontraron múltiples desventajas en ellos. La principal desventaja reside en la dificultad de implementar una estrategia de encaminamiento por parte del usuario. Los simuladores tienen estrategias implementadas, pero intentar crear una nueva conlleva crear un módulo del simulador y compilar todo el código de dicho simulador.

En el entorno de simulación diseñado se utilizan eventos para simular las acciones que llevan a cabo los nodos. Las acciones que realiza un nodo son las de dormido, transmisión y escucha del medio. Simular un protocolo con nodos con ciclo de sueño, escucha y transmisión puede ser complicado, por tiempo de ejecución, con otros programas de simulación.

La idea fundamental es desarrollar una herramienta que pueda ser usada para implementar cualquier tipo de protocolo de encaminamiento en redes de sensores inalámbricas ad-hoc. Para comprobar el funcionamiento de la maqueta, al final del presente capítulo se realizan varias simulaciones sin protocolo de enrutamiento en los nodos sensores. En dichas simulaciones se pueden observar cómo se ven afectados la tasa efectiva de transferencia de información (*throughput*) y la

batería media restante en la red en función de las variaciones de varios parámetros importantes, como puede ser el tiempo de escucha por periodo de cada nodo.

4.2. Valor añadido de la maqueta de simulación

El valor añadido de la maqueta de simulación se refiere a las estructuras de datos y mecanismos de *C++* que han sido utilizados en el proyecto con el fin de mejorar la eficiencia del mismo (disminuir la memoria utilizada y el procesamiento de CPU), y de facilitar el entendimiento del código escrito y la adición de nuevo código. La elección del lenguaje *C++* se debe a que ofrece las siguientes ventajas:

- Difusión: al ser uno de los lenguajes más empleados en la actualidad, posee un gran número de usuarios y existe una gran cantidad de libros, cursos, páginas web, etc. dedicados a él.
- Versatilidad: *C++* es un lenguaje de propósito general, por lo que se puede emplear para resolver cualquier tipo de problema. Permite elaborar desde aplicaciones sencillas, como un «Hello World!», hasta sistemas operativos dependiendo del manejo del lenguaje.
- Portabilidad: el lenguaje está estandarizado y un mismo código fuente se puede compilar en diversas plataformas.
- Eficiencia: *C++* es uno de los lenguajes más rápidos en cuanto ejecución.
- Herramientas: existe una gran cantidad de compiladores, depuradores, librerías, etc.
- Reutilización de código: existen muchos algoritmos cuyo pseudocódigo se encuentra ya desarrollado en *C++*, de manera que se puede adaptar al problema a solucionar.

4.2.1. Árbol AVL

Una estructura de datos utilizada en la implementación de la maqueta de simulación es un árbol AVL. Esta estructura se ha elegido por su eficiencia en las búsquedas y borrados de datos, ya que un gran número de estas operaciones se llevan a cabo en la variable elegida como árbol AVL. Si se quisiera recorrer un gran número de veces una estructura de datos , los árboles AVL no son la estructura adecuada en cuanto a eficiencia. Para definir un árbol AVL, se definirán a continuación árbol binario y árbol binario de búsqueda.

Un árbol binario (AB) puede definirse como una estructura de datos en la cual cada nodo como máximo puede tener dos hijos (de ahí el nombre «binario»), un hijo izquierdo y un hijo derecho. Los árboles binarios se utilizan frecuentemente para representar conjuntos de datos cuyos elementos se identifican por una clave única. En la Figura 4.1 se puede ver un árbol binario.

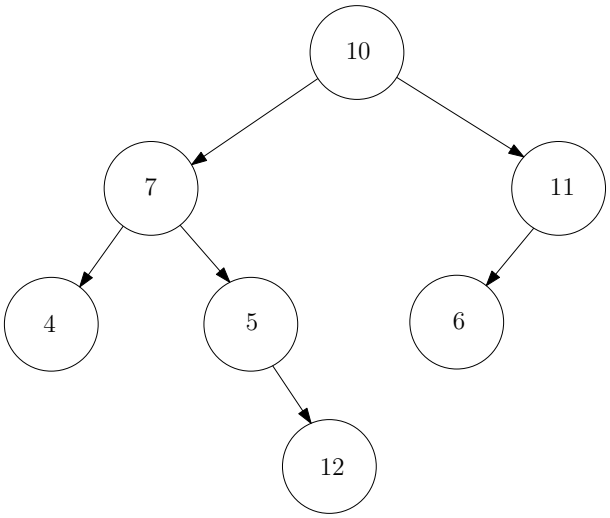


Figura 4.1: *Árbol binario.*

Un uso común de los árboles binarios son los árboles binarios de búsqueda. Un árbol binario de búsqueda (ABB) es un árbol binario que está organizado de tal manera que la clave de cada nodo es mayor que todas las claves su subárbol izquierdo, y menor que todas las claves del subárbol derecho. Para esta definición se considera que hay una relación de orden establecida entre los elementos de los nodos. En la Figura 4.2 se observa este tipo de árbol binario.

El árbol AVL toma su nombre de las iniciales de los apellidos de sus inventores, Adelson-Velskii y Landis. Lo dieron a conocer en la publicación de un artículo en 1962: *An algorithm for the organization of information* (Un algoritmo para la organización de la información). Se trata de un árbol binario de búsqueda en el que para cada nodo, las alturas de sus subárboles izquierdo y derecho no difieren en más de una unidad. En la Figura 4.3 se puede observar un árbol binario de búsqueda no equilibrado y uno equilibrado (AVL).

No se trata de árboles perfectamente equilibrados, pero sí son lo suficientemente equilibrados como para que su comportamiento sea lo bastante bueno como para usarlos donde los ABB no garantizan tiempos de búsqueda óptimos. Gracias a esta forma de equilibrio (o balanceo),

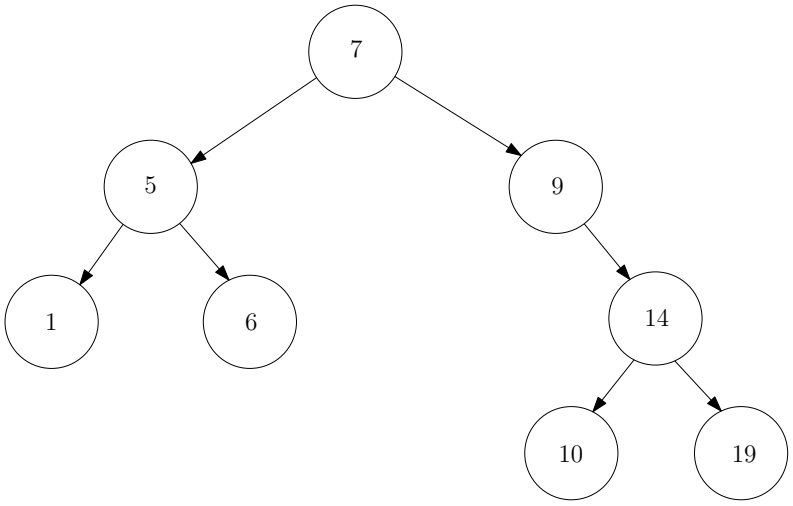
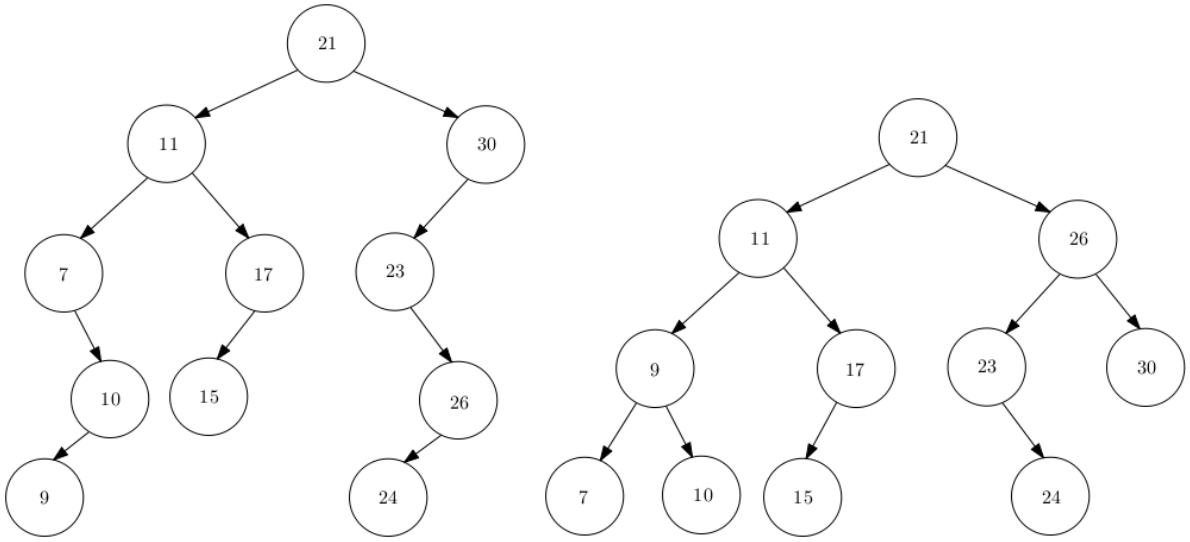


Figura 4.2: *Árbol binario de búsqueda.*

la complejidad de una búsqueda en uno de estos árboles se mantiene siempre en orden de complejidad $O(logn)$. El factor de equilibrio (diferencia de altura entre la rama derecha y la izquierda) puede ser almacenado directamente en cada nodo o ser computado a partir de las alturas de los subárboles. Para conseguir esta propiedad de equilibrio, la inserción y el borrado de los nodos se ha de realizar de una forma especial. Si al realizar una operación de inserción o borrado se rompe la condición de equilibrio, hay que realizar una serie de rotaciones de los nodos.



(a) Árbol binario no equilibrado (b) Árbol binario equilibrado (AVL)

Figura 4.3: *Árboles binario equilibrado (AVL) y no equilibrado*

4.2.2. Concepto de herencia y polimorfismo en *C++*

Tanto en la maqueta de simulación como en el protocolo de encaminamiento implementado se ha utilizado el concepto de herencia y polimorfismo. Después de definir ambos conceptos se explicará el por qué de su uso.

La herencia en *C++* es un mecanismo de abstracción creado para poder facilitar y mejorar el diseño de las clases de un programa. Con ella se pueden crear nuevas clases a partir de clases ya hechas, siempre y cuando tengan un tipo de relación especial.

Al crear nuevas clases a partir de clases existentes, se conservan las propiedades de la clase original, pudiéndose redefinir estos comportamientos (polimorfismo) y añadir comportamientos nuevos propios de las clases nuevas. La nueva clase obtenida se conoce como clase derivada, y a veces clase hija, subclase o clase descendiente. Las clases a partir de la/s cual/es se deriva recibe/n el/los nombre/s de clase/s base/s, en ocasiones también se la/s conoce/n como clase/s padre/s, superclase/s o clase/s ascendiente/s. Además, cada clase derivada puede usarse como clase base para obtener una nueva clase derivada [23].

La gran ventaja de los mecanismos de herencia es la reutilización del código, que permite que un programador pueda utilizar una clase como clase base de otras nuevas clases, con la característica añadida de que no hace falta comprender el código fuente de la clase base, sólo hace falta saber lo que hace.

Pero la herencia tiene otra característica interesante, la extensibilidad. Esta propiedad permite que los programas sean fácilmente ampliables, así de una clase base se pueden derivar varias clases que tengan un interfaz común, pero su realización y las acciones que llevan a cabo sean diferentes, así el programa principal controlará un grupo de estos objetos, pudiendo utilizar una función miembro a cualquier objeto, pero el efecto será diferente, dependiendo de las subclases específicas [69].

Antes de utilizar la herencia, hay que hacerse una pregunta, y si tiene sentido, se puede intentar usar esta jerarquía: Si la frase <claseB>es un <claseA>tiene sentido, entonces se está ante un posible caso de herencia donde clase A será la clase base y clase B la derivada.

El concepto de herencia ha sido utilizado para facilitar la implementación de protocolos de encaminamiento nuevos. Cuando se quiere implementar un protocolo de enrutamiento nuevo, se crea una clase derivada que hereda de la clase *Nodo* (clase que se explica en la Sección 4.3). Esto permite utilizar las características de un nodo genérico sin ningún protocolo de enrutamiento

montado sobre él, ya implementadas en la clase *Nodo* (por lo tanto no hay que volverlas a implementar). También ofrece la posibilidad de añadir las funciones propias del protocolo de encaminamiento a implementar. La herencia también se utiliza para implementar los eventos que utilizan los nodos en sus acciones (explicado en la Sección 4.3.7), ya que existen varios tipos de eventos con características comunes pero cada uno tiene algunas características propias.

Otro concepto muy importante que está relacionado con herencia es el polimorfismo. Se denomina polimorfismo a la capacidad que tienen los objetos de una clase de responder al mismo mensaje o evento en función de los parámetros utilizados durante su invocación. Un objeto polimórfico es una entidad que puede contener valores de diferentes tipos durante la ejecución del programa. Dicho de otra forma, el polimorfismo consiste en conseguir que un objeto de una clase se comporte como un objeto de cualquiera de sus subclases, dependiendo de la forma de llamar a los métodos de dicha clase o subclases. Una forma de conseguir objetos polimórficos es mediante el uso de punteros a la superclase. De esta forma se puede tener dentro de una misma estructura (arrays, listas, pilas, colas, ...) objetos de distintas subclases, haciendo que el tipo base de dichas estructuras sea un puntero a la superclase.

El polimorfismo es usado para que al implementar cualquier protocolo de encaminamiento nuevo, los objetos creados con dicha clase sean almacenados en una estructura de datos independiente del tipo de protocolo a implementar. También permite que una misma función común a todos los protocolos de encaminamiento (por ejemplo procesar mensaje) se comporte de manera distinta dependiendo del tipo de protocolo implementado. Además, el polimorfismo permite almacenar los distintos tipos de eventos generados por los nodos en la misma estructura de datos.

4.2.3. Uso de plantillas (*templates*) en *C++*

Debido al aumento de la complejidad de los programas y sobre todo, a los problemas a los que hay que enfrentarse, se tienen que repetir una y otra vez las mismas estructuras de datos. Por ejemplo, a menudo se tienen que implementar arrays dinámicos para diferentes tipos de objetos, o listas dinámicas, pilas, colas, árboles, etc. El código es similar siempre, pero hay que volver a rescribir ciertas funciones que dependen del tipo o de la clase del objeto que se almacena.

Las plantillas (*templates*) permiten parametrizar estas clases para adaptarlas a cualquier tipo de dato. Son el mecanismo de *C++* para implantar el paradigma de la programación genérica.

Permiten que una clase o función trabaje con tipos de datos abstractos, especificándose más adelante cuáles son los que se quieren usar. Por ejemplo, es posible construir un vector genérico que pueda contener cualquier tipo de estructura de datos. De esta forma se pueden declarar objetos de la clase de este vector que contengan enteros, flotantes, polígonos, figuras, fichas de personal, etc.

Las plantillas son utilizadas en las estructuras de datos *Arbol_AVL*, *Lista* y *Lista_dobenlace* para facilitar el uso de dichas estructuras con distintos tipos de datos. Así, solamente se tendrá una clase por cada estructura y no una clase por cada tipo de datos que se quiere almacenar.

4.3. Descripción de las clases implementadas en la maqueta de simulación

En esta sección se hará una descripción general de las clases que componen la maqueta de simulación, explicando las funcionalidades principales de cada clase.

4.3.1. Alea

La clase *Alea* está implementada en dos ficheros, *alea.cpp* y *alea.h*. En esta clase se declara la función *hnormal* que permite obtener la función de distribución de la normal estándar ($\Phi(x)$). A partir de esta función de distribución se calcula la función $\mathcal{Q}(x) = 1 - \Phi(x)$. La probabilidad de que haya un error en la transmisión de un mensaje, teniendo en cuenta que la longitud del mensaje son $31 * 8$ bits, se calcula como:

$$Pe = 1 - \left(1 - \mathcal{Q}\left(\sqrt{2\frac{Eb}{N0}}\right)\right)^{31*8} \tag{4.1}$$

Esta clase también ofrece otras funciones que no son utilizadas en este proyecto.

4.3.2. Arbol_AVL

El concepto de árbol AVL ya ha sido explicado en la Sección 4.2.1. En esta clase se implementan las características principales de la plantilla de un árbol de búsqueda binario auto-balanceable y está escrita en el fichero *arbol_AVL.hpp*. El árbol AVL se ha implementado para poder ser usado tanto con estructuras estáticas como con dinámicas. Un ejemplo de declaración de un

árbol AVL utilizando esta clase es la declaración del árbol de nodos sensores implementado en la maqueta de simulación.

4.3.3. Lista

La forma más simple de estructura dinámica es la lista simplemente enlazada. En esta forma los nodos se organizan de modo que cada uno apunta al siguiente, y el último no apunta a nada, es decir, el puntero del nodo siguiente vale *NULL*. Esta estructura es eficiente cuando se quieren ir procesando uno a uno sus elementos y el número de búsquedas y borrados es pequeño. En la Figura 4.4 se aprecia una lista simplemente enlazada. La clase *Lista* está implementada en el fichero *lista.hpp* como una plantilla de *C++* para aprovechar los beneficios de ésta.

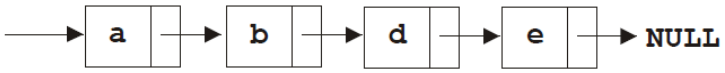


Figura 4.4: *Lista simplemente enlazada.*

4.3.4. Lista_dobenlace

Una lista doblemente enlazada es una lista lineal en la que cada nodo tiene dos enlaces, uno al nodo siguiente, y otro al anterior. Las listas doblemente enlazadas pueden recorrerse en ambos sentidos a partir de cualquier nodo. Esto es posible debido a que a partir de un nodo siempre se puede alcanzar cualquier otro nodo de la lista hasta que se llega a uno de los extremos. En la Figura 4.5 se observa la estructura de una lista doblemente enlazada. La clase *Lista_dobenlace* está escrita en el fichero *lista_dobenlace.hpp* como una plantilla de *C++* para que se pueda declarar con diferentes tipos de datos.

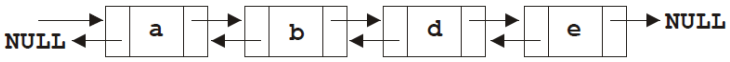


Figura 4.5: *Lista doblemente enlazada.*

4.3.5. Cadena

Al emplear algunos tipos de datos en las plantillas de *C++* existen algunas limitaciones. No se puede crear una lista de cadenas usando `Lista<char *>`, ya que de ese modo sólo se crea una lista de punteros. Para poder crear listas de cadenas se ha implementado una clase

especial para cadenas, en la que además de encapsular las cadenas se ha definido los operadores $=, ==, != , >, <, >=, <=$, algunos de los cuales son necesarios para poder crear listas con esta clase. Esta clase está implementada en *cadena.hpp*.

4.3.6. Mensaje

La clase *Mensaje* almacena las características de un mensaje genérico. Esta clase servirá como clase padre para implementar posteriormente los mensajes utilizados en cada protocolo de encaminamiento. Esta clase está escrita en *mensaje.cpp* y *mensaje.hpp*. El mensaje definido por esta clase se muestra en la Figura 4.6 y está compuesto por los siguientes campos:

- El tipo del paquete nos indica si es de datos, ACK o cualquier tipo del protocolo de encaminamiento usado.
- El identificador del nodo destino del siguiente salto.
- El identificador del nodo origen del siguiente salto.
- El identificador del nodo destino final al cual va dirigido el mensaje.
- El identificador del nodo origen inicial que generó el mensaje.
- El número de secuencia del origen inicial, que permite diferenciar los mensajes enviados por el nodo origen inicial del mensaje.

Además, este mensaje puede ser de dos tipos:

- DATOS: cuando se envía un mensaje de datos.
- ACK: cuando se envía un reconocimiento positivo.

También se define BROADCAST para poder enviar el mensaje teniendo como destino del siguiente salto a todos los vecinos del nodo que quiere enviar dicho mensaje.

Para decidir si dos mensajes son iguales o diferentes, se comprueba que coinciden los campos tipo, identificador del nodo destino final, identificador del nodo origen inicial y número de secuencia del origen inicial. Los campos identificador del nodo destino y origen del siguiente salto se utilizan para enviar el mensaje entre nodos vecinos.

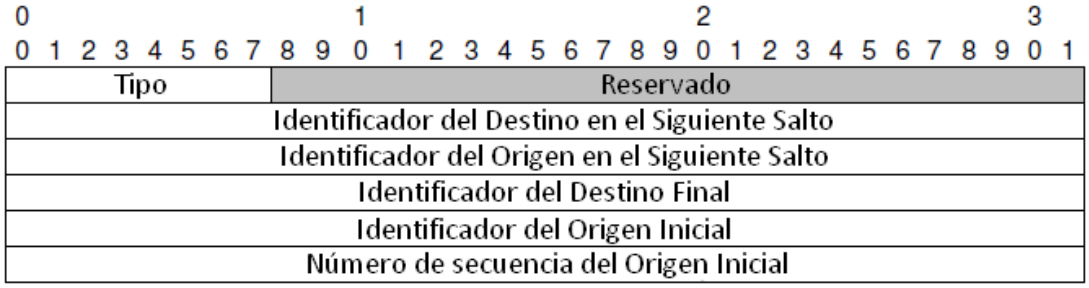


Figura 4.6: *Mensaje genérico.*

4.3.7. Evento

Esta clase es utilizada para implementar los eventos generados por cada uno de los nodos cuando realizan una acción. Además, se utiliza como clase base para las clases derivadas *Evento_TX* y *Evento_RX*, permitiendo el uso del polimorfismo para almacenar los distintos tipos de eventos en una lista polimórfica. Esta clase está implementada en *evento.cpp* y *evento.hpp*. Cada evento tiene un identificador propio (siempre será mayor que cero) que lo diferencia del resto , el identificador del nodo que lo ha generado, el tipo de evento que es, su inicio y duración en segundos. Además contiene una lista con los identificadores de los nodos vecinos que están pendientes de procesar el evento, es decir, comprobar si tiene alguna influencia en ellos. Todos estos campos se pueden observar en la Figura 4.7. Para conocer cómo se procesan los eventos hay que leer la clase *Gestor* (ir a la Sección 4.3.13). Los tipos de eventos son los siguientes:

- INICIO: tipo usado cuando se inicia la simulación. Cada nodo comienza en estado dormido pero con un desfase aleatorio desde el inicio de dicha simulación.
- TRANSMISION: este tipo es utilizado cuando un nodo quiere trasmitir un mensaje que no se ha generado tras la detección en su entorno de la magnitud sensada.
- RETRANSMISION: cuando se retransmite un mensaje se genera un evento de este tipo.
- TRANSMISION_ACK: al transmitir un reconocimiento positivo el tipo de evento es éste.
- TRANSMISION_EVENTO: cuando un nodo detecta en su entorno la magnitud que está sensando, generará un mensaje para transmitirlo a su nodo sumidero. El evento que permite la transmisión del mensaje es del tipo citado.
- RECEPCION: utilizado cuando un nodo está escuchando el medio.

- DORMIDO: el nodo estará dormido durante la duración de este tipo de evento.

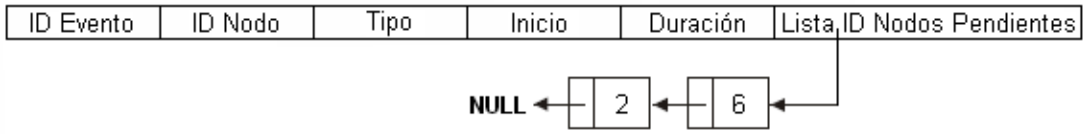


Figura 4.7: *Evento.*

4.3.8. Ficheros

La clase *Ficheros*, que está implementada en *ficheros.cpp* y *ficheros.hpp*, se utiliza para gestionar todos los ficheros que intervienen en la simulación, ya sean de entrada (ficheros de datos y configuración) como de salida (fichero de simulación). Para entender la información que contiene cada fichero (*WSN.conf*, *WSN.dat* y *WSN.sim*) habrá que leer la Sección 4.4. Esta clase almacena los siguientes valores:

- Duración en segundos de un paquete de datos.
- Duración en segundos de un paquete ACK.
- Duración en segundos del período de cada nodo. Se entiende por período de cada nodo a la suma de la duración de los tiempos en los cuales el nodo está dormido, está escuchando el medio y está transmitiendo.
- Porcentaje de período del nodo dedicado para escuchar.
- Duración en segundos del tiempo en el cual el nodo está escuchando el medio.
- Porcentaje de período del nodo dedicado a transmitir.
- Duración en segundos del tiempo en el cual el nodo está transmitiendo.
- Duración en segundos del tiempo en el cual el nodo está dormido.
- Probabilidad de ocurrencia de sucesos sensados. Se supone que si se da un suceso se detecta correctamente. Tras la ocurrencia de un evento sensado, el nodo que lo ha detectado envía un mensaje de datos a su nodo sumidero.

- Número de nodos que forman la red.
- Número máximo de vecinos de un nodo.
- Máxima anchura en metros del emplazamiento a sensar.
- Máxima longitud en metros del emplazamiento a sensar.
- Tiempo de fin de la simulación.
- Nombre del protocolo de encaminamiento a simular.
- Número de nodos sumideros en la red.
- Consumo de batería cuando el nodo está dormido.
- Consumo de batería cuando el nodo está activo.
- Número de iteraciones que se repite la simulación para evitar la aleatoriedad de los resultados.

Además de los valores anteriores, esta clase contiene una lista que guarda la información necesaria para realizar las simulaciones pedidas por el usuario. Cada nodo de la lista contiene:

- El nombre del parámetro a simular.
- El valor inicial con el cual el parámetro comenzará la simulación.
- El incremento del parámetro con cada simulación.
- El valor final del parámetro con el cual finalizará la simulación.
- Una lista con los parámetros de salida de la simulación (*throughput*, retardo medio en la red y batería media restante) que el usuario considera que son de su interés, y que se guardarán en el fichero de salida de la simulación (*WSN.sim*) para su posterior procesado.

Esta clase permite cargar los valores para la configuración de la maqueta de simulación. Estos valores se encuentran almacenados en el fichero *WSN.conf*. En el caso de que el fichero anterior no exista, la clase *Ficheros* generará uno con todos los parámetros por defecto que pueden utilizarse en la fase de configuración. También se proporciona una función para comprobar que el fichero de configuración a cargar no contiene ningún error.

Con respecto al fichero *WSN.dat*, esta clase lo trata de manera simular al fichero de configuración. Al intentar cargar el fichero de datos comprueba si existe o no, en caso afirmativo carga el fichero de datos comprobando que no tiene ningún error. Si el fichero de datos no existe, creará uno nuevo basándose en los parámetros del fichero de configuración.

El fichero de simulación *WSN.sim* es de salida y en él no se hace comprobación de ningún tipo. Solamente recalcar que cuando se inicia una nueva simulación el fichero de simulación anterior se borrará.

4.3.9. Simulación

En cada simulación es necesario almacenar un conjunto de parámetros de interés. Este almacenamiento es el que realiza la clase *Simulación*, además de calcular el retardo medio de la red. Esta clase está escrita en los ficheros *simulacion.cpp* y *simulacion.hpp*. Los parámetros definidos como una media aritmética de muestras se calculan de la siguiente manera. Para obtener un valor más fiable, se hacen varias iteraciones de la simulación (definidas en el fichero de configuración) y se va almacenando la suma del parámetro medio (suma de cada muestra del parámetro dividido por el número de muestras que se han tomado de dicho parámetro) tras cada iteración. Al terminar se obtiene el parámetro medio promediando con el número de iteraciones realizadas.

Los parámetros que se almacenan son:

- La batería media restante en la red.
- La batería total restante en la red en la última iteración.
- El retardo medio de la red.
- El retardo total de la red en la última iteración.
- La tasa efectiva de transporte de información.
- El tiempo de muerte del primer nodo.
- El número de mensajes transmitidos en la última iteración.
- El número de mensajes recibidos en la última iteración.

Para el cálculo del retardo se utiliza una lista en la que cada nodo de la lista contiene el mensaje transmitido, el instante de tiempo de generación del mensaje y el instante de tiempo

de recepción del mismo. El retardo se calcula fácilmente como la diferencia entre el instante de tiempo de recepción del mensaje y el instante de tiempo de generación del mismo. Cuando se habla de tiempo de generación del mensaje se refiere al tiempo en el cual el mensaje fue creado por el nodo origen de la transmisión, aunque este nodo no lo pueda enviar inmediatamente porque no conoce el destino final del mensaje, y tenga que aplicar un algoritmo de encaminamiento para conocer la ruta.

4.3.10. Evento_RX

Clase derivada de la clase *Evento*, utilizada para los eventos de tipo RECEPCION. Está implementada en los ficheros *evento_RX.cpp* y *evento_RX.hpp*. Esta clase, además del contenido de la clase *Evento*, posee un lista donde se van almacenando todos los eventos transmisores generados por los nodos vecinos. Estos eventos transmisores son procesados una vez que el evento de recepción finaliza. En esta clase, también se realiza la comprobación de las posibles colisiones entre los mensajes que han sido transmitidos al nodo que generó el evento receptor durante el transcurso de dicho evento, perdiéndose los mensajes que han colisionado. La comprobación de colisiones se lleva a cabo cuando el evento receptor finaliza. Un evento de este tipo se puede ver en la Figura 4.8

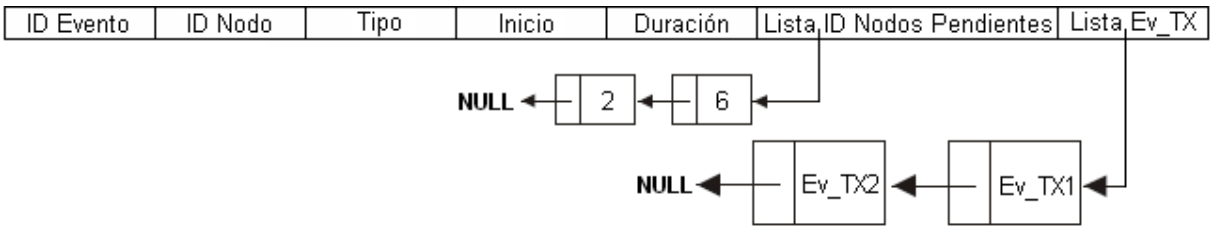


Figura 4.8: *Evento Receptor.*

4.3.11. Evento_TX

La clase Evento_TX es una clase hija de la clase *Evento*, usada para los eventos de tipo *TRANSMISION*, *TRANSMISION_EVENTO*, *TRANSMISION_ACK* y *RETRANSMISION*. Está implementada en los ficheros *evento_TX.cpp* y *evento_TX.hpp*. Esta clase tiene los comportamientos de su clase padre y además almacena el mensaje que ha dado lugar a la generación del evento. El mensaje guardado podrá ser de cualquier tipo debido al uso del polimorfismo, esto

se debe a que la clase Mensaje es la clase base de las demás clases de mensajes implementadas para cada protocolo de enrutamiento. Un evento transmisor puede observarse en la Figura 4.9.

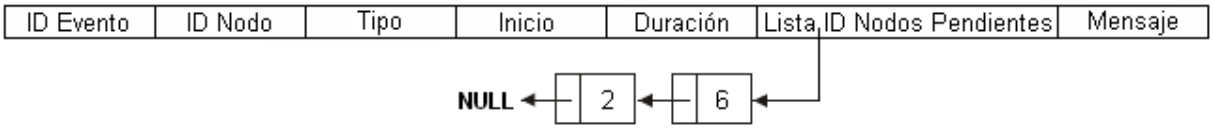


Figura 4.9: *Evento Transmisor.*

4.3.12. Nodo

La clase *Nodo* implementa, en los ficheros *nodo.cpp* y *nodo.hpp*, un nodo genérico de la red de sensores sin ningún protocolo de encaminamiento funcionando en él. Dicho nodo tiene una serie de características que se van citando a continuación. La posición en el emplazamiento a sensar, dado en coordenadas (x,y), es necesaria para conocer los vecinos del nodo. El identificador del nodo en la red (todos los identificadores serán mayores que cero) permite que el nodo sea reconocido en toda la red sin ninguna confusión. La batería restante del nodo le permite seguir con vida en la red. El identificador del nodo sumidero al cual se ha de enviar la información sensada, en caso de haber más de un nodo sumidero, se elegirá el nodo sumidero con menor distancia al nodo en cuestión. El número de secuencia para el siguiente mensaje que genere el nodo, para diferenciar los mensajes enviados por dicho nodo. El número máximo de vecinos que el nodo puede tener, dichos vecinos estarán ordenados por cercanía al nodo en cuestión. En caso de tener un mayor número de vecinos que el máximo permitido, el nodo se quedará con los vecinos más cercanos en cuanto a distancia se refiere (menores pérdidas de propagación). Los vecinos son almacenados en una lista ordenada de menores a mayores pérdidas de propagación (menor o mayor distancia al nodo). En dicha lista se guardan las pérdidas de propagación, la probabilidad de error para transmitir al nodo vecino y el identificador del nodo vecino en cuestión. El nodo tiene tres listas de mensajes, la que almacena los mensajes dispuestos a ser transmitidos en el siguiente evento de transmisión, una segunda que guarda los mensajes recibidos y una con los mensajes de datos. En la Figura 4.10 puede observarse la estructura de un nodo.

En la lista de mensajes a transmitir, además del mensaje, se almacena información para poder utilizar retransmisiones de los mensajes. El número de retransmisiones de dicho mensaje está pensado para que al llegar a un número máximo, no se reenvíe más y se elimine. El inicio

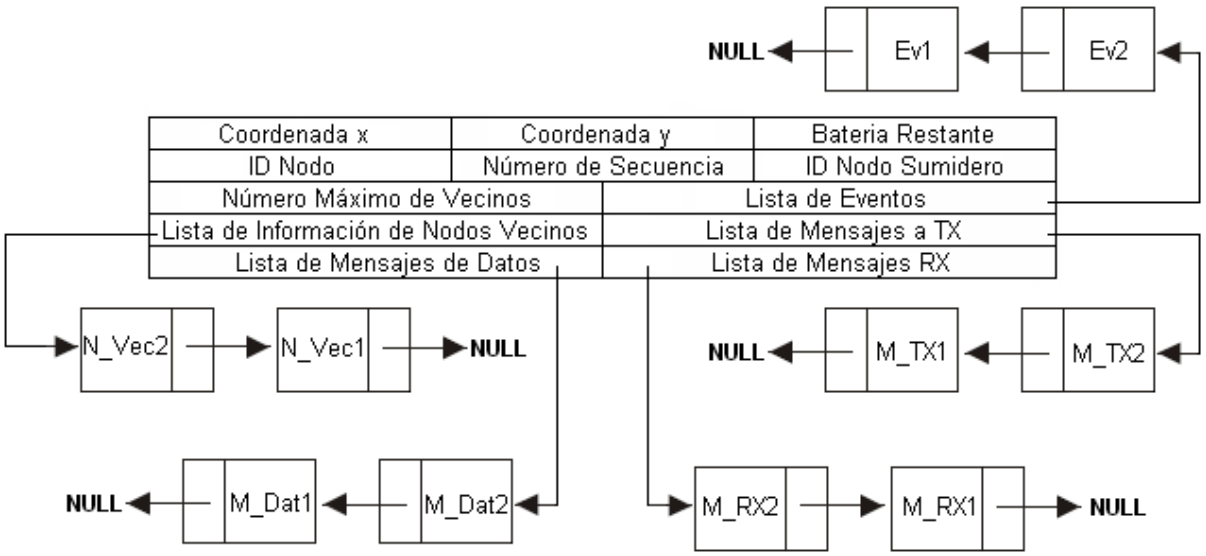


Figura 4.10: *Nodo.*

de la siguiente transmisión permite separar las retransmisiones durante un intervalo de tiempo, en dicho intervalo se podría recibir el reconocimiento positivo por parte del destino y no volver a retransmitir el mensaje. Por último existe una lista con el/los identificador/es de el/los nodo/s de el/los cual/es se debe recibir el/los reconocimiento/s positivo/s al mensaje. En el entorno de simulación diseñado se realizaron varias pruebas utilizando reconocimientos positivos para los mensajes, pero dado que no se recibían apenas reconocimientos por usar un acceso al medio por contienda (el nodo transmite cuando detecta la magnitud sensada sin importarle si el medio está ocupado), únicamente se asienten positivamente los mensajes de datos. El reconocimiento positivo va desde el destino del mensaje hasta el origen, haciendo caso omiso al mismo los nodos intermedios. Por el mismo motivo, no se utilizan retransmisiones de los mensajes.

La lista de mensajes recibidos contiene otro campo además del mensaje, dicho campo es el instante de tiempo en el que finalizó la recepción del mensaje. Este campo es de gran utilidad ya que permite usar un intervalo de tiempo, a partir del instante citado, en el cual el nodo si recibe el mensaje no lo procesará. Una vez pasado ese tiempo el mensaje se borrará, pudiéndose dar el caso de recibirlo y procesarlo otra vez.

La lista de mensajes de datos permite almacenar el mensaje hasta conocer la ruta hacia el destino final. Una vez conocida la ruta, se busca el mensaje en la lista, se borra la entrada de la lista y se envía el mensaje.

En el nodo existe otra lista muy importante, es la lista de eventos. En esta lista se almacenan todos los eventos que el nodo va a realizar en un futuro cercano. Una vez que un evento ya ha sido procesado, se borra de la lista. Toda la información de generación y procesado de eventos se encuentra en la clase *Gestor* (Sección 4.3.13).

Tras la ubicación de todos los nodos, cada uno de ellos realiza una búsqueda de nodos vecinos. Se considera nodo vecino al nodo que se encuentra en un radio en el que las pérdidas de propagación son menores o iguales a las pérdidas de propagación L calculadas para una SNR de $10dB$ según las siguientes ecuaciones:

$$Ni = KBT + F + Fa \tag{4.2}$$

$$Pr = SNR + Ni \tag{4.3}$$

$$L = Pt - Pr \tag{4.4}$$

siendo K la constante de Boltzmann ($K = 1,38 * 10^{23} J/K$), B el ancho de banda en Hz ($B = 200KHz$) y T la temperatura absoluta ($T = 290K$). Con los valores citados, KBT toma un valor de $-121dBm$. F es la figura de ruido del receptor ($10dB$), Fa la figura de ruido de la antena ($10dB$) y Pt la potencia transmitida en dBm ($15dBm$). El valor de L, operando con los parámetros anteriores, es de $-91dB$.

Para calcular las pérdidas entre dos nodos y ver si son vecinos, se utiliza el modelo de propagación *Log distance Path Loss Model* de dos pendientes [27]. Este modelo se ha elegido porque es apropiado para entornos reducidos en área urbana y entornos de interior. La finalidad es poder montar una red de sensores de interior con algún protocolo funcionando de los simulados mediante la maqueta de simulación implementada, y que los resultados reales tengan relación con los resultados obtenidos con la simulación. El modelo de propagación implementado devuelve las pérdidas de propagación en decibelios. Las ecuaciones del modelo son las siguientes:

$$L(d) = \begin{cases} L_{0_1} + 10 n_1 log_{10}(d) & si \quad d \leq d_r \\ \\ L_{0_2} + 10 n_2 log_{10}(d) & si \quad d > d_r \end{cases} \tag{4.5}$$

donde en L_{0_1} son las perdidas a 1 metro, n_1 y n_2 son los factores de decaimiento, d es la distancia entre los dos nodos expresada en metros y d_r el punto de ruptura en metros, situado

entre los dos nodos, en el que se cambian las pendientes. Los valores de estas variables han sido cogidos de un estudio realizado en [27] tomando como escenario el patio de la Politécnica de Cartagena. Los valores son $L_{0_1} = 25dB$, $n_1 = 2{,}76$, $n_2 = 4{,}01$ y $d_r = 3{,}2$.

La representación de la potencia recibida por un nodo con respecto a la distancia entre dicho nodo y el nodo transmisor se puede observar en la Figura 4.11. Como se puede apreciar en dicha imagen, la distancia de $26,87m$ indica que para que un nodo sea vecino de otro debe estar a un radio menor o igual a dicha distancia.

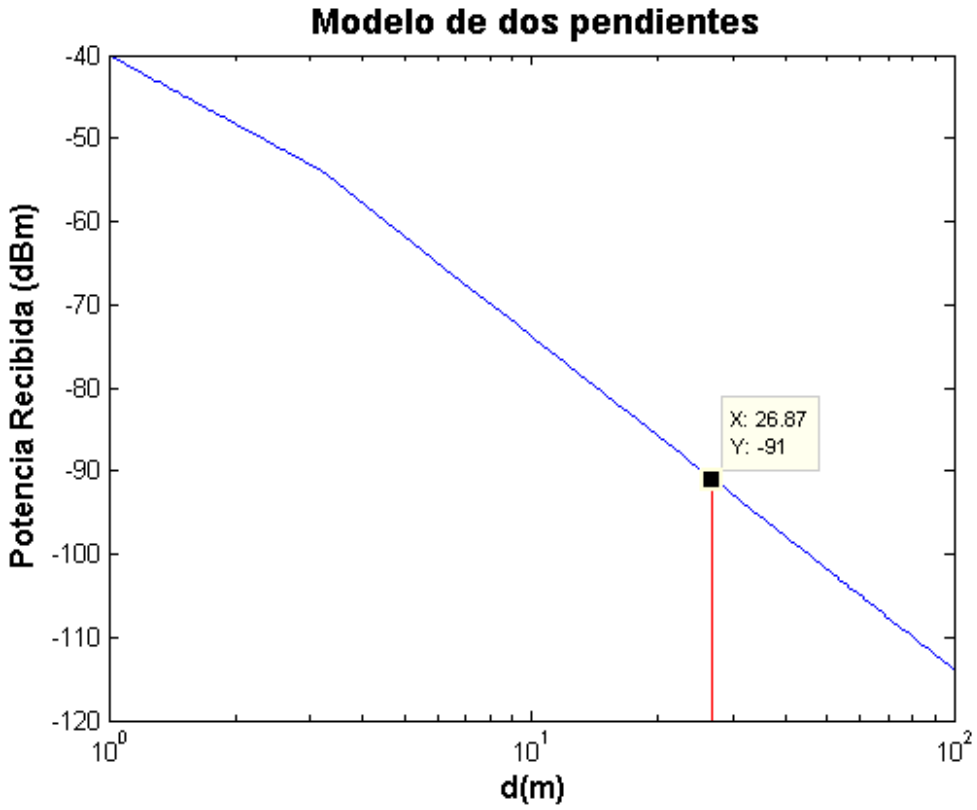


Figura 4.11: *Log distance Path Loss Model.*

En esta clase se definen dos funciones virtuales muy importantes para la posterior implementación del protocolo de encaminamiento. Definir una función como virtual permite que pueda ser redefinida en las clases derivadas de la clase en la cual se definió la función. Las dos funciones virtuales son `Mensaje_al_Detectar_Sensor` y `Procesar_Mensaje`. La función `Mensaje_al_Detectar_Sensor` permite a cada protocolo de encaminamiento definir la búsqueda de la ruta necesaria cuando se quiere enviar un mensaje de datos, tras detectar la magnitud sensada, y no se conoce la ruta hacia el destino. La función `Procesar_Mensaje` define las de-

cisiones de encaminamiento tras procesar cada mensaje, así como, qué hacer cuando llega un mensaje de datos o de reconocimiento positivo.

Para ver cómo funciona el entorno de simulación sin protocolos de encaminamiento, cuando un nodo detecta la magnitud sensada se genera un destino aleatorio entre sus vecinos y se envía el mensaje de datos. Cuando se recibe un mensaje, se comprueba si el destino final es el nodo en cuestión, en caso afirmativo se crea un mensaje ACK y se envía al nodo del cual se ha recibido el mensaje, en caso negativo el mensaje se descarta. Por tanto, se puede obtener cómo varía el *throughput* y la batería media restante en la red con respecto a la variación del porcentaje de escucha, el periodo y la probabilidad de ocurrencia de sucesos a sensar. Estas pruebas están explicadas en la Sección 4.5. En este caso, al no estar simulando protocolos de enrutamiento, el periodo de transmisión, definido con su porcentaje en el fichero de configuración, no se utilizará ya que cada nodo, tras detectar la magnitud sensada, envía a un destino aleatorio entre sus vecinos. Dicho mensaje llegará al destino o se perderá pero no generará ningún proceso posterior.

4.3.13. Gestor

Es la clase más importante de todas ya que se encarga de «encajar las piezas del puzzle» para que todo funcione correctamente. Está escrita los ficheros *gestor.cpp* y *gestor.hpp*. Contiene el identificador del evento siguiente a generar, éste se utiliza para ir asignando un identificador diferente a cada evento generado. El número de nodos que han finalizado la simulación es otra variable usada para saber cuándo han finalizado la simulación la totalidad de nodos y se puede dar fin a la ejecución. Esta clase permite generar números aleatorios entre un valor inicial y un valor final con un cierto número de decimales. Esto es usado principalmente al inicio de la simulación, cuando cada nodo empieza dormido con un desfase en el tiempo aleatorio. Permite, además, ir tomando muestras en el tiempo de la batería media restante en la red, declarándose el período de muestreo en el fichero de configuración. Tiene un gestor de ficheros para poder realizar las operaciones necesarias con cada uno de los ficheros utilizados por el ejecutable para obtener o almacenar datos. El gestor de la simulación también tiene cabida en esta clase, y es utilizado para almacenar los parámetros de interés en cada simulación.

La clase *Gestor* tiene una lista en la que se almacenan los identificadores de los nodos sumideros en la red. También tiene un árbol AVL polimórfico (en el cual se pueden almacenar cualquier clase derivada de la clase *Nodo*) en el que se almacenan los nodos que forman la red.

Como se comentó cuando se hizo la explicación de qué es un árbol AVL, esta estructura se utiliza porque en cada simulación es necesario hacer múltiples búsquedas de nodos como posteriormente se explicará.

En esta clase se gestiona la carga del fichero de configuración, ya sea a partir de uno existente o a partir de uno creado nuevo. También permite la carga del fichero de datos que contiene las posiciones de cada nodo en la red. Los identificadores de los nodos que harán el papel de nodo sumidero en la red son generados de forma aleatoria en esta clase. Los nodos sumideros no envían ningún tipo de mensaje puesto que son los nodos en los cuales se procesa la información de detección. Una vez que los identificadores de los nodos sumideros han sido generados y almacenados en la lista correspondiente, se asigna el nodo sumidero más cercano a cada nodo de la red. La búsqueda de vecinos por parte de cada nodo se dirige por esta clase y hace uso de la función correspondiente implementada en la clase *Nodo*.

La simulación del protocolo de enrutamiento especificado en el archivo de configuración se lleva a cabo en esta clase. Se hace uso de la lista de parámetros a simular que tiene el gestor de ficheros y para cada parámetro se realiza una simulación, ofreciendo como salida los parámetros de interés especificados en el fichero de configuración y almacenados en la lista citada anteriormente. La simulación se repite una serie de iteraciones (definidas en el fichero de configuración) ya que la aleatoriedad influye en ella por utilizar ciertos parámetros aleatorios (probabilidad de ocurrencia de sucesos a sensar, probabilidad de error entre dos nodos, generación aleatoria del inicio de la transmisión de un paquete de datos tras detectar algo en el medio...). Los parámetros de salida de la simulación se promedian al final de cada iteración como se explicó en la Sección 4.3.9.

Dos de las acciones fundamentales en el desarrollo de la simulación de protocolos de encaminamiento por parte de la maqueta de simulación implementada son la generación y el procesado de los eventos que llevan a cabo cada uno de los nodos. Estas dos acciones son descritas a continuación.

La generación de eventos se realiza de la siguiente manera. Al iniciarse la simulación, el primer evento generado será de tipo *DORMIDO*, cuyo tiempo de inicio será establecido de manera aleatoria entre 0 y el tiempo de duración del período de nodo definido en el fichero de configuración. El período de nodo (o ciclo del nodo) se define como el intervalo de tiempo que el nodo va repitiendo de manera periódica en el tiempo. Dicho período está formado por el ciclo pasivo (dormido) más el ciclo activo (escucha más transmisiones). El período de nodo

se puede observar en las Figuras 4.12 y 4.13. La duración de un evento *DORMIDO* dependerá del porcentaje de dormido sobre el periodo de nodo, obteniéndose el porcentaje de dormido al restar al 100 % del periodo de nodo, el porcentaje de escucha y el porcentaje de transmisiones. Aplicando el porcentaje de dormido sobre la duración de periodo de nodo se obtiene la duracion de un evento *DORMIDO*. Para que un nodo genere el siguiente evento a realizar, su lista de eventos tiene que estar vacía, es decir, ya se han procesado todos los eventos anteriores.

Una vez generado el primer evento para cada nodo de la red, los siguientes eventos se generan dependiendo del tipo de evento anterior generado. Así, tras generar el primer evento *DORMIDO* para cada nodo, el siguiente evento generado será de tipo *RECEPCION*. Dicho evento se iniciará justamente detrás de la finalización del evento *DORMIDO*, y tendrá una duración que dependerá del porcentaje de escucha respecto al periodo de nodo que está establecido en fichero de configuración. En la Figura 4.12 se aprecia la generación de un evento *DORMIDO* con un cierto desfase desde el inicio de la simulación, y la generación de un evento *RECEPCION* a continuación.

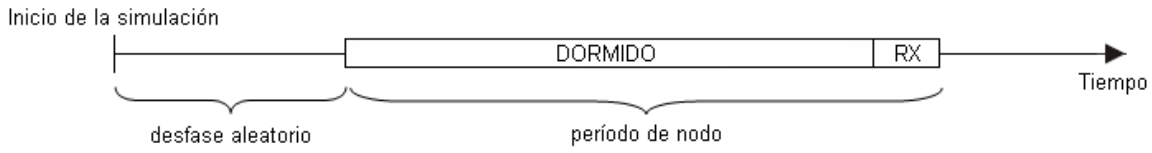


Figura 4.12: *Generación de eventos al inicio de la simulación.*

Tras generar un evento *RECEPCION*, el siguiente evento generado, si existen mensajes almacenados en la lista de eventos a transmitir del nodo que quiere generar el nuevo evento, será de tipo *TRANSMISION*, *RETRANSMISION* (si se utilizan retransmisiones) o *TRANSMISION_ACK*. En caso de que existan mensajes a transmitir, se generarán tantos eventos transmisores como mensajes haya almacenados en la lista de mensajes a transmitir y quepan en el intervalo de transmisiones. El intervalo de transmisiones se inicia tras la finalización del evento anterior, y su duración depende del porcentaje de transmisiones definido en el fichero de configuración. Se generen o no eventos de transmisión, el siguiente evento generado dependerá de la probabilidad de ocurrencia de sucesos a sensar definida en el fichero de configuración. Con esa probabilidad se generará un evento de tipo *TRANSMISION_EVENTO* para informar al nodo sumidero de que el nodo que ha generado dicho evento ha detectado algo. El inicio del evento de tipo *TRANSMISION_EVENTO* se establece de forma aleatoria en un intervalo que

comienza en la finalización del evento anterior y termina en la suma de dicha finalización más la duración de un evento *DORMIDO*. Dependiendo del inicio del evento de tipo *TRANSMISION_EVENTO*, se pueden crear uno o dos eventos en los que el nodo estará dormido. En caso de que el inicio del evento *TRANSMISION_EVENTO* no sea al comienzo o al final del intervalo citado anteriormente, el nodo generará un evento *DORMIDO* desde el final del evento anterior hasta el inicio de la transmisión. Ese tiempo se descontará a la duración del siguiente evento *DORMIDO* generado. Al final del evento *TRANSMISION_EVENTO* el nodo generará un evento *DORMIDO*, que se inicia en dicho final y termina en la suma de dicho final más el tiempo de dormido restante (diferencia entre duración de evento *DORMIDO* y duración del tiempo dormido anterior). Si el inicio del evento de tipo *TRANSMISION_EVENTO* se produce justamente tras la finalización del evento anterior (al inicio del intervalo citado anteriormente), solamente se generará un evento *DORMIDO* cuyo inicio será tras el evento de tipo *TRANSMISION_EVENTO* y cuya duración será el tiempo de dormido. En el caso de que el inicio del evento de tipo *TRANSMISION_EVENTO* se produce al final del intervalo donde se puede generar dicho evento, se producirá un evento *DORMIDO* cuyo inicio será al inicio del intervalo y cuya duración será el tiempo de dormido. En la Figura 4.13 se pueden apreciar los seis casos de generación de eventos. Una aclaración de dicha Figura es que *TX_EV* se refiere al evento generado tras detectar la magnitud sensada en el medio y *TX* se refiere a las transmisiones de cualquier otro tipo.

Una explicación más general de la generación de eventos es la siguiente. Tras un evento *DORMIDO* se genera un evento *RECEPCION*. Tras un evento *RECEPCION*, se pueden generar eventos de transmisión si existen mensajes almacenados en la lista de mensajes a transmitir, posteriormente, e independientemente de que se generen eventos de transmisión, se crea con una probabilidad definida en el fichero de configuración un evento de tipo *TRANSMISION_EVENTO* y antes (si hay hueco) y/o después (si hay hueco) de este evento se crea un evento *DORMIDO*. Cada vez que se genera uno o varios eventos, se insertan en la lista de eventos del nodo que los ha generado.

Una vez se ha generado el primer evento en cada nodo de la red, se procesan los eventos. Para ello, se recorre el árbol AVL de nodos y se obtiene uno de ellos. Dicho nodo tiene una lista de eventos, por tanto se empezará por el primer evento a procesar. El evento a procesar tendrá una lista con los identificadores de los nodos vecinos que están pendiente de procesar dicho evento. Se tomará el primer identificador de la lista de identificadores y se hará una búsqueda en el

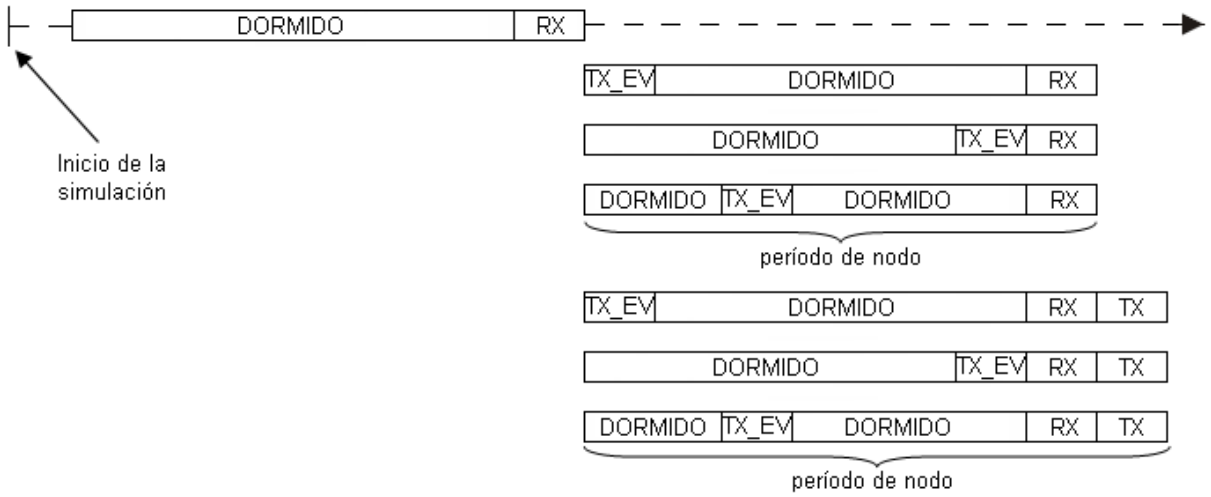
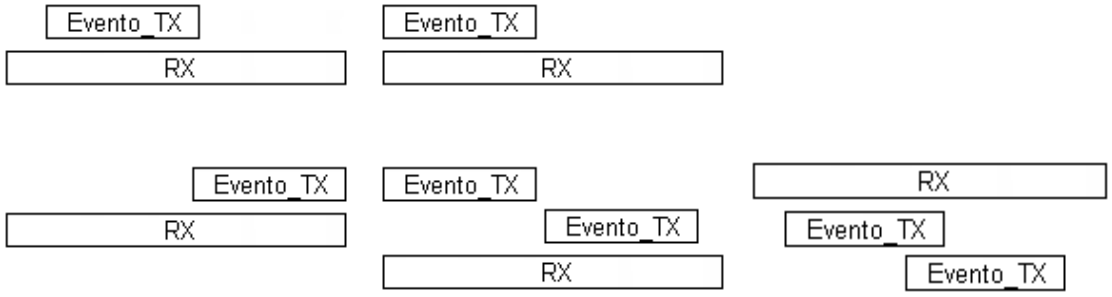


Figura 4.13: *Generación de eventos después del inicio de la simulación.*

árbol AVL de nodos para hallar el nodo vecino. En el nodo vecino se irá recorriendo su lista de eventos para ver si el evento a procesar tiene alguna influencia en alguno de sus eventos o alguno de sus eventos tienen alguna influencia en el evento a procesar. Las influencias entre dos eventos se dan cuando:

- Un evento a procesar de tipo TRANSMISION, TRANSMISION_EVENTO, TRANSMISION_ACK o RETRANSMISION se encuentra dentro del intervalo de duración de un evento receptor generado por un nodo vecino (el inicio del Evento_TX es mayor o igual que del evento vecino y el fin del Evento_TX es menor o igual que el fin del evento vecino). En este caso, el nodo receptor puede recibir el mensaje dependiendo de si no hay otro u otros mensajes recibidos por dicho nodo que colisionan con éste y de la probabilidad de error en la transmisión del mensaje que va a recibir. El cálculo de dicha probabilidad de error viene citado en la Sección 4.3.1. Además, se borra el identificador del nodo que generó el Evento_TX de la lista de identificadores de nodos vecinos del nodo vecino, puesto que el Evento_TX ya ha sido procesado por el vecino. Por la misma razón, también se borra el identificador del nodo vecino de la lista de identificadores de nodos vecinos del nodo que generó el Evento_TX. Este caso se puede observar en la Figura 4.14, en la cual el evento a procesar es el superior y el evento vecino el inferior en cada caso.

- Un evento a procesar del tipo TRANSMISION, TRANSMISION_EVENTO, TRANSMI-



(a) Recepción correcta (b) Colisión

Figura 4.14: *Ejemplos de recepción.*

SION_ACK o RETRANSMISION se encuentra dentro del intervalo de duración de un evento que no es de tipo RECEPTOR generado por un nodo vecino. En este caso, se borra el identificador del nodo que generó el Evento_TX de la lista de identificadores de nodos vecinos del nodo vecino, puesto que el mensaje almacenado en el Evento_TX no puede ser escuchado por el nodo vecino. Un ejemplo de esta situación se puede ver en la Figura 4.15, en la que el evento a procesar es el superior y el evento vecino el inferior en cada caso.

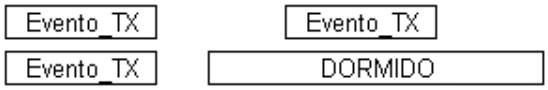


Figura 4.15: *Pérdida del mensaje porque el nodo vecino no está escuchando.*

- Un evento a procesar de tipo TRANSMISION, TRANSMISION_EVENTO, TRANSMISION_ACK o RETRANSMISION cuyo inicio es menor o igual que el fin del evento vecino y cuyo fin es mayor o igual que el inicio del evento vecino. Además, no se puede dar que el inicio del evento a procesar sea mayor o igual que el inicio del evento vecino y que el fin del evento a procesar sea menor o igual que el fin del evento vecino, puesto que esta opción ya está comprobada en una consulta realizada anteriormente y que está explicada en los dos párrafos anteriores. Independientemente del tipo del evento vecino, se elimina el identificador del nodo vecino de la lista de identificadores de nodos vecinos del evento a procesar, puesto que el mensaje almacenado en el Evento_TX no puede ser escuchado por el nodo vecino. Si además el evento vecino de tipo TRANSMISION, TRANSMISION_EVENTO, TRANSMISION_ACK o RETRANSMISION, se borra el identificador del nodo que generó el evento a procesar de la lista de identificadores de nodos vecinos del nodo vecino

por la misma razón anterior. En la Figura 4.16 se puede observar este caso, en la que el evento a procesar es el superior y el evento vecino el inferior en cada caso.

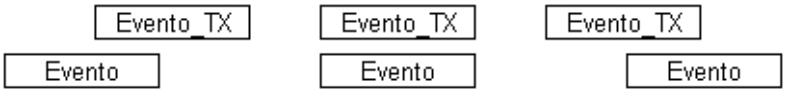


Figura 4.16: *Pérdida del mensaje porque el nodo vecino no está escuchando durante todo el intervalo de transmisión.*

- Un evento a procesar de tipo TRANSMISION, TRANSMISION_EVENTO, TRANSMISION_ACK o RETRANSMISION cuyo fin es mayor o igual que el fin del evento vecino, siendo el evento vecino de tipo RECEPCION o DORMIDO. En este caso, se borra el identificador del nodo que generó el evento a procesar de la lista de identificadores de nodos vecinos del nodo vecino, puesto que el mensaje almacenado en el Evento_TX no puede ser escuchado por el nodo vecino. En la Figura 4.17 se puede observar este caso, en la cual el evento a procesar es el superior y el evento vecino el inferior en cada caso.

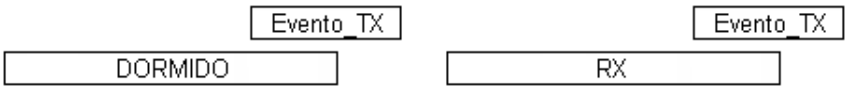


Figura 4.17: *Pérdida del mensaje porque el nodo vecino no está escuchando durante todo el intervalo de transmisión.*

- Un evento a procesar de tipo RECEPCIÓN o DORMIDO cuyo inicio es igual que el inicio del evento vecino y cuyo fin es igual que el fin del evento vecino. Si el evento vecino es de tipo RECEPTOR o DORMIDO, se elimina tanto el identificador del nodo que generó el evento vecino de la lista de identificadores de nodos vecinos del evento a procesar como el identificador del nodo que generó el evento a procesar de la lista de identificadores de nodos vecinos del evento vecino. En la Figura 4.18 se puede ver esta situación, en la cual el evento a procesar es el superior y el evento vecino el inferior en cada caso.

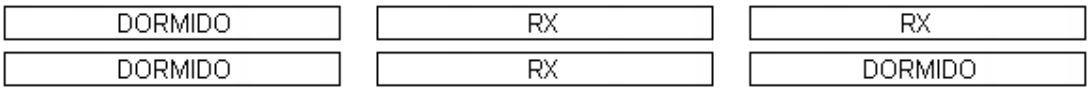


Figura 4.18: *Procesado de eventos dormido y recepción.*

- Un evento a procesar de tipo RECEPCIÓN o DORMIDO cuyo fin es mayor o igual que el fin del evento vecino. Si el evento vecino es de tipo RECEPTOR o DORMIDO se elimina el identificador nodo que generó el evento a procesar de la lista de identificadores de nodos vecinos del evento vecino. En la Figura 4.19 se puede observar esta situación, en la cual el evento a procesar es el superior y el evento vecino el inferior en cada caso.

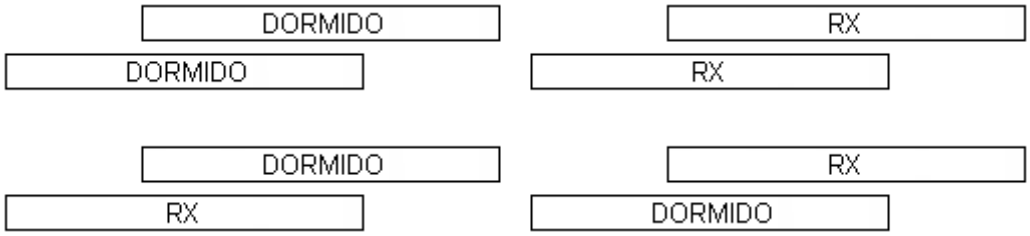


Figura 4.19: *Procesado de eventos dormido y recepción.*

- Un evento a procesar de tipo RECEPCIÓN o DORMIDO cuyo fin es menor o igual que el fin del evento vecino. Independientemente del tipo del evento vecino, se elimina el identificador del nodo que generó el evento vecino de la lista de identificadores de nodos vecinos del evento a procesar. En la Figura 4.20 se puede observar esta situación, en la cual el evento a procesar es el superior y el evento vecino el inferior en cada caso.

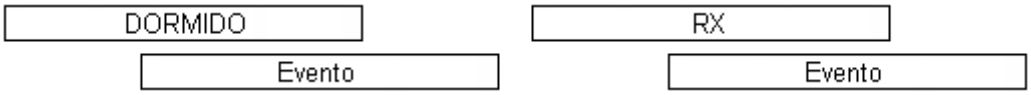


Figura 4.20: *Procesado de eventos dormido y recepción.*

Cuando un evento de un nodo tiene vacía su lista de identificadores de nodos vecinos que están pendiente de procesar, dicho evento se puede eliminar de la lista de eventos del nodo, puesto que se ha comprobado la influencia que tiene dicho evento en los eventos de sus nodos vecinos. En el caso de que la lista de eventos de un nodo vaya a quedarse vacía (se va a eliminar el único evento restante en la lista), se generarán uno o varios eventos nuevos dependiendo del tipo del evento restante en la lista.

Todo el proceso explicado anteriormente se repite para cada nodo del árbol de eventos, para cada evento de la lista de eventos de cada nodo y para cada identificador de nodo vecino de la lista de los mismos que tiene cada evento.

4.4. Descripción de los ficheros utilizados para la simulación

En esta sección se hará una explicación del contenido y formato de los ficheros utilizados por la maqueta de simulación para llevar a cabo su tarea. Los ficheros utilizados son tres: el fichero de configuración (*WSN.conf*), el fichero de datos (WSN.dat) y el fichero de simulación (WSN.sim).

4.4.1. Fichero de configuración

En el fichero *WSN.conf* se definen los parámetros necesarios para el correcto funcionamiento del programa. El usuario puede crear este fichero si conoce el formato del mismo. Si al inicio de la ejecución el fichero de configuración no existe, se crea uno con los parámetros por defecto de la ejecución.

En este fichero las líneas empiezan por # son comentarios. Como se puede apreciar en el ejemplo del fichero de configuración, dicho archivo está muy comentado y es casi autoexplicativo. Las variables definidas en dicho fichero se utilizan para modificar el comportamiento del simulador. Se usa una tabulación para la separación entre el nombre de las variables y el valor o valores que toman. El formato general para definir una variable se ve en la Figura 4.21. Dichas variables se explican a continuación.

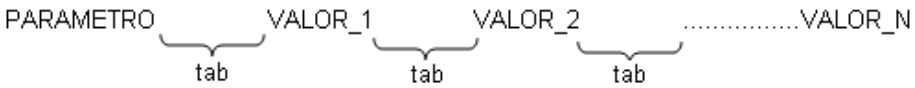


Figura 4.21: *Formato para definir una variable en el fichero de configuración.*

Ejemplo del fichero de configuración

```
#Numero de nodos
NUM_NODOS 6

#Numero de vecinos
NUM_VECINOS 50

#Anchura maxima del recinto en metros
MAX_ANCHURA 50
```

```
#Longitud maxima del recinto en metros
MAX_LONGITUD 50

#Consumo cuando el nodo esta dormido en mA
CONSUMO_DORMIDO 0.390000

#Consumo cuando el nodo esta activo (TX/RX) en mA
CONSUMO_ACTIVO 44.000000

#Batería inicial de cada nodo en mA
BAT_INICIAL 3000.000000

#Duracion de un paquete de datos en segundos (31 bytes que se transmiten a 250
Kbps)
DUR_TRANSMISION_DATOS 0.000992

#Duracion de un paquete ACK en segundos (11 bytes que se transmiten a 250 Kbps)
DUR_TRANSMISION_ACK 0.000044

#Periodo de cada nodo compuesto por tiempo activo mas tiempo dormido en segundos
PERIODO 100.000000

#Porcentaje de tiempo de escucha (0-100 \%) sobre el periodo
PORC_ESCUCHA 40.000000

#Porcentaje de transmisiones (0-100 \%) sobre el periodo
PORC_TRANSMISIONES 5.000000

#Probabilidad de ocurrencia de sucesos a sensar
PROB_EVENTO 0.010000
```

```
#Numero de rutas maximo en cada nodo
NUM_RUTAS_MAX 5

#Tiempo final en segundos en la simulacion del protocolo en cuestion
TIEMPO_FIN_SIMULACION 400000.000000

#Numero de nodos sumideros en la red
NUM_SUMIDEROS 1

#Protocolo de encaminamiento a simular (SIN_PROT_ENCAM, AODV)
PROT_ENCAMINAMIENTO AODV

#Se simula como varia el/los parametro/s de salida con respecto al porcentaje
de escucha desde el valor inicial a incrementos hasta el valor final
VAR_PORC_ESCUCHA 10.000000 10.000000 90.000000 THROUGHPUT BAT_MEDIA RETARDO_ME-
DIO

#Se simula como varia el/los parametro/s de salida con respecto al periodo des-
de el valor inicial a incrementos hasta el valor final
VAR_PERIODO 100.000000 100.000000 1000.000000 THROUGHPUT BAT_MEDIA RETARDO_ME-
DIO

#Se simula como varia el/los parametro/s de salida con respecto a la probabili-
dad de ocurrencia de sucesos a sensar en el periodo desde el valor inicial a
incrementos hasta el valor final
VAR_PROB_EVENTO 0.100000 0.100000 0.900000 THROUGHPUT BAT_MEDIA RETARDO_ME-
DIO

#Se simula como varia el/los parametro/s de salida con respecto al tiempo des-
de el valor inicial a incrementos hasta el valor final
VAR_TIEMPO 0.000000 3600.000000 500000.000000 BAT_MEDIA
```

```
#Se simula como varia el/los parametro/s de salida con respecto al numero de
nodos sumideros desde el valor inicial a incrementos hasta el valor final
VAR_NUM_SUMIDEROS 1 1 4 THROUGHPUT BAT_MEDIA RETARDO_MEDIO
```

La variable NUM_NODOS define el número de nodos que componen la red de sensores inalámbricos. Este número de nodos tiene que coincidir con el número de nodos definido en el fichero de datos (WSN.dat).

NUM_VECINOS permite declarar el número máximo de vecinos que puede tener un nodo. Como se explicó en la Sección 4.3.12, si el nodo tiene un mayor número de vecinos que el especificado en esta variable, dicho nodo se quedará con los vecinos con distancias menores respecto a él.

El parámetro MAX_ANCHURA establece la anchura máxima en metros del recinto a sensar. De forma semejante, la variable MAX_LONGITUD declara la longitud máxima del recinto en metros.

El consumo de cada nodo viene definido mediante dos variables. CONSUMO_DORMIDO establece el consumo en mA cuando el nodo está dormido y CONSUMO_ACTIVO define el consumo en mA cuando el nodo está activo, ya sea por que está escuchando el medio o transmitiendo. Además, BAT_INICIAL declara la capacidad de batería en mA con la que los nodos comienzan su vida.

La duración de un paquete de datos en segundos viene declarada por el parámetro DUR_TRANSMISION_DATOS, y la duración de un paquete ACK por la variable DUR_TRANSMISION_ACK.

Para definir los tiempos de duración en los que el nodo está escuchando, transmitiendo o durmiendo se hace uso del período de nodo. El período de nodo está compuesto por el tiempo en el cual el nodo está dormido más el tiempo en el cual el nodo está activo (escuchando el medio o transmitiendo). La variable PERIODO define la duración del mismo en segundos. PORC_ESCUCHA declara el porcentaje del tiempo sobre el período de nodo en el cual el nodo está escuchando y PORC_TRANSMISIONES el porcentaje de tiempo sobre el período en el que el nodo está transmitiendo. La duración de dormido será el porcentaje restante del período de nodo tras haber restado los dos porcentajes anteriores al 100 % de dicho período.

El parámetro PROB_EVENTO es utilizado para definir la probabilidad de ocurrencia de sucesos a sensar. Tras la detección, el nodo intentará informar a su nodo sumidero mediante un

mensaje de datos.

Otra variable utilizada es el número de rutas máximo en cada nodo, que viene definida por NUM_RUTAS_MAX. Este parámetro obliga a cada nodo a borrar una ruta de su tabla de encaminamiento si dicha tabla tiene almacenadas un número de rutas igual a dicho parámetro.

TIEMPO_FIN_SIMULACION establece el tiempo final en segundos en la simulación del protocolo en cuestión. Este tiempo es utilizado cuando no se realiza la simulación definida por VAR_TIEMPO, ya que esta simulación tiene definido su propio tiempo de fin de la simulación.

La variable NUM_SUMIDEROS define el número de nodos sumideros en la red.

El número de veces que se repite la simulación para evitar la aleatoriedad de los resultados viene definido por NUM_ITERACIONES.

El protocolo de encaminamiento a simular se declara con la variable PROT_ENCAMINAMIENTO. En este proyecto se ha implementado el protocolo de enrutamiento *AODV*. También se puede simular cómo funciona la maqueta de simulación estableciendo PROT_ENCAMINAMIENTO a un valor SIN_PROT_ENCAM.

Los parámetros de salida de las simulaciones son el *throughput*, el retardo medio en la entrega de datos de la red y la batería media restante en la red. Cada simulación viene definida por la variación de la variable a simular (VAR_PERIODO, VAR_PROB_EVENTO, VAR_TIEMPO y VAR_NUM_SUMIDEROS), el valor inicial, el incremento y el valor final de dicha variable. También se definen los parámetros de salida que quieren ser almacenados en el fichero de simulación.

Las simulaciones que puede realizar el usuario son las siguientes:

- VAR_PERIODO: cómo varía/n el/los parámetro/s de salida con respecto al porcentaje de escucha desde el valor inicial de dicho porcentaje a incrementos hasta el valor final.
- VAR_PROB_EVENTO: cómo varía/n el/los parámetro/s de salida con respecto a la probabilidad de ocurrencia de sucesos a sensar desde el valor inicial de dicha probabilidad a incrementos hasta el valor final.
- VAR_TIEMPO: cómo varía la batería media restante en la red con respecto al tiempo desde el valor inicial en el tiempo a incrementos hasta el valor final.
- VAR_NUM_SUMIDEROS: cómo varía/n el/los parámetro/s de salida con respecto al número de nodos sumideros desde el valor inicial de dicho número de sumideros a incrementos

hasta el valor final.

4.4.2. Fichero de datos

El fichero *WSN.dat* permite cargar las posiciones de los nodos en el emplazamiento a sensar. Dichas posiciones vienen definidas mediante coordenadas (x, y). Este fichero, en caso de que no exista, se crea al comienzo de la ejecución. El usuario puede crear uno si se ciñe al formato del mismo. Un ejemplo de este fichero se puede ver a continuación.

Ejemplo del fichero de datos

```
NUM_NODOS 6
MAX_ANCHURA 50
MAX_LONGITUD 50

0 25
16 10
30 10
50 25
30 40
16 40
```

En el inicio del fichero de datos se definen el número de nodos cuyas posiciones están definidas en el fichero, la anchura máxima y la longitud máxima en metros del recinto. Estas variables permiten detectar si el fichero de configuración y el fichero de datos están en concordancia, es decir, en el archivo *WSN.conf* se definen un número de nodos y un emplazamiento que se corresponden con los datos del archivo *WSN.dat*. Tras definir las tres variables anteriores se deja una línea en blanco y, posteriormente, se escriben las posiciones de los nodos en la red como posición x y posición y separadas por un tabulador.

4.4.3. Fichero de simulación

Este fichero (*WSN.sim*) recoge toda la información sobre los parámetros de salida de las simulaciones especificadas en el fichero de configuración. Dicho fichero se borra en cada ejecución, por tanto, si se quieren procesar los datos de salida habrá que hacer una copia del fichero en otro directorio. Cada parámetro simulado comienza con un título en mayúsculas con el nombre

de dicha simulación. A continuación, desde el valor inicial de dicho parámetro hasta el valor final a intervalos del valor especificado como intervalo en el fichero de configuración, se van almacenando en el fichero de simulación los parámetros de salida especificados. El formato del fichero se puede ver en el ejemplo de dicho fichero que se encuentra a continuación.

Ejemplo del fichero de simulación

```
SIMULACION PORC_ESCUCHA
Porc_escucha: 2
Tasa efectiva de transporte: 0.020668
Bateria Media: 2854.68
Retardo Medio: 0.00122457

Porc_escucha: 4
Tasa efectiva de transporte: 0.0414373
Bateria Media: 2752.7
Retardo Medio: 0.00321874

Porc_escucha: 6
Tasa efectiva de transporte: 0.0607783
Bateria Media: 2650.71
Retardo Medio: 0.00530164

Porc_escucha: 8
Tasa efectiva de transporte: 0.0798638
Bateria Media: 2548.72
Retardo Medio: 0.0246397

Porc_escucha: 10
Tasa efectiva de transporte: 0.0983095
Bateria Media: 2446.73
Retardo Medio: 0.0354809
```

4.5. Pruebas realizadas de la maqueta de simulación

Para comprobar el funcionamiento de la maqueta de simulación se han realizado tres pruebas de la misma sin ningún protocolo de encaminamiento implementado en los nodos. Los parámetros de salida son el *throughput* y la batería media restante en la red. En estas simulaciones no tiene sentido representar el retardo, puesto que el envío de mensajes será entre nodos vecinos y no habrá comunicaciones de más de un salto (ver final de la Sección 4.3.12). En estas pruebas se ha utilizado una red formada por seis nodos situados como muestra la Figura 4.22. La red se ha diseñado para que cada nodo tenga dos vecinos (el nodo 1 tiene como vecinos al 2 y al 6, el nodo 2 tiene como vecinos al 1 y al 3 y así sucesivamente). El número máximo de vecinos para cada nodo es 50. El recinto tiene una anchura de 50 metros y una longitud de 50 metros. El consumo cuando el nodo está dormido es de 390 μAh y cuando está activo 44 mAh. La batería inicial de cada nodo es de 3000 mAh. El tamaño de los paquetes de datos es de 31 bytes, por tanto, como se transmiten a una tasa de 250 $Kbps$, su duración es de 992 μs. El tamaño de los paquetes de reconocimiento positivo es de 11 bytes, siendo su duración 44 μs. El periodo de nodo es de 100 segundos, siendo el porcentaje de escucha del 40 % y el de transmisiones del 5 %. La probabilidad de ocurrencia de sucesos a sensar es de 0.01. El tiempo final de la simulación es de 400000 segundos. Para ejecutar la simulación, una vez configurados los ficheros necesarios, bastaría con lanzar desde una consola linux el ejecutable llamado *PFC*, almacenado en la carpeta «Programa» del cd-rom adjunto al proyecto.

4.5.1. Variación del porcentaje de escucha

En esta simulación se ha ido variando el porcentaje de escucha desde el 10 % hasta el 90 %. En la Figura 4.23 se puede observar que al aumentar el porcentaje de escucha va disminuyendo de forma lineal la batería media restante en la red. Al aumentar el porcentaje de escucha, el nodo pasa más tiempo escuchando el medio y menos tiempo dormido. Por tanto, el consumo de batería es mayor al aumentar el tiempo activo y disminuir el tiempo de dormido.

La relación entre el porcentaje de escucha y el *throughput* se observa en la Figura 4.24. El *throughput* aumenta de forma lineal al aumentar el porcentaje de escucha. La explicación es sencilla, ya que al aumentar el período de escucha se podrá recibir un mayor número de mensajes.

Para terminar esta sección se muestra en una gráfica conjunta la Figura 4.23 y la Figura

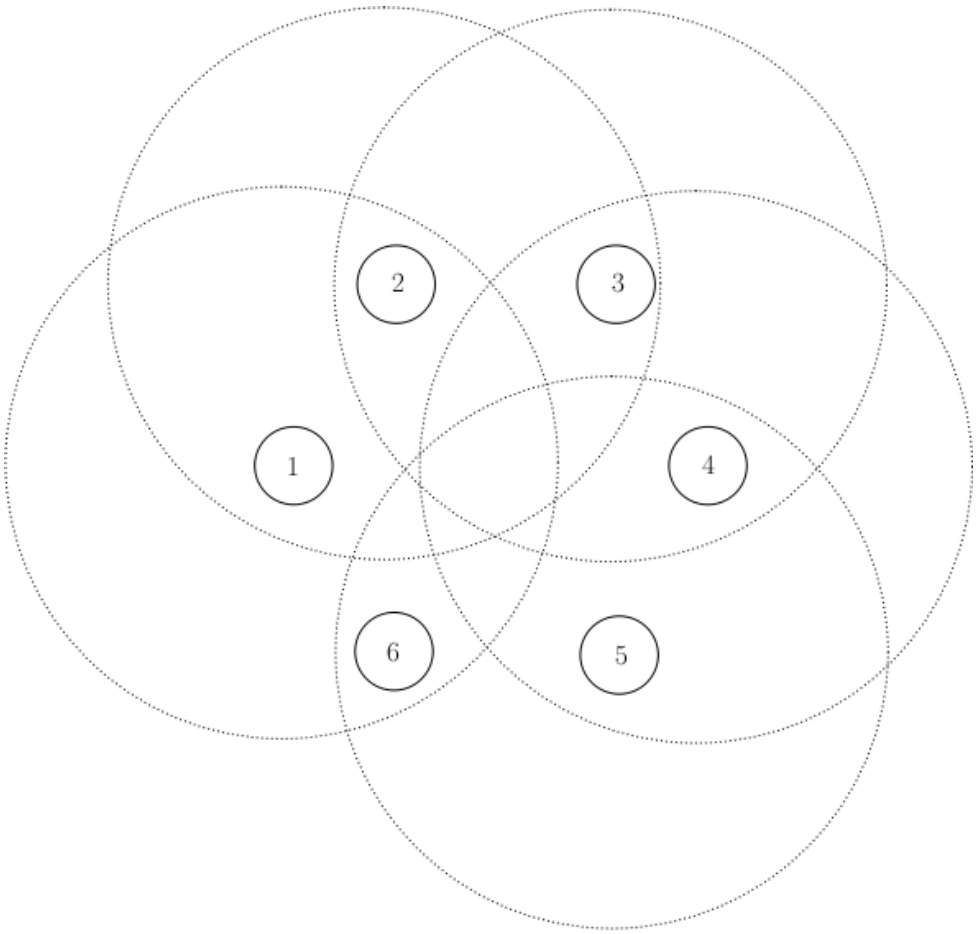


Figura 4.22: *Despliegue de nodos.*

4.24. En esta gráfica (Figura 4.25) se puede observar el punto de compromiso entre obtener el mayor *throughput* y el menor consumo de batería. Para un porcentaje de escucha del 32.88 % se obtiene el mayor *throughput* (35 %) con el menor consumo de batería (batería media restante igual a 849.9 mA).

4.5.2. Variación del período de nodo

Para ver cómo influye el período de nodo en el *throughput* y en la batería media restante en la red, se ha ido variando éste desde 100 hasta 1000 segundos.

Al aumentar el período de nodo, la batería media restante en la red va aumentando muy lentamente. El período de un nodo está formado por ciclo de escucha más ciclo de transmisiones (si las hay) más ciclo dormido, y al aumentar el período, aumentan los tres ciclos anteriores. Se debe tener en cuenta que en esta simulación se ha utilizado un período de dormido mayor

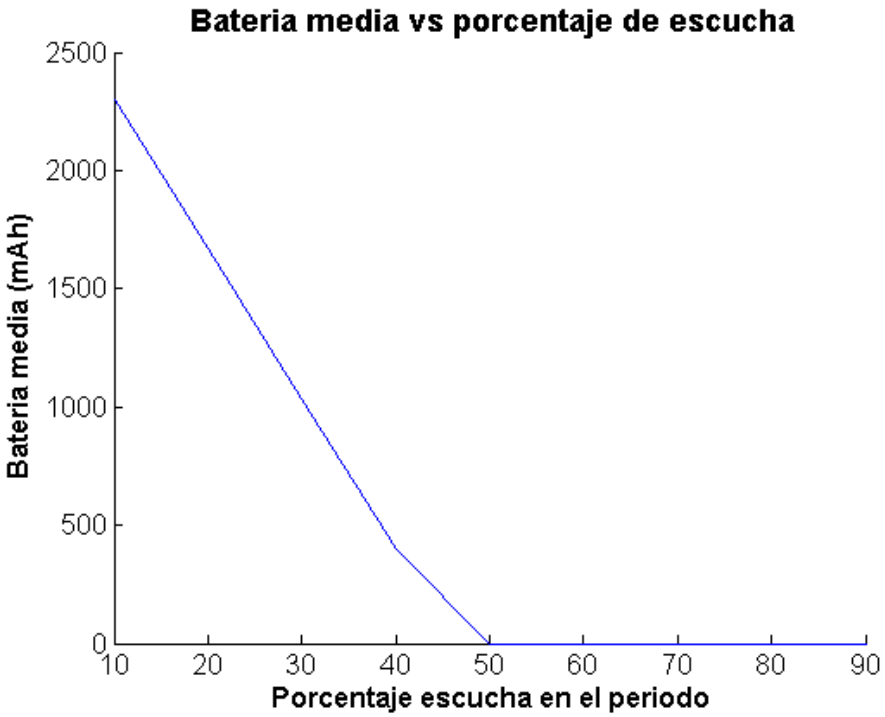


Figura 4.23: *Batería media restante en la red vs porcentaje de escucha.*

que el de escucha más transmisiones, cosa que ocurre normalmente en las redes de sensores inalámbricos. Para un mismo tiempo de finalización de la simulación, el nodo pasará mayor tiempo dormido a medida que se aumente el período, debido a que el porcentaje de ciclo dormido es mayor que el de escucha más transmisiones. Este fenómeno se puede apreciar en la Figura 4.26.

Al aumentar el período de nodo, aumentan de forma proporcional el tiempo de escucha, el tiempo de dormido y el tiempo de transmisiones. Por tanto, al no verse afectada la proporción entre estos tiempos, el *throughput* se mantiene más o menos constante. Esto se puede observar en la Figura 4.27.

4.5.3. Variación de la probabilidad de ocurrencia de sucesos sensados

La probabilidad de ocurrencia de sucesos sensados tiene una relación lineal con el tráfico entrante. Al aumentar la probabilidad de ocurrencia de sucesos sensados, aumenta también el tráfico entrante. En esta sección se ofrece la relación entre el tráfico entrante con el *throughput* y la batería media restante en la red.

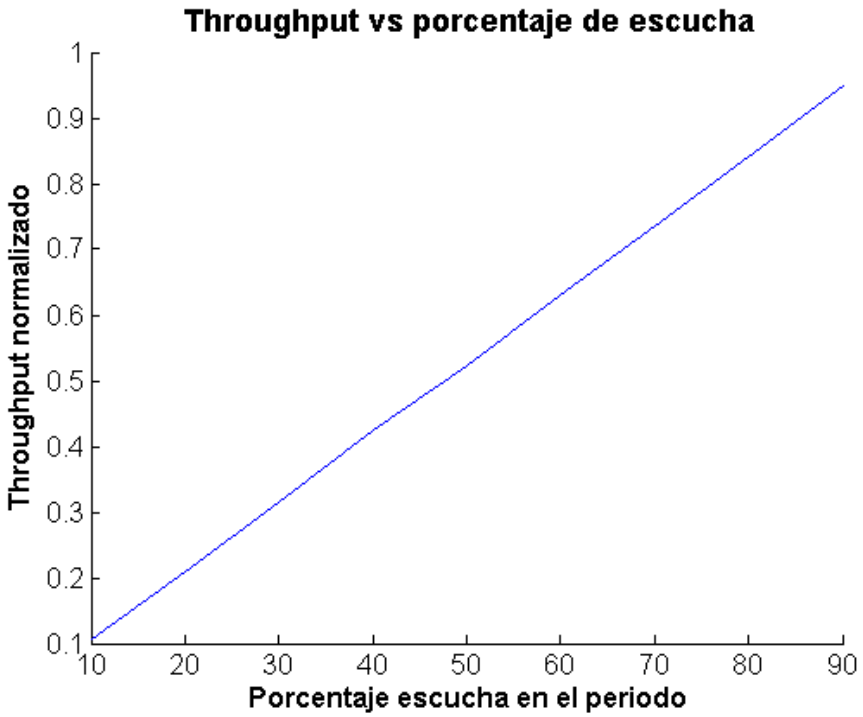


Figura 4.24: *Throughput vs porcentaje de escucha.*

La batería media va disminuyendo muy lentamente conforme aumenta el tráfico entrante. Esto se debe a que al aumentar el tráfico entrante habrá un mayor número de transmisiones y un mayor consumo de batería (ver Figura 4.28).

Al aumentar el tráfico entrante, el *throughput* va aumentando hasta un máximo (0.2 %) en el que empieza a bajar. La subida se debe a que al aumentar ligeramente el tráfico entrante se reciben mayor número de mensajes y hay pocas colisiones. Pero cuando se aumenta demasiado dicho tráfico, el número de transmisiones es tan grande que hay un gran número de colisiones y el *throughput* disminuye. Esto se puede observar en la figura 4.29.

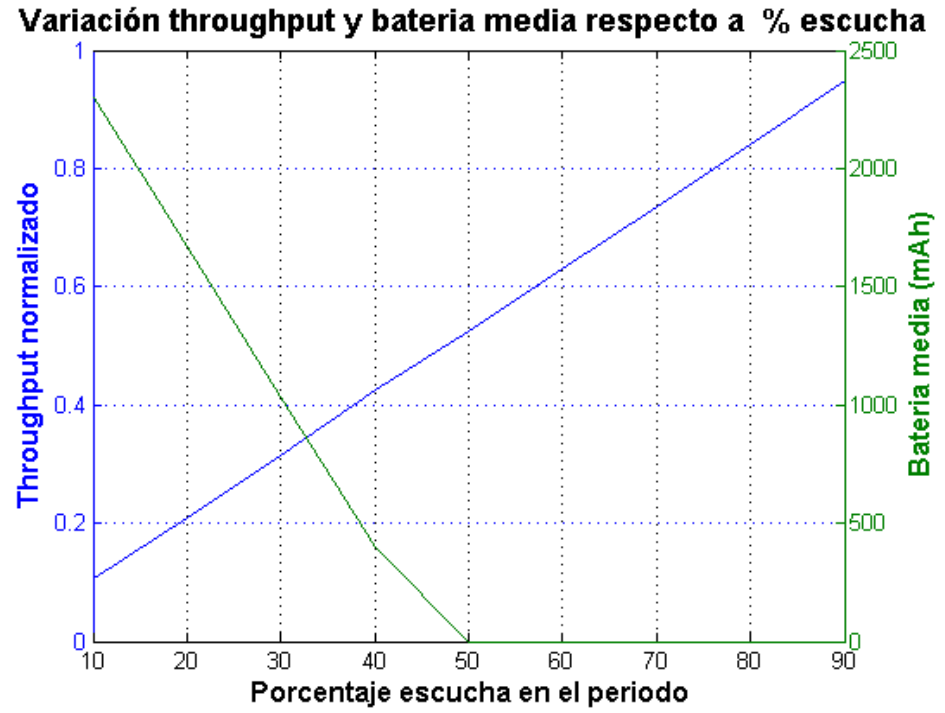


Figura 4.25: *Variación del throughput y de la batería media respecto al porcentaje de escucha.*

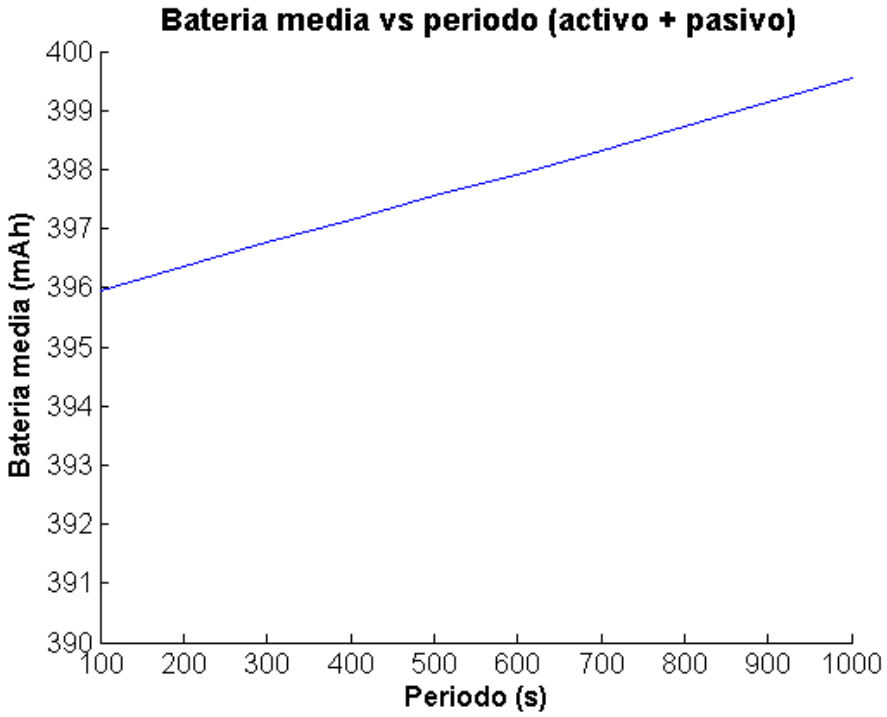


Figura 4.26: *Batería media restante en la red vs período de nodo.*

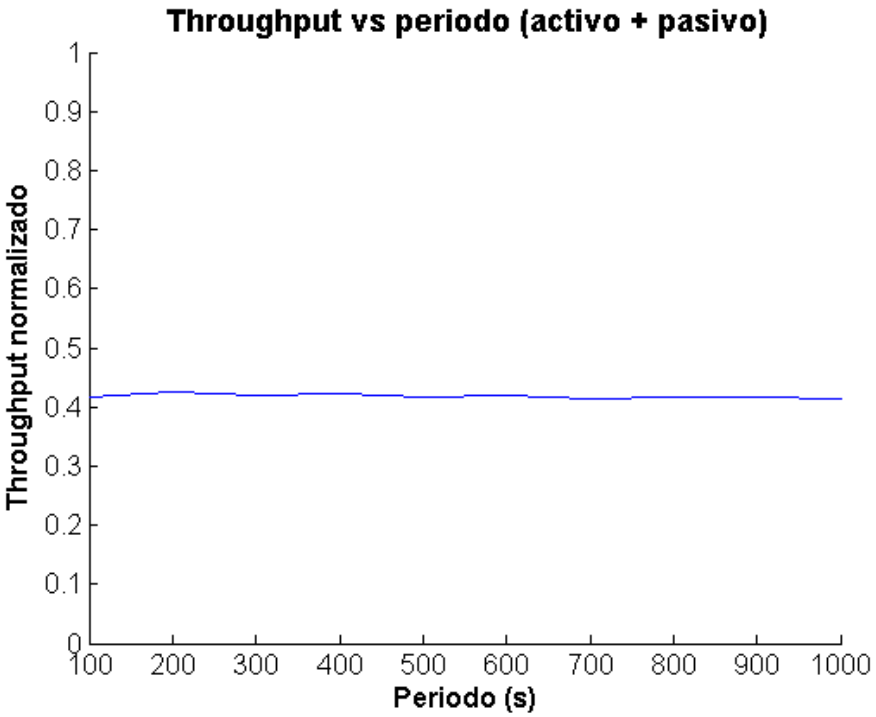


Figura 4.27: *Throughput vs período de nodo.*

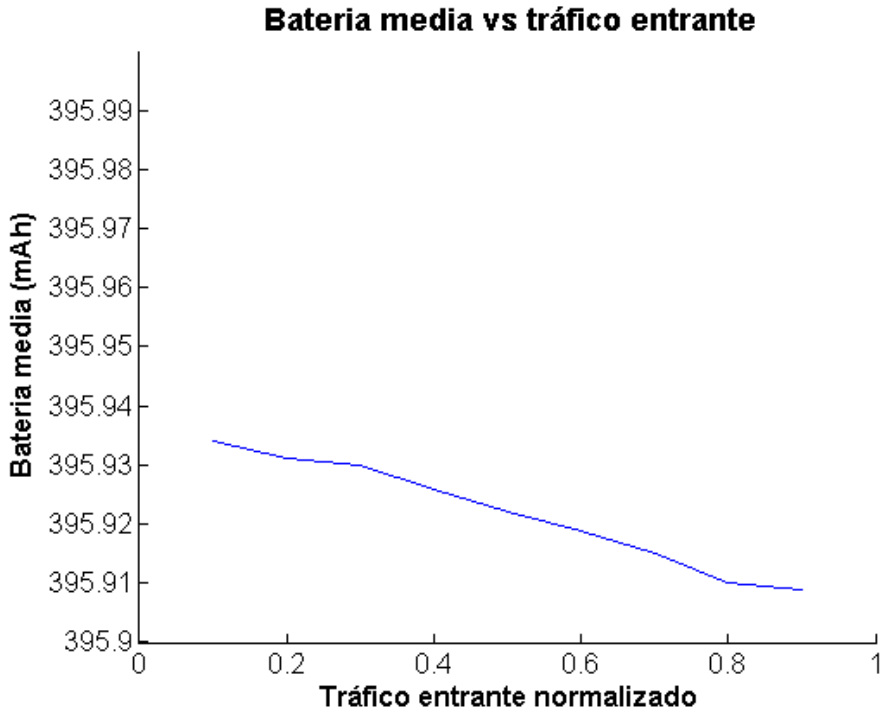


Figura 4.28: *Batería media restante en la red vs tráfico entrante.*

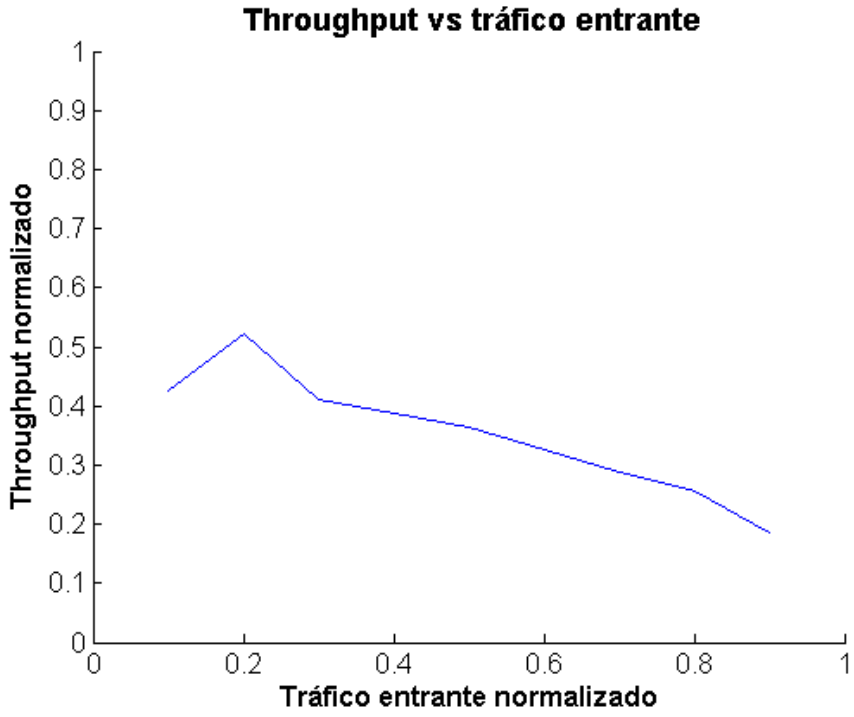


Figura 4.29: *Throughput vs tráfico entrante.*

Capítulo 5

APLICACIÓN DE LA MAQUETA DE SIMULACIÓN AL ANÁLISIS DE PROTOCOLOS DE ENRUTAMIENTO

Al ser *Zigbee* [68] una de las tecnologías más utilizadas en redes de sensores ad-hoc, el protocolo de encaminamiento elegido, para simularlo en la maqueta implementada, ha sido el protocolo usado por dicha tecnología. El algoritmo de enrutamiento utilizado por Zigbee en la capa de red es el *Ad-hoc On-demand Distance Vector (AODV)*, el cual permite crear en modo dinámico las rutas en el interior de la red, garantizando la posibilidad de encontrar caminos alternativos en el caso de que pudiese fallar un enlace en la comunicación.

5.1. Protocolo de encaminamiento *AODV*

5.1.1. Visión general

El protocolo *AODV* (*Ad-Hoc On-demand Distance Vector*) fue creado en 1999 por Charles E. Perkins, del grupo de desarrollo avanzado de Sun Microsystems, y Elizabeth M. Royer, de la Universidad de California, Santa Barbara [73]. El propósito fue diseñar un protocolo de encaminamiento para redes ad-hoc formadas por nodos móviles tomando como punto de partida el protocolo *DSDV*, con el fin de solventar sus deficiencias: el alto número de envíos en modo broadcast y la latencia de transmisión [73, 17]. Dicho protocolo se describe en la RFC (Request

For Comments) número 3561 que se publicó en Julio de 2003 con la categoría de experimental [72].

Como se ha indicado en la Sección 3.3, el protocolo *AODV* perteneciente a la familia de los protocolos reactivos, ya que el proceso de búsqueda de rutas se inicia sólo cuando un nodo necesita enviar información a otro nodo y desconoce cómo acceder a él. Además, está basado en la familia de algoritmos de vector de distancias y puede transmitir en modo unicast y multicast. Su modelo de comunicación es multicanal y tiene direccionamiento plano y estructura uniforme. Este protocolo utiliza poca señalización de control y funciona sólo con enlaces bidireccionales.

El protocolo *AODV* combina técnicas extraídas de los protocolos *DSDV* y *DSR*, dando lugar a un algoritmo que usa el ancho de banda de manera eficiente y que responde con rapidez a los cambios en la red, al tiempo que garantiza la ausencia de bucles [73, 72, 56]. Con el fin de mantener sólo la información de encaminamiento más reciente, el protocolo *AODV* toma prestado de su predecesor, el protocolo *DSDV*, el concepto de número de secuencia. Tanto en el protocolo *AODV* como en el protocolo *DSDV*, cada nodo se encarga de mantener su propio contador o número de secuencia. Este número no es más que un valor entero, que cada nodo incrementa monótonamente antes de generar un mensaje de control para copiarlo en éste antes de enviarlo. De manera complementaria al número de secuencia, cada nodo se distingue por un identificador único dentro de la red. De este modo, con la pareja de valores formada por el identificador del nodo y el número de secuencia, es posible distinguir la información válida de la anticuada. Si un nodo recibe dos paquetes con el mismo identificador de nodo pero con diferentes números de secuencia, la información más reciente será la incluida en el paquete de mayor número de secuencia. Puesto que cada nodo es responsable de su propio contador, no es necesario mantener un reloj único y común a toda la red, lo cual simplifica enormemente la implementación del protocolo [73]. El uso de estos números de secuencia garantiza la ausencia de bucles en todo momento y evita problemas como el de la «cuenta al infinito» [56, 72]. En los protocolos que no incluyen ningún mecanismo de prevención, esta anomalía se produce cuando cae un enlace, afectando al menos a dos nodos. Considérese un primer nodo con dos enlaces, uno de ellos le conecta con un segundo nodo y el otro ha dejado de estar operativo. El segundo nodo, que desconoce que ha fallado el enlace, desea acceder a un destino al otro lado del enlace averiado, por lo que tendría que pasar por el primero de los nodos. Sin embargo, como el primer nodo no puede usar el enlace dañado, reenvía el paquete al segundo. Cada nodo cree equivocadamente que puede llegar a su destino a través del otro, por lo que ambos degeneran en un bucle infinito. Este

problema de cuenta al infinito está típicamente asociado a los protocolos clásicos de vector de distancias, ya que con este modelo de información de estado, los nodos sólo conocen el siguiente salto y no mantienen la topología completa de la red [53].

Este protocolo emplea un mecanismo de descubrimiento de rutas en modo broadcast que también emplea el protocolo *DSR* aunque con ciertas modificaciones. En el protocolo *DSR*, es el nodo origen quien se encarga de calcular la ruta completa hasta el nodo destino [44]. Esto puede degradar la prestaciones de la red cuando ésta es muy extensa, ya que cada paquete incluye en su cabecera la secuencia de nodos por los que debe pasar desde el origen hasta el destino. Por el contrario, en el protocolo *AODV*, el camino se forma gracias a la información mantenida en las tablas de rutas de los nodos intermedios.

5.1.2. Información de encaminamiento

El protocolo *AODV* almacena la información de encaminamiento en forma de tablas de rutas. Cada uno de los nodos de la red tiene asociada su propia tabla, que tiene tantas entradas como destinos conoce el nodo. Una entrada en dicha tabla consta de los siguientes campos:

- Identificador del nodo destino: permite diferenciar el destino de cada ruta.
- Identificador de siguiente salto: identificador del nodo adyacente al que se debe enviar el paquete para llegar al destino deseado.
- Número de secuencia del nodo destino: número asociado al nodo destino, cuyo valor se obtiene de los mensajes de control. Se utiliza para distinguir entre información nueva e información antigua y de esta forma evitar formación de bucles y transmisiones de rutas antiguas.
- Indicador de validez del número de secuencia del nodo destino: si se pretende alcanzar un nodo destino y ha fallado uno de los enlaces implicados, o la ruta ha expirado, el número de secuencia asociado a ese nodo destino se marca como inválido.
- Otros indicadores sobre estado y rutas: por ejemplo, indicadores sobre si la ruta es o no válida, y en este último caso si es reparable, no es reparable y se debe buscar un camino alternativo, o bien, si está siendo reparada.
- Número de saltos: número de saltos necesarios para alcanzar el destino desde este nodo.

- Tiempo de vida de la ruta: tiempo en el que la ruta caduca o debe ser borrada. Evita que viajen paquetes perdidos por la red y que se utilicen enlaces de los que no se conoce su estado desde hace mucho tiempo.

Cuando un nodo de la red ad-hoc desea comunicarse con otro, lo primero que debe hacer es buscar en su tabla de encaminamiento si existe una ruta hacia este destino previamente calculada. En el caso de encontrarla no iniciaría ningún proceso de descubrimiento de ruta, supondría que la que tiene almacenada en su tabla de encaminamiento es correcta y está actualizada. En el caso contrario, comenzará el proceso de descubrimiento de ruta para encontrar un camino válido. Existe otro concepto conocido como mantenimiento de ruta, que sirve para detectar la rotura de un enlace a lo largo de una ruta.

5.1.3. Paquetes usados por *AODV*

5.1.3.1. Paquete RREQ

El mensaje RREQ (Route Request) se envía para solicitar una ruta hacia un determinado nodo destino. En la Figura 5.1 se muestra la estructura básica del mensaje RREQ, la cual contiene los siguientes campos:

- Tipo: indica el tipo del paquete.
- J: el flag de Join se usa cuando el nodo origen quiere participar en un grupo multicast.
- R: el flag de Reparación (Repair flag) se usa cuando un nodo quiere iniciar una reparación de dos partes del árbol Multicast que hayan sido desconectadas.
- G: el flag RREP Gratuito (Gratuitous RREP flag) indica si se debe enviar un RREP Gratuito en modo Unicast al nodo indicado en el campo Identificador del Destino Final.
- D: el flag de Sólo Destino (Destination Only flag) indica que tan sólo puede responder el destino a un paquete RREQ.
- U: el flag de Número de Secuencia Desconocido (Unknow Sequence Flag) indica que no se conoce el número de secuencia del nodo destino.
- Reservado: bits reservados para uso futuro. Si se envía como 0, se ignora en la recepción.

- Contador de saltos (Hop Count): el número de saltos desde el nodo origen al destino.
- RREQ ID: número de secuencia que junto con el identificador del nodo origen identifican unívocamente a un RREQ en particular.
- Identificador del Destino Final: identificador del nodo con el que se quiere establecer la comunicación.
- Número de Secuencia del Destino Final: último número de secuencia recibido en el nodo fuente de cualquier ruta hacia el destino.
- Identificador del Origen Inicial: identificador del nodo que ha originado la petición de ruta.
- Número de Secuencia del Origen Inicial: número de secuencia actual del nodo origen.

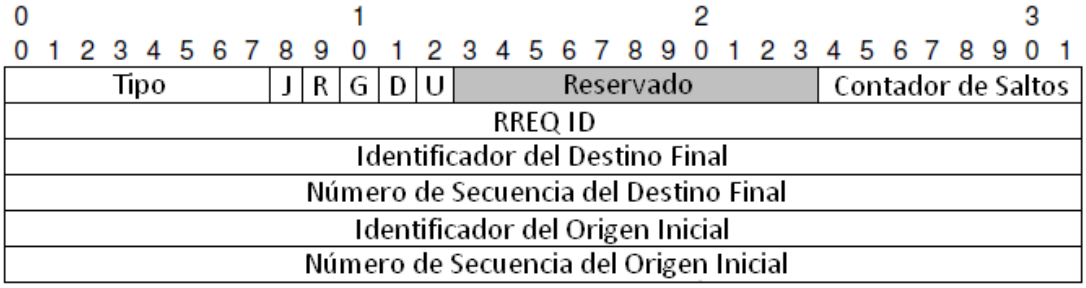


Figura 5.1: *Paquete RREQ.*

5.1.3.2. Paquete RREP

El mensaje RREP (Route Reply) se envía como contestación a un RREQ confirmando un establecimiento de ruta. Sigue el formato que se observa en la Figura 5.2 y los campos que lo forman se citan a continuación:

- Tipo: indica el tipo del paquete.
- R: el flag de Reparación (Repair flag) se usa cuando un nodo quiere iniciar una reparación de dos partes del árbol Multicast que hayan sido desconectadas.
- A: el flag de Reconocimiento de Peticiones (Acknowledgment Required) se activa cuando se quiere comprobar si un enlace es unidireccional.

- Reservado: bits reservados para uso futuro. Si se envía como 0, se ignora en la recepción.
- Longitud del Prefijo: si no es cero, especifica que el siguiente salto indicado puede ser usado, por nodos con el mismo prefijo, como si fuese el destino de la petición.
- Contador de saltos (Hop Count): el número de saltos desde el nodo origen al destino.
- Identificador del Destino Final: identificador del nodo con el que se quiere establecer la comunicación.
- Número de Secuencia del Destino Final: último número de secuencia recibido en el nodo fuente de cualquier ruta hacia el destino.
- Identificador del Origen Inicial: identificador del nodo que ha originado la petición de ruta.
- Tiempo de Vida (Lifetime): tiempo en milisegundos durante el cual los nodos que han recibido un RREP deben considerar válida la ruta.

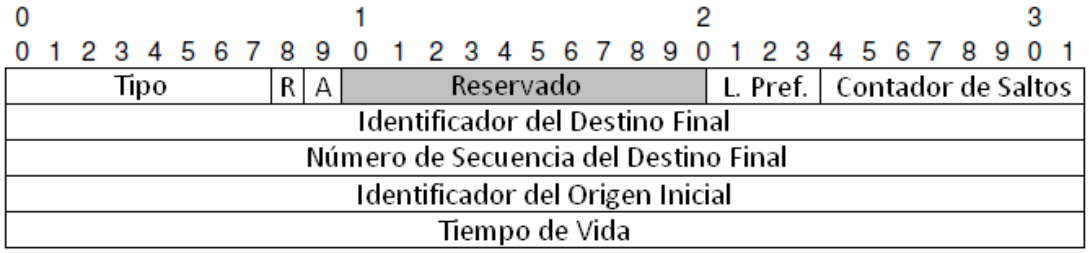


Figura 5.2: *Paquete RREP.*

5.1.3.3. Paquete RERR

El mensaje RERR (Route Error) se envía si un error en el enlace causa que uno o más destinos se vuelvan inalcanzables desde ciertos nodos vecinos. En la Figura 5.3 se muestra la estructura básica del mensaje RERR, la cual contiene los siguientes campos:

- Tipo: indica el tipo del paquete.
- Flag N: el flag de No Borrado (No Delete) se usa cuando se está realizando un Local Repair de un enlace y el nodo no debe borrar la ruta.
- Reservado: bits reservados para uso futuro. Si se envía como 0, se ignora en la recepción.

- Contador de Destinos: número de destinos inalcanzables que se encuentran en el mensaje. Como mínimo será 1.
- Identificador del Destino Inalcanzable: identificador del destino que ha resultado inalcanzable por culpa del error en el enlace.
- Número de secuencia del Destino Inalcanzable: último número de secuencia conocido, incrementado en 1, del destino que se encuentra en el campo Identificador del Destino Inalcanzable.

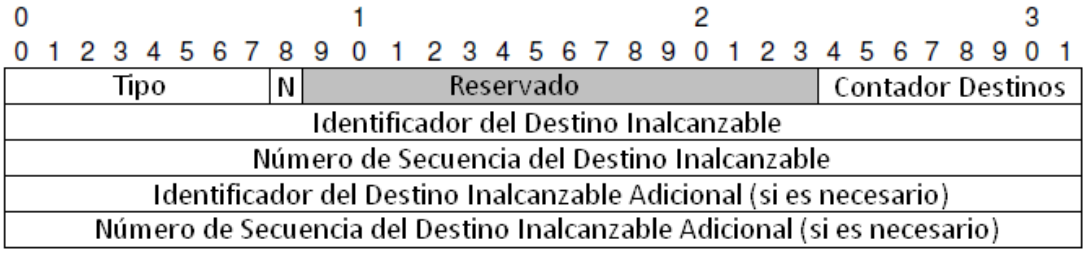


Figura 5.3: *Paquete RERR.*

5.1.4. Descubrimiento de rutas

El proceso de descubrimiento de rutas se inicia cuando un nodo origen desea comunicarse con otro nodo pero desconoce cómo acceder a él, es decir, no tiene información de encaminamiento en su tabla de rutas acerca de ese nodo destino. Para ello, se intercambian principalmente dos tipos de mensajes: mensajes de solicitud o petición de ruta (RREQ, Route Request) y mensajes de respuesta de ruta (RREP, Route Reply). En esta operación de búsqueda de rutas se pueden distinguir a su vez dos fases: la formación del camino de vuelta y la formación del camino de ida. La formación del camino de vuelta establece todos los itinerarios posibles desde el origen hasta el destino, trazados por el recorrido de los mensajes RREQ. La formación del camino de ida determina la ruta que finalmente seguirán los paquetes desde el nodo origen hasta el nodo destino una vez finalizado el descubrimiento de caminos [72, 73].

5.1.4.1. Formación del camino de vuelta

Cuando un nodo origen desea alcanzar un nodo destino y desconoce cómo acceder a él, genera un mensaje RREQ. En él se incluyen los identificadores y los números de secuencia de

los nodos origen y destino. Antes de enviar esta solicitud, el nodo origen incrementa su número de secuencia para evitar conflictos con peticiones anteriores. En el campo correspondiente al número de secuencia de destino, el nodo incluye el último valor aprendido, en caso de que ya hubiese solicitado esa ruta con anterioridad, o bien indica que es desconocido. El mensaje RREQ se difunde por inundación. La inundación es una técnica de envío de paquetes por la que cuando un nodo tiene información dirigida a un destino concreto, la transmite a sus vecinos. Si el nodo que la recibe no es el destinatario de esta información, la reenvía de nuevo. Este proceso continúa sucesivamente hasta alcanzar el nodo destino [71]. En complemento a la técnica de inundación y con el objeto de evitar un consumo excesivo del ancho de banda, el nodo origen emplea el algoritmo de búsqueda expansiva en anillo (*expanding ring search*). De acuerdo a este algoritmo, inicialmente el mensaje RREQ tiene asociado un valor pequeño de su tiempo de vida TTL (Time-To-Live), de tal manera que el mensaje se descarta cuando este tiempo expira. Si no se encuentra el destino antes de un plazo determinado, este valor se incrementa progresivamente en el envío de las posteriores solicitudes de rutas. Con el fin de que un nodo no permanezca eternamente intentando alcanzar un destino inaccesible, se tiene un número máximo de intentos, cuyo valor es típicamente dos [72]. La Figura 5.4 representa una red compuesta por seis nodos, en la que se indica mediante flechas el recorrido de los mensajes RREQ. El nodo origen (nodo 1) inicia el proceso de inundación con mensajes RREQ, que llegan a sus dos vecinos, quienes a su vez reenvían sucesivamente la solicitud. En este caso, se considera que ninguno de los nodos intermedios conoce el camino, por lo que la inundación se propaga hasta alcanzar el nodo destino (nodo 4).

Cada vez que un nodo recibe el mensaje RREQ, comprueba si es él el destino buscado o si al menos conoce cómo acceder a él. Si no es así, el nodo recoge un registro de la solicitud y vuelve a reenviar el mensaje RREQ a sus vecinos, continuando con el proceso de inundación. Los nodos toman nota de los mensajes RREQ recibidos para no reenviar la misma solicitud varias veces, ya que esto sobrecargaría la red de manera innecesaria. La Figura 5.5 ilustra la fase de formación de los caminos de vuelta. Puesto que los nodos intermedios anotan de dónde proviene la solicitud, se forman dos caminos de vuelta, representados con líneas azules. El proceso de inundación, y en consecuencia, la formación del camino de vuelta, se detiene cuando el nodo que recibe la solicitud es el nodo destino o conoce cómo llegar al mismo, dando lugar a la siguiente fase, la formación del camino de ida.

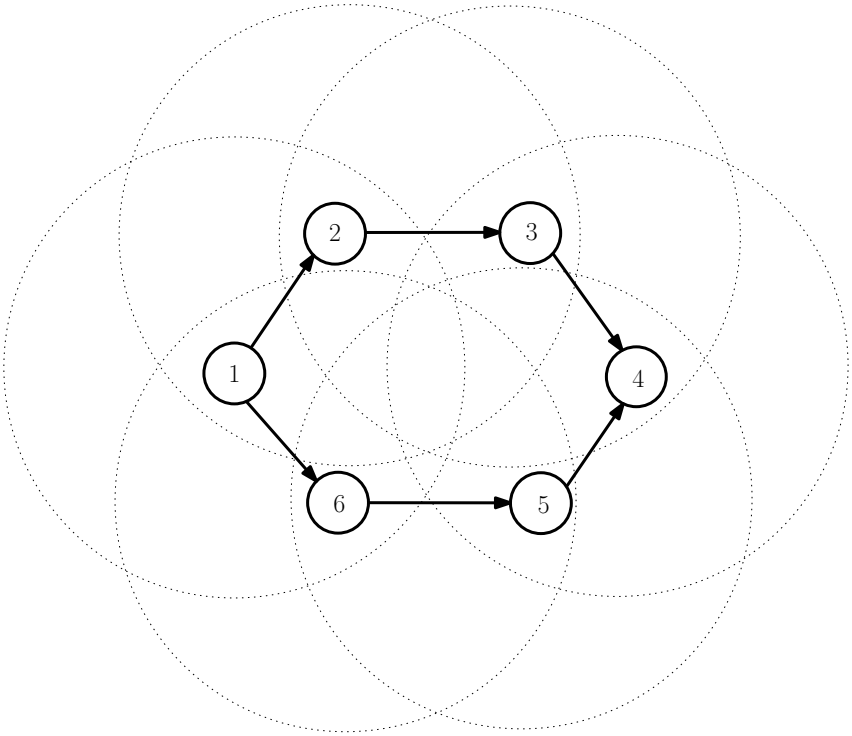


Figura 5.4: *Inundación con mensajes RREQ.*

5.1.4.2. Formación del camino de ida

Si el nodo que recibe la solicitud es el propio nodo destino o es un nodo intermedio que tiene una ruta activa (válida) hacia el destino, se genera un mensaje de respuesta de ruta. Se considera que un nodo intermedio tiene una ruta activa hacia el destino cuando el número de secuencia del nodo destino almacenado en la tabla de rutas es mayor o igual al número de secuencia del nodo destino de la solicitud. Cuando es el nodo destino quien genera la respuesta, incluye en el mensaje RREP como número de secuencia el valor máximo entre su propio número de secuencia y los números de secuencia de destino incluidos en los mensajes de solicitud de rutas. A diferencia de la solicitud, el mensaje RREP se reenvía de vuelta al origen de forma unicast. Un mensaje RREP siempre sigue el camino inverso de su mensaje RREQ correspondiente, por lo que los nodos típicamente asumen que los enlaces son bidireccionales. La Figura 5.6 detalla la trayectoria de las respuestas para completar la fase de formación del camino de ida, en la que se indica mediante flechas el recorrido de los mensajes RREP. En este caso, puesto que se supone que ninguno de los nodos intermedios conoce la ruta, es el nodo destino quien genera la respuesta, por lo que los dos mensaje RREP que llegan al nodo origen tienen el mismo número

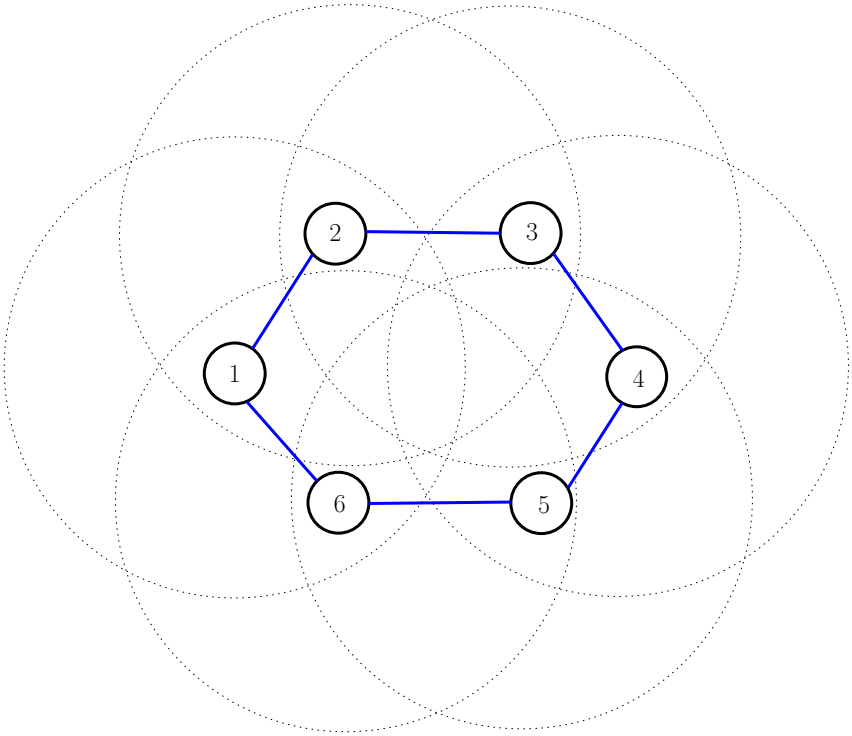


Figura 5.5: *Generación de caminos de vuelta potenciales.*

de secuencia de destino. Para escoger una de las dos rutas posibles se atiende al menor número de saltos. Como en este caso existe el mismo número de saltos por ambos caminos, se selecciona el camino cuyo RREP llegue antes (camino trazado en rojo por ejemplo).

Cuando los nodos intermedios por los que pasó previamente la solicitud reciben la respuesta de rutas, pueden verse en la necesidad de actualizar su tabla de rutas. Un nodo intermedio procede a la actualización de rutas en dos casos. En primer lugar, refresca su ruta si el nuevo número de secuencia asociado al nodo destino que se incluye en el mensaje RREP es mayor que el que figura en su tabla para ese destino. En segundo lugar, cuando ambos números de secuencia coinciden, se procede a la actualización cuando el número de saltos indicado en la respuesta es inferior al indicado en su tabla. Para computar el número real en función del número de saltos que aparece en la respuesta ha de sumarse una unidad para incluir al propio nodo. Los mensajes RREP redundantes o con un número de secuencia de destino menor se descartan automáticamente. Cuando finalmente el nodo origen recibe el mensaje de respuesta, guarda la ruta hacia el destino y puede comenzar a transmitir paquetes de datos. La Figura 5.7 muestra el camino de ida definitivo. Sobre él se muestra el recorrido de los paquetes de datos mediante

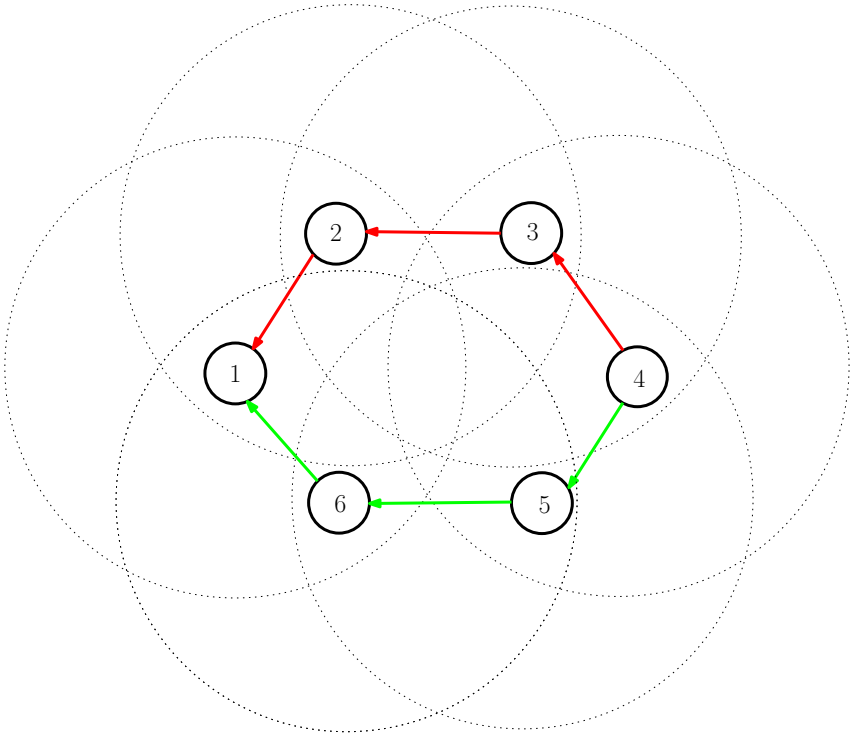


Figura 5.6: *Reenvío de mensajes RREP.*

flechas.

5.1.4.3. Mantenimiento de caminos

AODV se mueve en el medio radio, que es un entorno bastante hostil por sus continuas oscilaciones en la calidad del canal y sobretodo por la movilidad de los nodos, lo que provoca la rotura de los enlaces. Cuando una ruta es encontrada se le da un tiempo de vida y se considera útil hasta que este tiempo no expira. Esto se utiliza para no tener que iniciar un descubrimiento de ruta para cada mensaje de información que se quiere enviar.

El protocolo *AODV*, al igual que otros protocolos de encaminamiento, emplea mensajes *HELLO* o beacons para que los nodos anuncien a sus vecinos su pertenecencia a la red y, de esa manera, se pueda monitorizar en una ruta activa el estado del enlace hacia el siguiente salto. Los mensajes *HELLO* se envían de manera periódica, lo que permite detectar fallos de enlace. Cuando un nodo deja de recibir estos mensajes por parte de alguno de sus vecinos, puede concluir que el enlace ha dejado de estar operativo. En el momento en el que un nodo advierte un fallo en un enlace, difunde por broadcast un mensaje de error de ruta (RERR, Route Error)

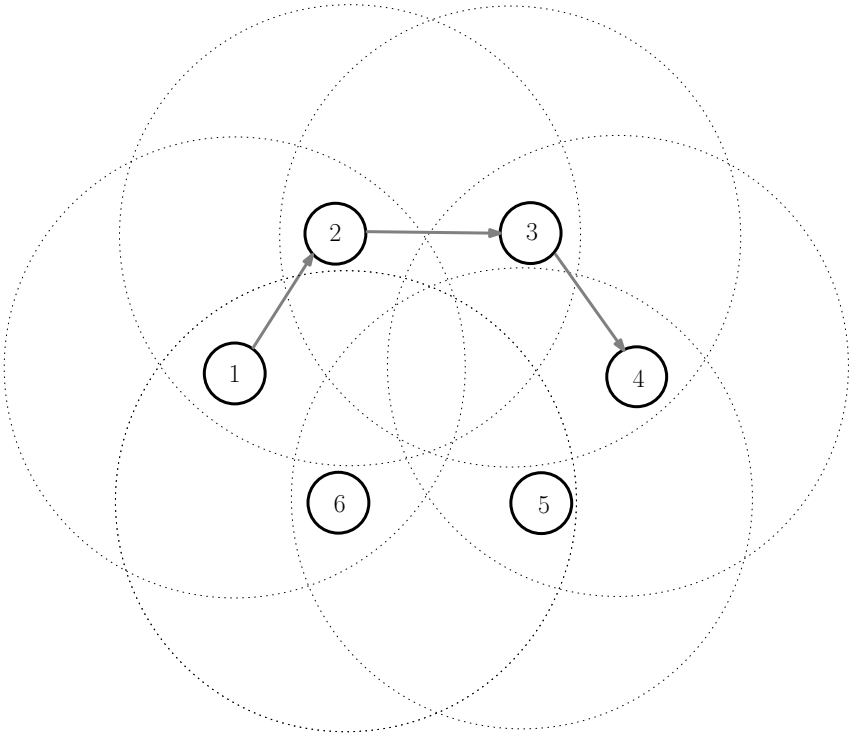


Figura 5.7: *Envío de paquetes de datos sobre el camino de ida finalmente establecido.*

a sus vecinos, que a su vez lo propagan hacia nodos cuyas rutas podrían verse afectadas por esta eventualidad. Puesto que el mensaje RERR se propaga hacia el nodo origen, cada nodo intermedio marca como inválida la ruta cuando recibe el mensaje de error. No obstante, el nodo origen perjudicado puede reiniciar su operación de descubrimiento de rutas en caso de que aún necesite alcanzar ese nodo destino [56, 18].

5.2. Cómo implementar un protocolo de encaminamiento haciendo uso de la maqueta de simulación

En esta sección se va a explicar cómo implementar un protocolo de encaminamiento para poder simularlo con la maqueta de simulación implementada. El protocolo de enrutamiento implementado es una versión más sencilla del protocolo *AODV* descrito en la sección anterior. Aclarar que es una versión más sencilla porque no hará uso de los paquetes de error (no se han simulado roturas de enlaces) ni de las actualizaciones de las rutas en las tablas de encaminamiento por medio de los números de secuencia del nodo destino. Las tablas de encaminamiento tienen un número máximo de rutas y, en caso de intentar añadir una ruta nueva cuando la tabla está

llena, se borra la primera entrada de la misma.

5.2.1. Descripción de las clases a implementar

Para implementar un protocolo de encaminamiento se requiere la creación de una serie de clases. Se utiliza una clase por cada implementación de los mensajes que utiliza dicho protocolo y otra clase para implementar las funciones del protocolo. En el protocolo implementado estas clases están definidas en los ficheros *mensajes_AODV.hpp*, *aodv.cpp* y *aodv.hpp*.

5.2.1.1. Clases que implementan los mensajes del protocolo de encaminamiento

Para implementar los mensajes utilizados por un protocolo bastaría con generar un archivo *.hpp* en el que se implementara una clase por cada mensaje. Estas clases heredan de la clase *Mensaje* y los atributos de cada clase son los campos que forman los mensajes.

En el protocolo *AODV* implementado se ha utilizado el archivo *mensajes_AODV.hpp* para declarar todos los mensajes que utiliza dicho protocolo. Por cada mensaje se ha generado una clase que hereda de la clase *Mensaje* y tiene como atributos los campos de dichos mensajes. En el ejemplo propuesto existen tres clases, para el mensaje RREQ, para el mensaje RREP y para el mensaje RERR.

Los mensajes utilizados en la implementación del protocolo *AODV* difieren un poco respecto a los mensajes definidos en dicho protocolo. Al mensaje RREQ definido por el protocolo *AODV* se le añaden los campos de Identificador del Origen del Siguiente Salto e Identificador del Destino del Siguiente Salto. El mensaje RREP implementado tiene tres campos más que el definido: Identificador del Origen del Siguiente Salto, Número de Secuencia del Origen Inicial e Identificador del Destino del Siguiente Salto. Al mensaje RERR se le añaden todos los campos del mensaje genérico que muestra la Figura 4.6 excepto el campo Tipo.

5.2.1.2. Clase que implementa el funcionamiento del protocolo de encaminamiento

Esta clase deriva de la clase *Nodo* y en ella se implementan los atributos y funciones necesarios para el correcto funcionamiento del protocolo de enrutamiento a ejecutar en la maqueta de simulación. Las dos funciones principales en esta clase son `Mensaje_al_Detectar_Sensor` y `Procesar_Mensaje`. Estas funciones están definidas como virtuales en la clase *Nodo* (como se explicó en la Sección 4.3.12).

En la función `Mensaje_al_Detectar_Sensor` se iniciará el proceso de búsqueda de rutas en caso de que no se conozca una ruta hacia el destino. En caso de conocer la ruta hacia el destino, se envía el mensaje de datos hacia ese destino.

En la función `Procesar_Mensaje` se llevan a cabo todas las acciones definidas en el protocolo de encaminamiento a simular para que la búsqueda de rutas se realice correctamente. En ella, se procesarán todos los mensajes recibidos por cada nodos y se tomarán las decisiones que conlleve dicho procesamiento. Además, también define qué hacer cuando un nodo recibe un mensaje de datos o de reconocimiento positivo.

El protocolo AODV implementado se encuentra escrito en los ficheros *aodv.cpp* y *aodv.hpp*. Este protocolo almacena, en cada nodo de la red, el identificador del siguiente mensaje RREQ a generar y una lista con las rutas descubiertas. Esta lista almacena por cada ruta el identificador del destino de la ruta, el identificador del nodo vecino al cual se debe enviar el paquete para llegar al destino deseado, el número de secuencia del destino (no utilizado en esta implementación), el número de saltos necesarios para alcanzar el destino desde este nodo, el tiempo de vida de la ruta (no utilizado en esta implementación) y el tiempo de última modificación de la ruta (no utilizado en esta implementación).

Cuando un nodo *AODV* detecta la magnitud sensada en su entorno, hará uso de la función `Mensaje_al_Detectar_Sensor` para iniciar la búsqueda de la ruta hacia el destino final o, si conoce dicha ruta (el destino final es un nodo vecino o ya ha descubierto la ruta), enviará el paquete de datos hacia ese destino. Si se inicia el proceso de descubrimiento de ruta, se genera un mensaje RREQ con los campos contador de saltos a cero, identificador de RREQ con el valor del atributo `_id_sig_RREQ`, identificador del destino del siguiente salto a BROADCAST, identificador del origen del siguiente salto con el valor del nodo que ha generado el mensaje, identificador del nodo destino final con el valor del nodo sumidero del nodo origen, identificador del nodo origen inicial con el identificador del nodo que crea el mensaje y número de secuencia del nodo origen inicial con el número de secuencia del nodo que crea el mensaje. Dicho número de secuencia es idéntico tanto para el mensaje de datos generado como para el mensaje RREQ. El número de secuencia en cada nodo se incrementa tras la generación de cualquier mensaje, y el identificador de RREQ se incrementa después de la creación de un mensaje de ese tipo. El mensaje de datos pendiente de enviar se almacena en la lista de mensajes de datos correspondiente, para que cuando se conozca la ruta hacia el destino pueda ser enviado. Además, dicho mensaje se guarda en la lista de retardos junto con el instante de tiempo en el cual se generó.

En caso de conocer la ruta hacia el destino, se envía el mensaje de datos, almacenándose en la lista de retardos dicho mensaje y el instante de tiempo de generación del mismo.

Después de que un nodo *AODV* recibe un mensaje, utiliza la función `Procesar_Mensaje` para procesarlo. La primera comprobación que se hace es saber si el mensaje ya ha sido recibido, y para ello se tiene la lista de mensajes recibidos. Dicha lista almacena los mensajes recibidos durante un periodo de tiempo, tras el cual, se puede volver a procesar el mensaje en caso de volverlo a recibir. Los mensajes se diferencian por el tipo, identificador del nodo destino final, identificador del nodo origen inicial y número de secuencia del origen inicial. Si el mensaje ya ha sido recibido se descarta, en caso contrario, se procesa dependiendo del tipo del mismo.

Si el mensaje es de tipo RREQ y tiene como destino final algún vecino del nodo receptor del mismo o el nodo receptor conoce una ruta hacia ese destino final, se genera el mensaje RREP correspondiente. Los campos del RREP se especifican a continuación. El contador de saltos tendrá un valor uno si el destino final es un nodo vecino, o tendrá un valor igual número de saltos almacenados en la lista de encaminamiento en cualquier otro caso. El identificador del destino del siguiente salto será el campo identificador origen del siguiente salto del mensaje RREQ recibido si el destino final es un nodo vecino de receptor; en caso de que el destino final sea cualquier otro nodo, el identificador del destino del siguiente salto será el especificado en la lista de enrutamiento del nodo receptor del mensaje RREQ. El identificador del origen del siguiente salto será el identificador del nodo que genera el mensaje RREP. El identificador del destino final será el identificador del origen inicial del mensaje RREQ recibido. El identificador del origen inicial será el identificador del destino final del mensaje RREQ recibido y el número de secuencia del origen inicial será el número de secuencia del origen inicial que traía el mensaje RREQ. Los mensajes RREP, RREQ y de datos que se generan con el mismo proceso llevan los mismos números de secuencia. En caso de que no se conozca la ruta hacia el destino final se reenvía el mensaje RREQ recibido por inundación a los vecinos del nodo receptor, aumentando en una unidad el número de saltos. Además, en caso de que el nodo receptor no tenga en su lista de rutas una ruta para el origen inicial del RREQ, o la ruta que tiene es de mayor número de saltos que la especificada en el mensaje RREQ, se añade la ruta especificada en el mensaje RREQ a la lista de rutas del nodo receptor.

En caso de que el mensaje recibido sea RREP y el nodo receptor sea el destino final, se almacena en la lista de rutas del nodo receptor la ruta hacia el origen inicial del mensaje RREP, se busca el mensaje de datos en la lista correspondiente y se envía al destino. Si el mensaje

RREP recibido tiene como destino final un nodo vecino del nodo receptor o un nodo cuya ruta tiene almacenada en su lista de encaminamiento, el nodo receptor reenvía el mensaje RREP aumentando en una unidad el contador de saltos del mismo. Además, en caso de que el nodo receptor no tenga una ruta para el origen inicial del RREP o la ruta que tiene es de mayor número de saltos que la especificada en el mensaje RREQ, se añade la ruta especificada en el mensaje RREP a la lista de rutas del nodo receptor.

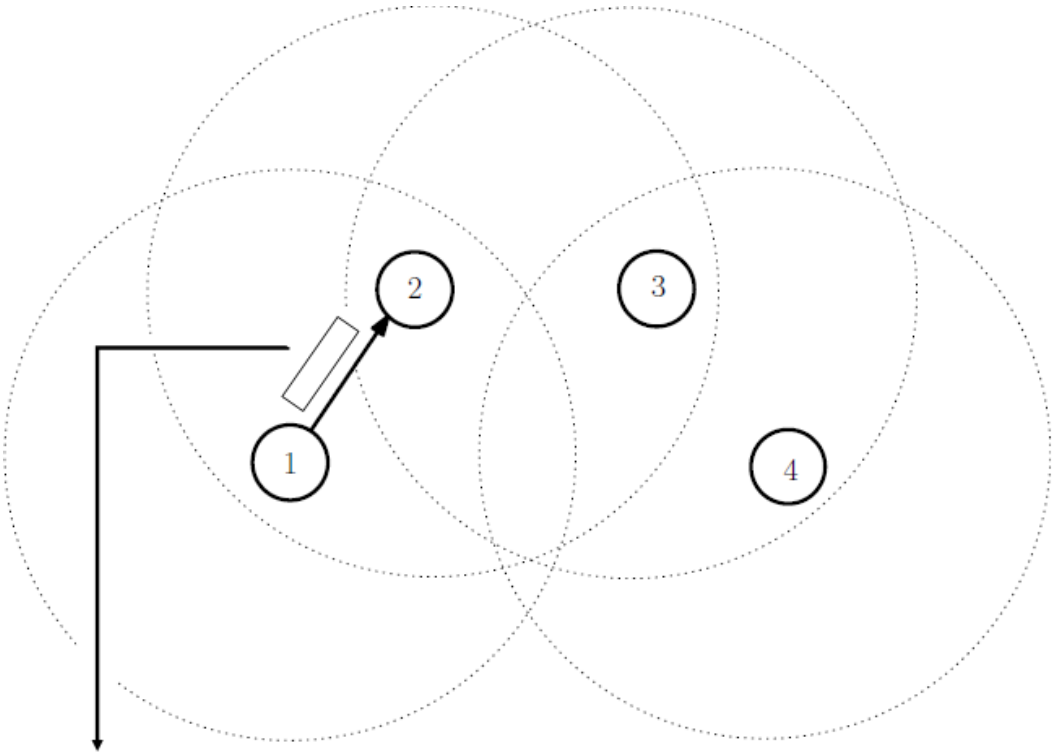
Si el mensaje recibido es de datos y el destino final es el nodo receptor, se crea un mensaje ACK y se envía al nodo origen inicial del mensaje de datos. Además de crear el mensaje ACK, se calcula el retardo del mensaje recibido. Si el mensaje de datos tiene como destino final un nodo vecino o un nodo cuya ruta conoce el nodo receptor, dicho nodo transmite el mensaje al nodo vecino correspondiente para alcanzar el destino final especificado en el mensaje de datos.

El último tipo de mensaje que puede recibir un nodo es ACK. Si el destino final del mensaje ACK es el propio nodo no se hace nada. En caso de que el destino final sea un vecino del nodo receptor o dicho nodo conozca una ruta hacia ese destino final, se envía el mensaje al vecino correspondiente para alcanzar dicho destino final.

En la Figura 5.8, Figura 5.9, Figura 5.10 y Figura 5.11 se muestra un proceso de descubrimiento de ruta iniciado por el nodo 1 para comunicarse con el nodo 4. La Figura 5.8 muestra el inicio del proceso de descubrimiento de ruta mediante el envío por inundación de un paquete RREQ por parte del nodo 1. En la Figura 5.9 se observa como dicho paquete ha llegado al nodo 2, que lo retransmite incrementando el contador de saltos. Cuando el mensaje llega al nodo 3, éste actualiza su lista de rutas y, posteriormente, como se observa en la Figura 5.10, genera un paquete RREP, ya que conoce una ruta para llegar al destino final del paquete RREQ, y lo envía de forma unicast al nodo 2. Cuando el nodo 2 recibe el paquete RREP actualiza su lista de rutas, como muestra la Figura 5.11, incrementa el contador de saltos del mensaje RREP y lo envía de forma unicast al nodo 1. Al recibir el nodo 1 el mensaje RREP, actualiza su lista de rutas y ya está dispuesto a iniciar la transmisión de datos al nodo 4.

5.3. Simulaciones realizadas del protocolo de encaminamiento *AODV*

Se han realizado cinco simulaciones diferentes para ver cómo se comporta el protocolo *AODV* implementado frente a las variaciones de algunos parámetros que influyen en el funcionamiento



0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1							
Tipo = RREQ	J	R	G	D	U	Reservado	CS = 0
RREQ ID = 1							
Identificador del Destino en el Siguiente Salto = BROADCAST							
Identificador del Origen en el Siguiente Salto = 1							
Identificador del Destino Final = 4							
Número de Secuencia del Destino Final = 0							
Identificador del Origen Inicial = 1							
Número de Secuencia del Origen Inicial = 1							

Figura 5.8: *Envío de RREQ del nodo 1 al nodo 2.*

del mismo. Con estas simulaciones se pretende mostrar una serie de métricas para comparar los distintos protocolos de encaminamiento que se pueden implementar en la maqueta de simulación realizada. Estas simulaciones, para un protocolo concreto implementado por el usuario, son costosas de realizar en los simuladores del mercado, debido a que la implementación de un protocolo concreto es difícil de integrar en el entorno de dichos simuladores. Dichos simuladores se basan en utilizar paquetes de código propios con protocolos ya implementados, pero no dan facilidades al usuario para implementar un protocolo diferente de los implementados.

En estas pruebas se ha utilizado la misma red usada para las simulaciones de la maqueta sin protocolo de encaminamiento. Dicha red está representada en la Figura 4.22. Los parámetros

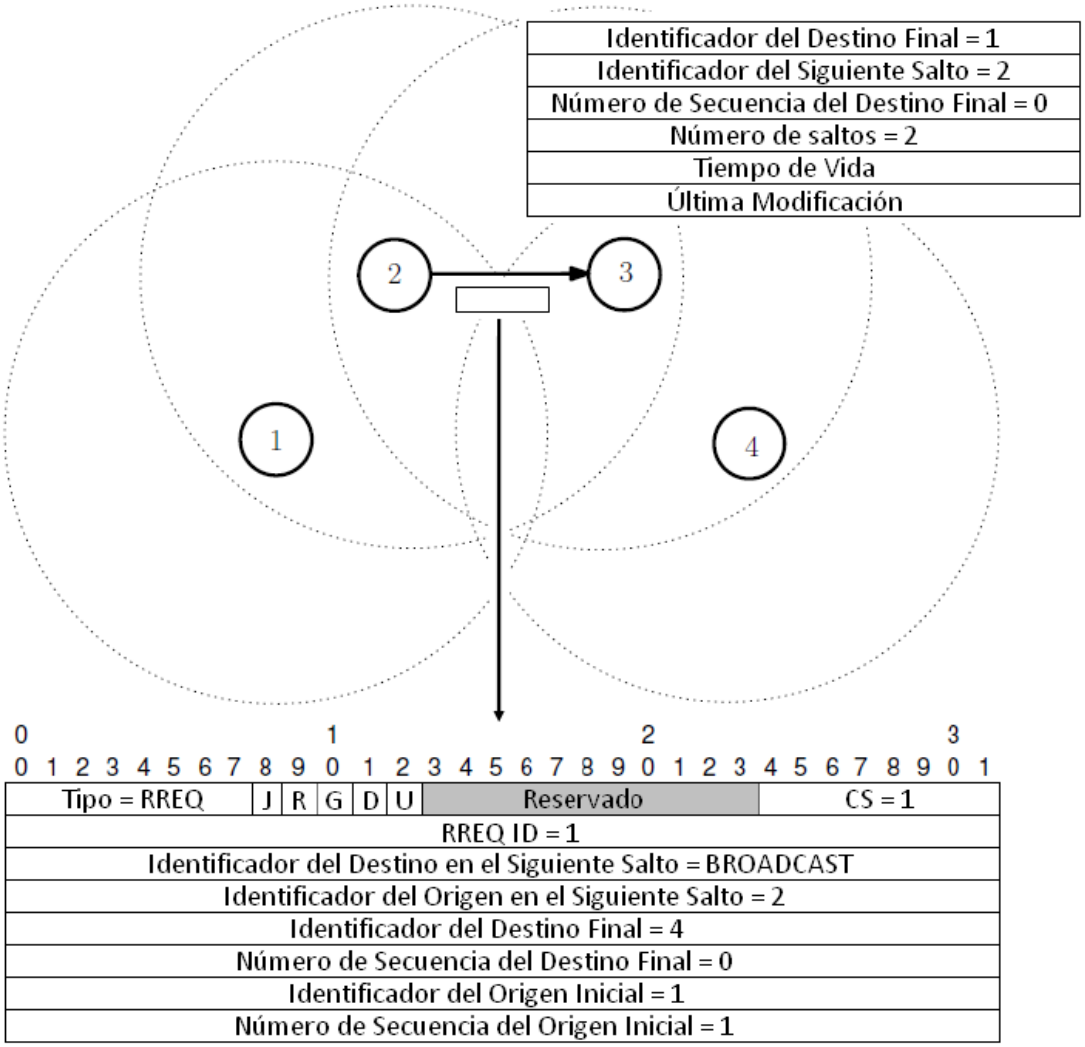


Figura 5.9: *Envío de RREQ del nodo 2 al nodo 3 y actualización de la lista de rutas del nodo 3.*

utilizados son los mismos que los usados en las pruebas de la maqueta de simulación (ver Sección 4.5), con el añadido de que el número de sumideros será uno en todas las simulaciones excepto en las simulaciones en las que varíe dicho parámetro y el número máximo de rutas será dos.

5.3.1. Variación del porcentaje de escucha

La variación del porcentaje de escucha se ha realizado desde el 10 % hasta el 90 %. Esta simulación permite observar qué ocurre cuando un nodo permanece más tiempo escuchando y menos tiempo dormido.

En la Figura 5.12 se observa que al aumentar el porcentaje de escucha, el retardo medio en la red va aumentando. Esto se debe a que al aumentar el porcentaje de escucha se está aumentando

el período de escucha y, como tras dicho período se producen las transmisiones de los mensajes recibidos, conforme el período de escucha aumente tardarán más en enviarse dichos mensajes.

La Figura 5.13 muestra cómo al aumentar el porcentaje de escucha va aumentando de forma lineal el *throughput*. Este fenómeno es idéntico al ocurrido cuando se ha simulado la maqueta. Se debe a que al aumentar el período de escucha se podrá recibir un mayor número de mensajes.

La batería media restante en la red disminuye de forma lineal cuando se aumenta el porcentaje de escucha como se puede ver en la Figura 5.14. Este fenómeno también ha ocurrido al simular la maqueta y se debe a que al aumentar el porcentaje de escucha, el nodo pasa más tiempo escuchando el medio y menos tiempo dormido. El consumo de batería es mayor al aumentar el tiempo activo y disminuir el tiempo de dormido.

En la Figura 5.15 se muestra cómo afecta el aumento del porcentaje de escucha al instante temporal en el cual muere el primer nodo de la red. Se puede observar cómo al aumentar el porcentaje de escucha, el instante temporal de muerte del primer nodo en la red disminuye de forma exponencial. El tiempo hasta el agotamiento de la batería del primer nodo en la red muestra la robustez de dicha red. Cuando el nodo tiene un porcentaje de escucha del 10 % el tiempo de muerte del primer nodo es de 25 días aproximadamente, si se pasa al 90 % de porcentaje de escucha, el tiempo de muerte del primer nodo pasa a 6 días aproximadamente.

5.3.2. Variación del período de nodo

La variación del período de nodo también se ha realizado para ver qué ocurre cuando hay un protocolo de encaminamiento funcionando en la red. Las conclusiones de la variación del período con respecto al *throughput* y la batería media restante en la red son las mismas que se obtuvieron en la maqueta de simulación.

La Figura 5.16 muestra que al aumentar el periodo de nodo, el retardo medio en la red aumenta de forma lineal. Al aumentar el período de nodo se está aumentando el tiempo de escucha y, por la misma razón que en la sección anterior, aumenta el retardo medio de la red.

En la Figura 5.17 se puede observar cómo permanece constante el *throughput* al aumentar el período de nodo. Por la misma razón dada que cuando se ha simulado la maqueta, esto ocurre porque al aumentar el periodo de nodo aumentan de manera proporcional el tiempo de dormido, de transmisiones y de escucha, y por tanto, la relación entre ellos no varía. Al no variar la relación entre los tiempos que forman el período de nodo, el *throughput* será el mismo.

Cuando aumenta el período de nodo, la batería media restante en la red va aumentando muy lentamente como muestra la Figura 5.18. Para esta simulación se ha utilizado un período de dormido mayor que el de escucha más transmisiones, cosa que ocurre normalmente en las redes de sensores inalámbricos. Al aumentar el período, aumentan el tiempo de escucha, de transmisiones y de dormido. Para un mismo tiempo de finalización de la simulación, el nodo pasará mayor tiempo dormido a medida que se aumente el período, debido a que el porcentaje de ciclo dormido es mayor que el de escucha más transmisiones.

5.3.3. Variación de la probabilidad de ocurrencia de sucesos a sensar

Como se comentó en la Sección 4.5.3, la probabilidad de ocurrencia de sucesos sensados tiene una relación lineal con el tráfico entrante. Al aumentar dicha probabilidad, aumenta también el tráfico entrante. En esta sección se va a variar el tráfico entrante para ver cómo afecta al *throughput*, al retardo medio en la red y a la batería media restante.

Conforme aumenta el tráfico entrante, el retardo medio va disminuyendo hasta un punto (punto óptimo), a partir del cual va aumentando. Este fenómeno ocurre porque a medida que aumenta el tráfico entrante, va aumentando el número de mensajes que se transmiten en la red. Conforme aumenta el número de transmisiones, un mayor número de mensajes serán recibidos por los nodos pero también habrá un mayor número de colisiones. Hay que llegar a un equilibrio entre el aumento del número de transmisiones y el aumento del número de colisiones. El punto de equilibrio se da para un tráfico entrante normalizado de 0.4 como muestra la Figura 5.19. A partir de este punto de equilibrio, las colisiones dan lugar a un aumento progresivo del retardo.

La Figura 5.20 muestra cómo al aumentar el tráfico entrante aumenta el *throughput* hasta un punto, a partir del cual comienza a bajar. El aumento del *throughput* se debe a que al aumentar el tráfico entrante, aumenta el número de mensajes transmitidos y por tanto se reciben un mayor número de mensajes habiendo pocas colisiones entre ellos. El *throughput* comienza a bajar cuando el número de trasmisiones es tan grande que da lugar a un gran número de colisiones.

La batería media restante en la red va disminuyendo muy lentamente conforme aumenta el tráfico entrante como se puede apreciar en la Figura 5.21. Este fenómeno ocurre debido a que al aumentar el tráfico entrante habrá un mayor número de transmisiones y un mayor consumo de batería.

5.3.4. Variación del número de sumideros en la red

Otra métrica utilizada para la simulación de protocolos de encaminamiento es la variación del número de nodos sumideros en la red. En la red utilizada en la simulación esta variación parte de un nodo sumidero y llega hasta cuatro nodos.

Conforme aumenta el número de sumideros en la red, el retardo medio va disminuyendo. Esto es debido a que al aumentar el número de sumideros, los nodos tendrán su nodo sumidero en un lugar más cercano a ellos (menor número de saltos) y por tanto el retardo será menor. Este fenómeno se puede observar en la Figura 5.22.

En la Figura 5.23 se puede apreciar cómo al aumentar el número de nodos sumideros, el *throughput* aumenta. Esto se debe a que al ir aumentando el número de sumideros, los nodos tendrán más cerca a su nodo sumidero y los paquetes de datos tendrán que recorrer menos saltos para llegar al destino. Bajo este razonamiento, los paquetes de datos, en general, llegarán de forma más fácil al destino, y al aumentar el número de paquetes recibidos, el *throughput* aumenta.

El número de nodos sumideros no afecta al rendimiento de la batería de los nodos como se puede apreciar en la Figura 5.24. Esto es debido a que la batería está relacionada con el período del nodo y dicho período no sufre ninguna variación cuando se modifica el número de nodos sumideros en la red.

5.3.5. Variación de la batería media restante en la red frente al tiempo

En esta sección se va a representar cómo varía la batería media restante en la red a lo largo del tiempo. A medida que se va avanzando en el tiempo, la batería media va disminuyendo de forma lineal con el mismo. El primer nodo muere en 575990 segundos (6.6 días) cuando se utiliza un con un porcentaje de escucha del 40 %.

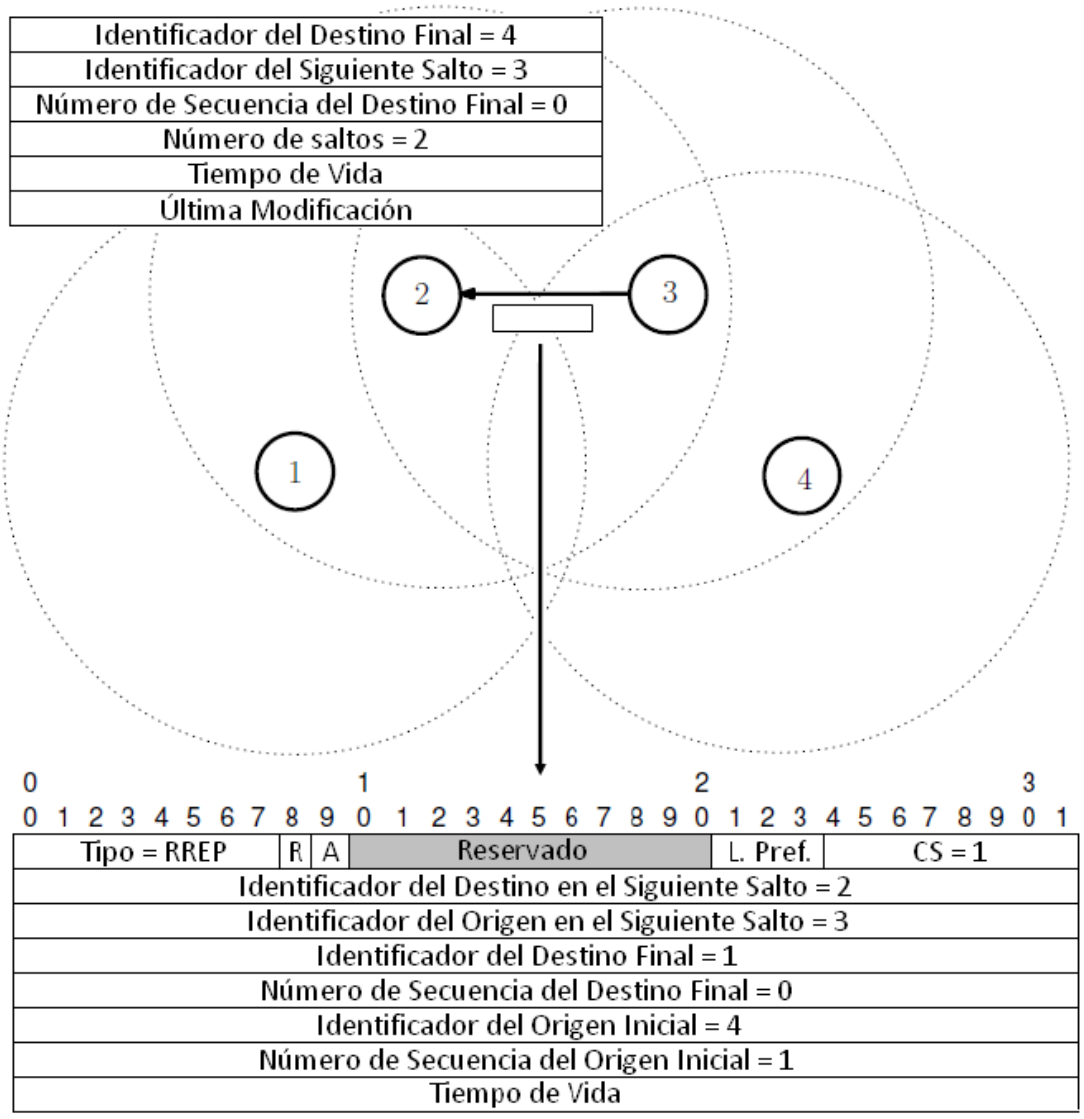


Figura 5.10: *Envío de RREP del nodo 3 al nodo 2 y actualización de la lista de rutas del nodo 2.*

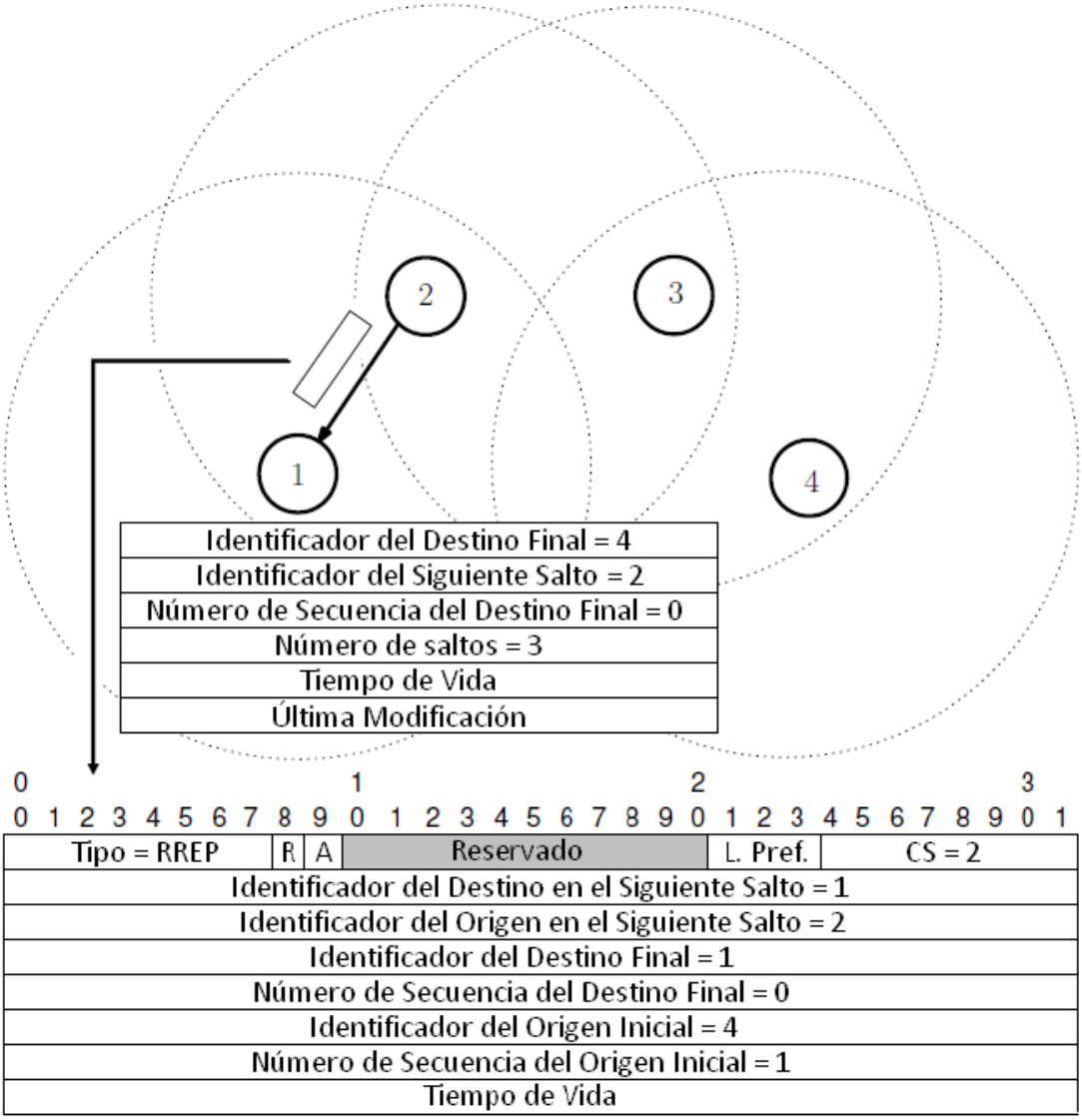


Figura 5.11: *Envío de RREP del nodo 2 al nodo 1.*

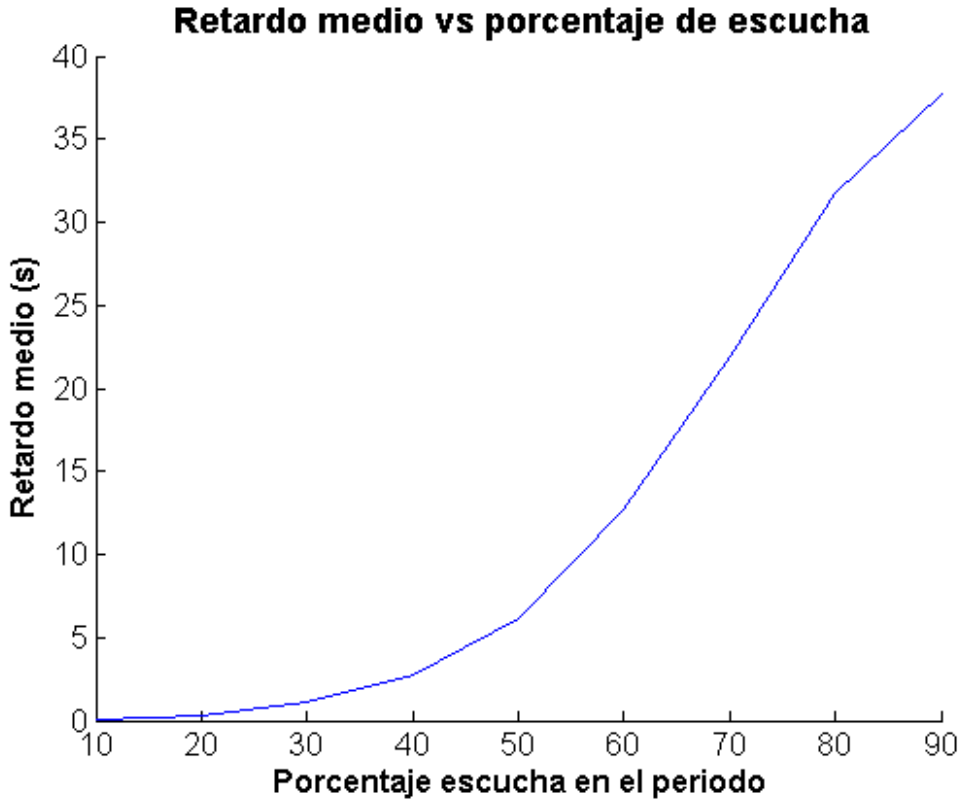


Figura 5.12: *Retardo medio en la red vs porcentaje de escucha.*

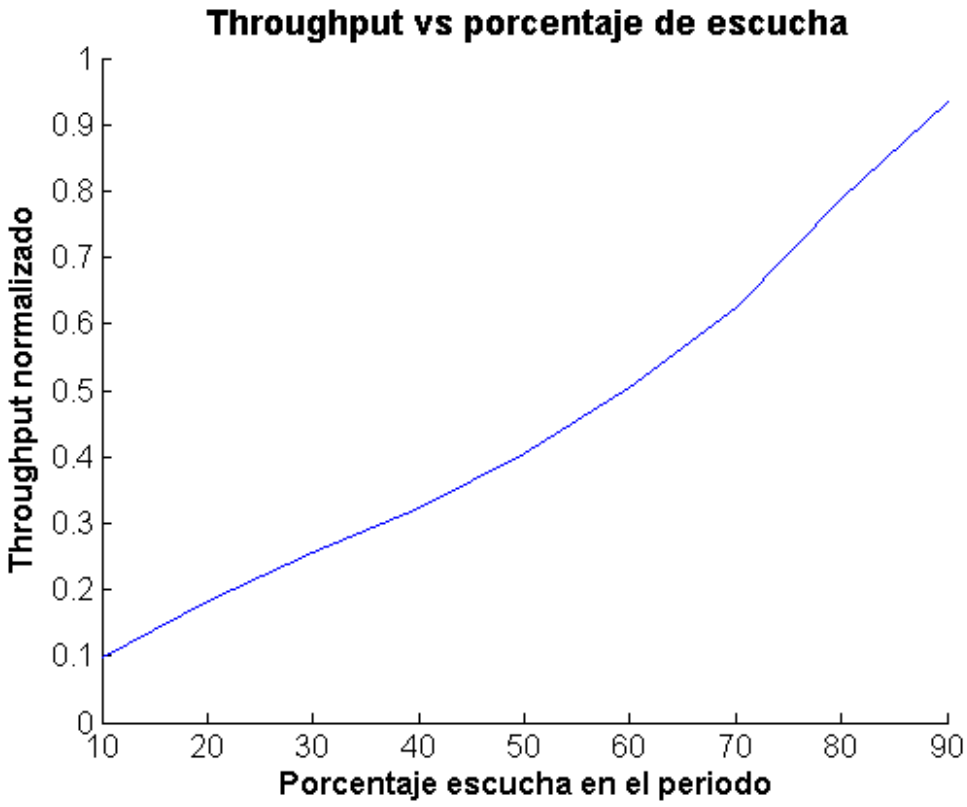


Figura 5.13: *Throughput vs porcentaje de escucha.*

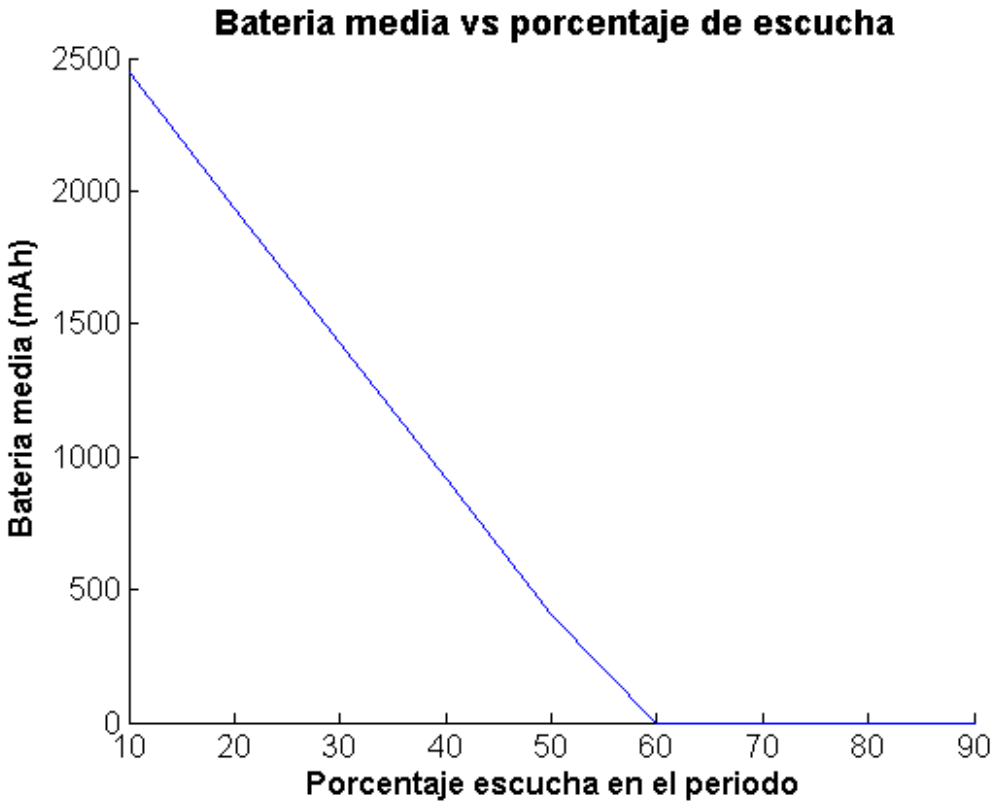


Figura 5.14: *Batería media restante en la red vs porcentaje de escucha.*

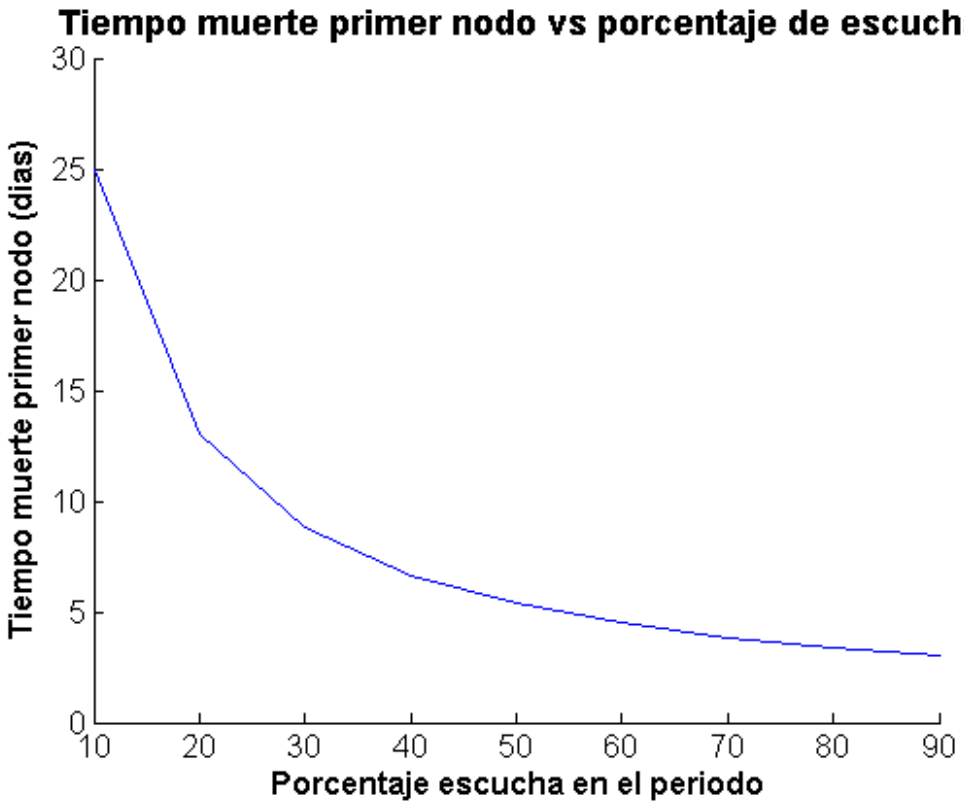


Figura 5.15: *Tiempo muerte primer nodo vs porcentaje de escucha.*

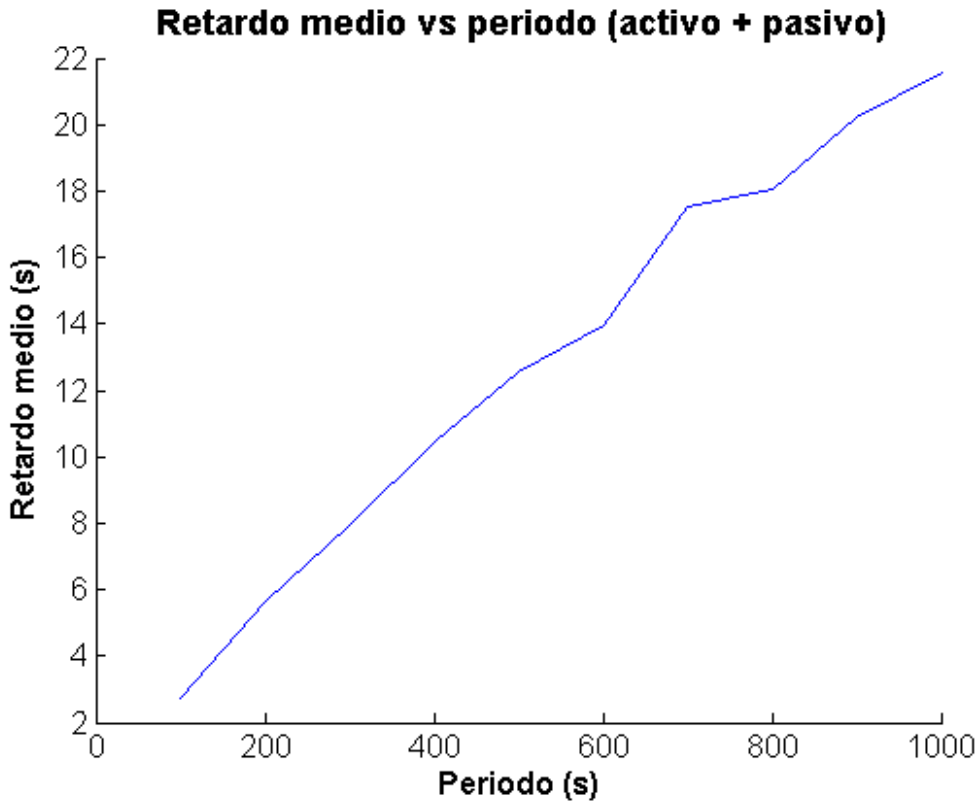


Figura 5.16: *Retardo medio en la red vs período de nodo.*

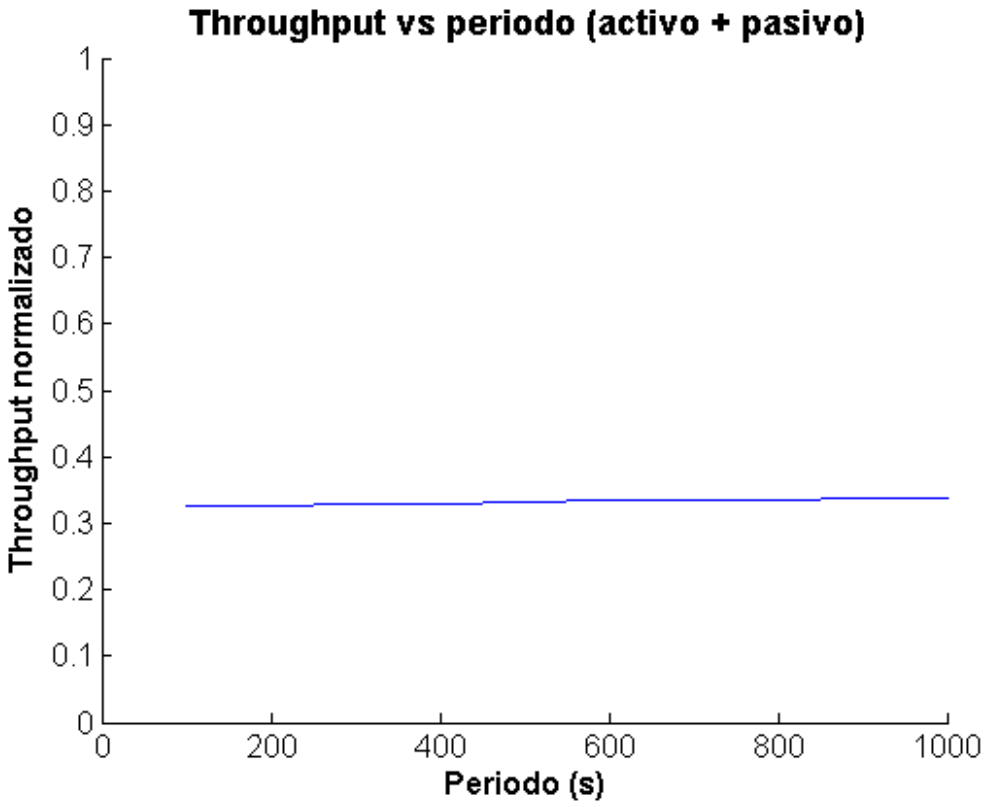


Figura 5.17: *Throughput vs período de nodo.*

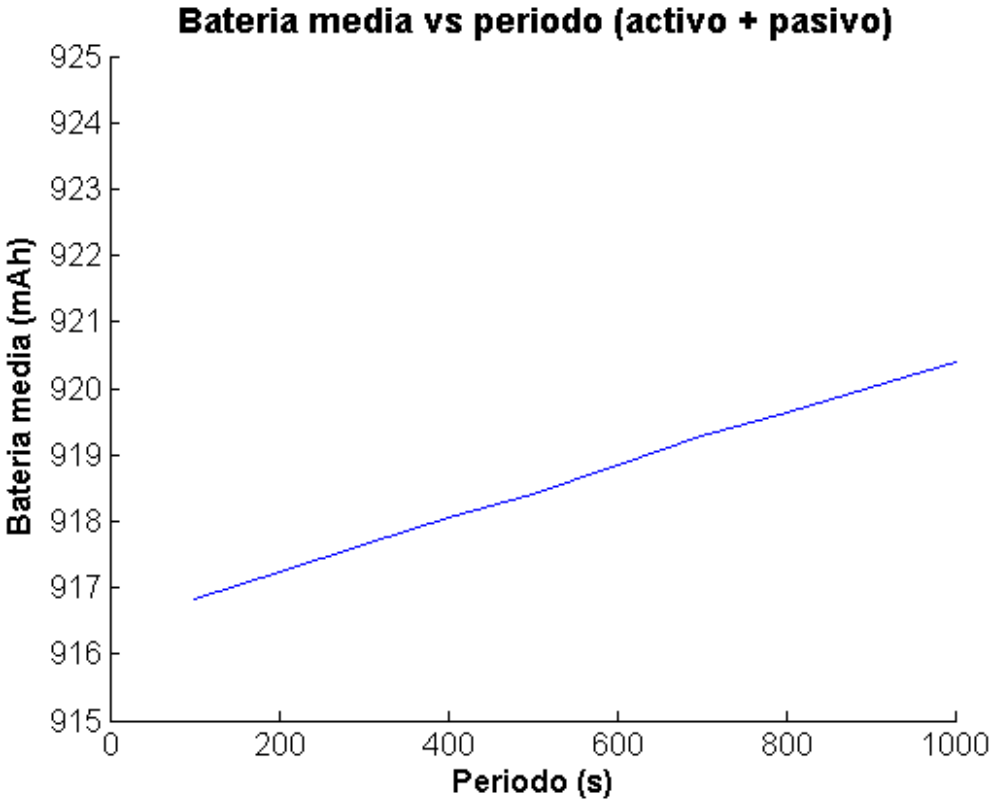


Figura 5.18: *Batería media restante en la red vs período de nodo.*

Figura 5.19: *Retardo medio en la red vs tráfico entrante.*

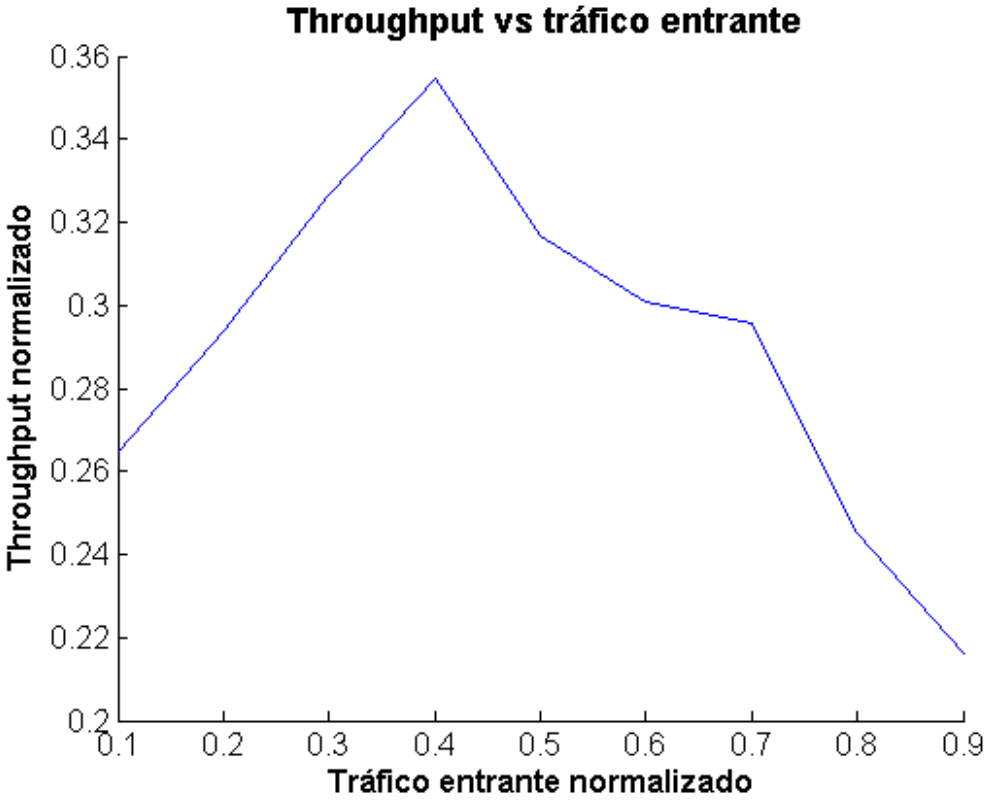


Figura 5.20: *Throughput vs tráfico entrante.*

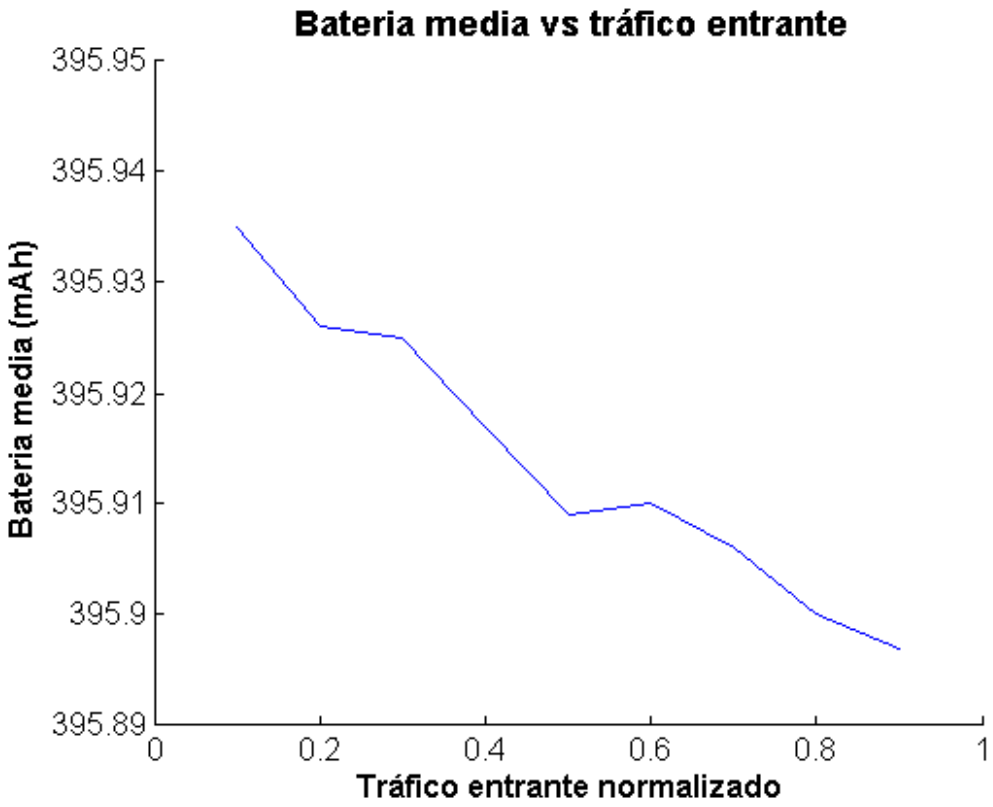


Figura 5.21: *Batería media restante en la red vs tráfico entrante.*

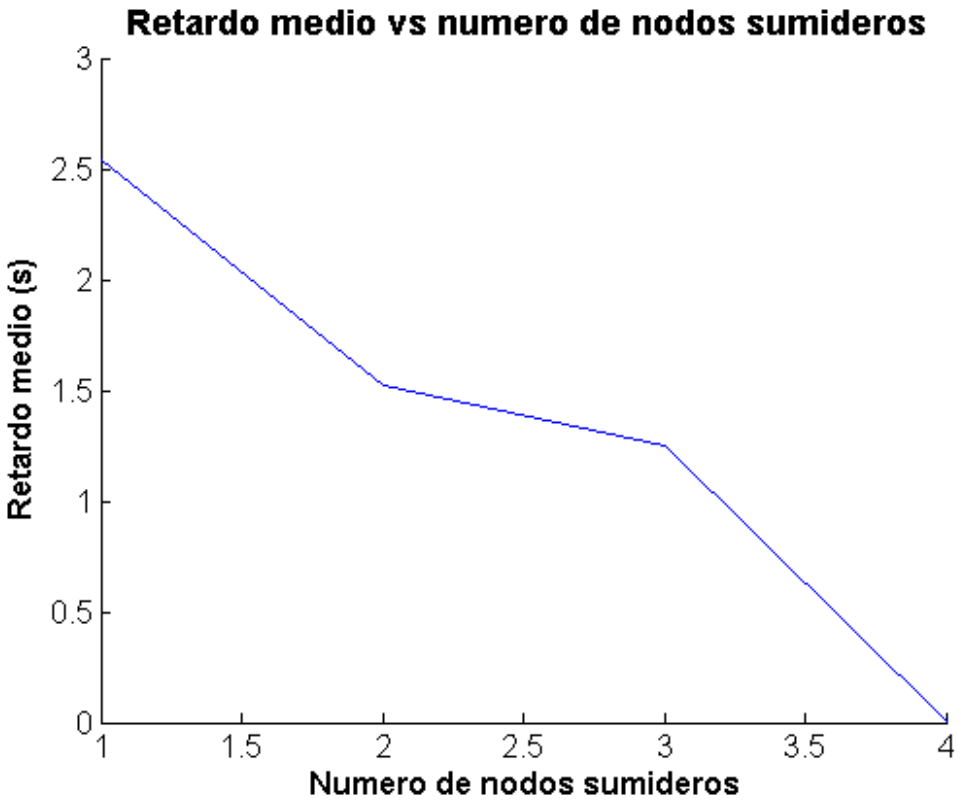


Figura 5.22: *Retardo medio en la red vs número de sumideros.*

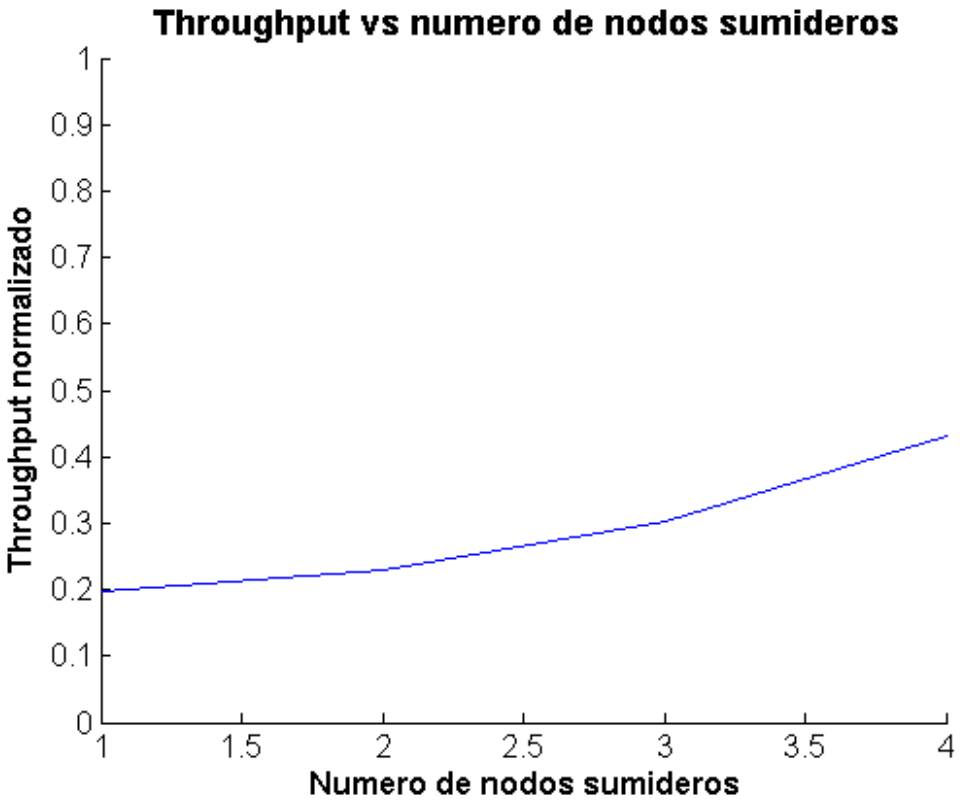


Figura 5.23: *Throughput vs número de sumideros.*

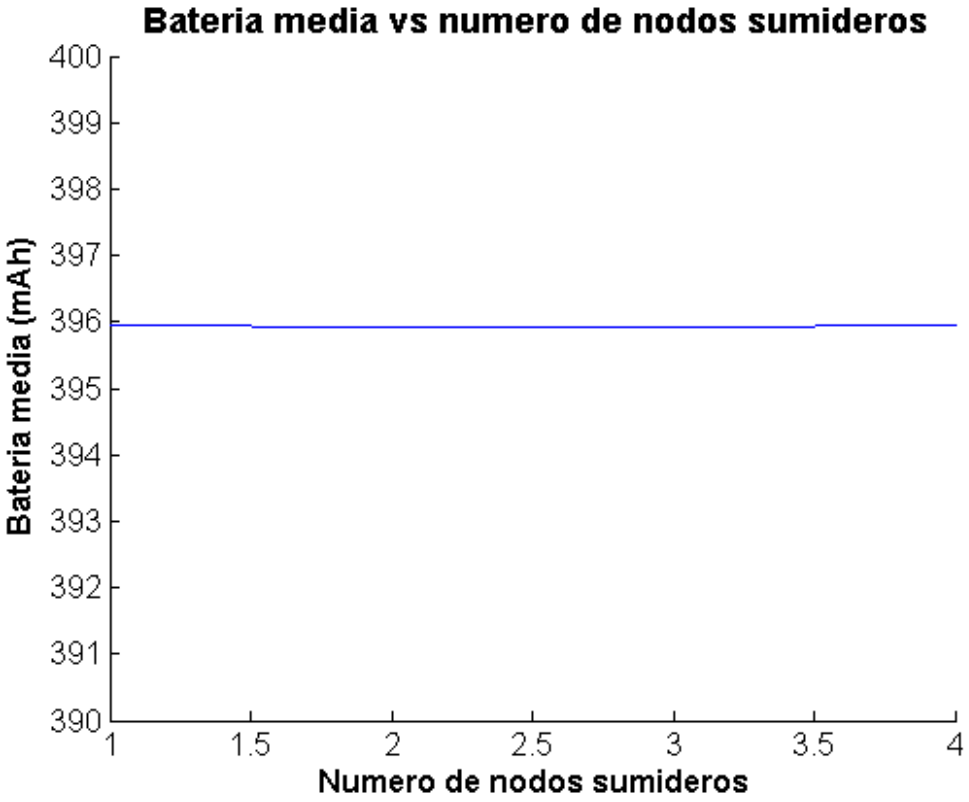


Figura 5.24: *Batería media restante en la red vs número de sumideros.*

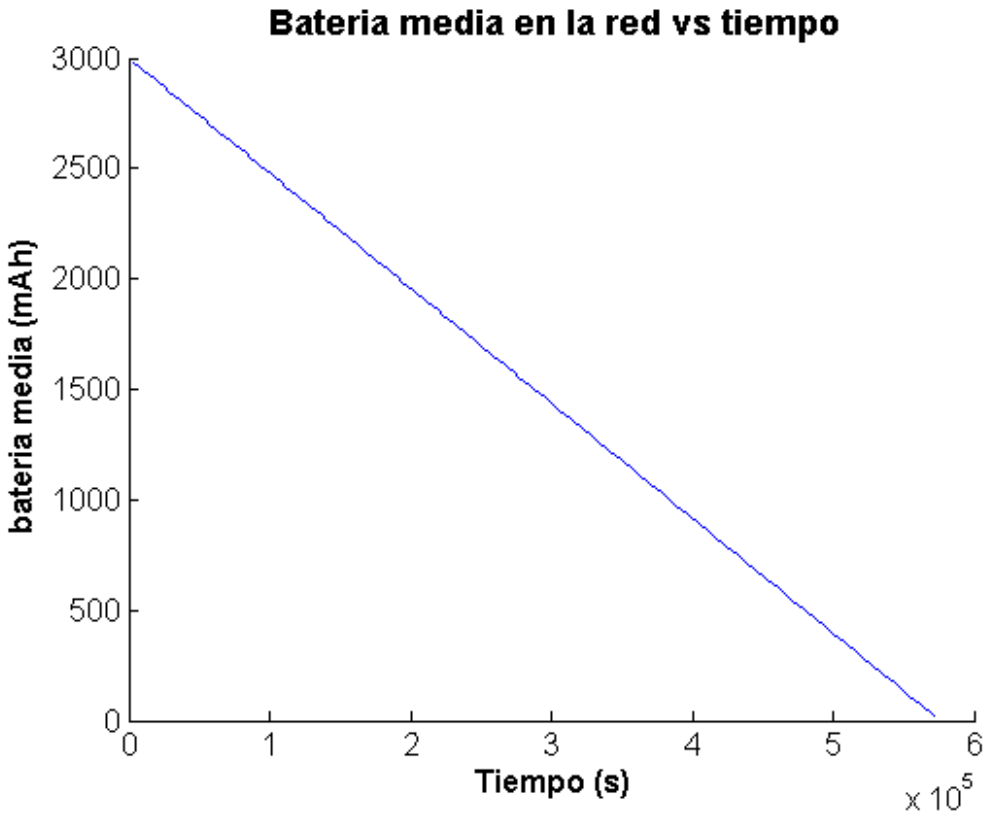


Figura 5.25: *Variación de la bateria media restante en la red frente al tiempo.*

Capítulo 6

CONCLUSIONES Y LÍNEAS FUTURAS

6.1. Conclusiones

Debido al gran número de aplicaciones de las redes de sensores inalámbricas y a los numerosos proyectos de investigación centrados en estas redes, es importante caracterizar el funcionamiento de las mismas.

En este proyecto se ha realizado un estudio del estado del arte de las redes de sensores inalámbricas y de los algoritmos de encaminamiento en estas redes. Basándose en el estudio, en dicho proyecto se implementa una maqueta de simulación para poder caracterizar el comportamiento de cualquier algoritmo de encaminamiento que pueda usarse en redes de sensores inalámbricos.

La implementación de la maqueta de simulación tiene su justificación en el pequeño número de simuladores existentes en el mercado capaces de ofrecer una buena caracterización de los protocolos de encaminamiento en las redes de sensores inalámbricos. Estos simuladores reúnen complejidad y, dificultad de uso y desarrollo de los módulos que los forman. La ventaja de estos simuladores está en el uso de protocolos ya implementados dentro de los mismos, esta ventaja se convierte en desventaja cuando se quiere implementar un protocolo distinto a los implementados.

El entorno de simulación propuesto está pensado para ofrecer una buena aproximación del funcionamiento de los protocolos de encaminamiento en redes de sensores ah-hoc, en las cuales todos los nodos tienen el mismo papel excepto los nodos sumideros, que se encargan de procesar la información sensada. Además, el entorno está creado para que se ejecute de manera eficiente ofreciendo la posibilidad de una configuración rápida mediante ficheros.

Cabe citar las numerosos dificultades que han surgido a la hora de simular las acciones que

llevan a cabo los nodos. Se ha optado por un entorno de simulación mediante eventos, los cuales se van generando a medida que va avanzando el tiempo de la simulación. Una parte clave está en el procesamiento y la generación de dichos eventos. Se han comprobado varias formas de generarlos y procesarlos, eligiendo la actual por el tema de eficiencia de memoria y de tiempo de ejecución.

Para comprobar el funcionamiento de la maqueta de simulación se ofrecen varias pruebas realizadas con los parámetros de configuración más importantes. La red utilizada para las simulaciones es una red sencilla formada por seis nodos, en la que cada uno de los nodos tiene dos vecinos. En dichas figuras se puede comprobar la importancia de los tiempos de dormido, transmisión y escucha en los nodos sensores y del consumo de batería en los mismos.

Como ejemplo de un protocolo de encaminamiento se ha implementado el protocolo AODV que utiliza el estándar Zigbee. Este protocolo ha usado las facilidades que ofrece el entorno de simulación para la implementación. Solamente con la creación de tres ficheros, uno en el que se definen los mensajes utilizados en el protocolo y otros dos en los cuales se define el funcionamiento del mismo, se ha implementado el protocolo de encaminamiento citado anteriormente.

Como punto de finalización del proyecto, se han mostrado varias métricas para poder comparar protocolos de encaminamiento implementados en el entorno de simulación propuesto por este proyecto. Las métricas utilizadas son la tasa efectiva de trasporte de información, la batería media restante en la red, el tiempo de muerte del primer nodo y el retardo medio en la red. Como en la mayoría de los casos, existe un compromiso entre conseguir un mayor *throughput* en la red y disminuir el consumo de batería en los nodos.

6.2. Líneas futuras

Como trabajo futuro se propone implementar, en la maqueta de simulación, varios protocolos de encaminamiento propuestos en el estudio del estado del arte realizado en la Sección 3. En dichos protocolos se podría comparar su funcionamiento en el simulador con su funcionamiento en la vida real, para poder ajustar los parámetros de la maqueta y poder obtener resultados más que aceptables. De esta forma se puede tener un modelo que se ajusta bastante a la realidad.

En el entorno de simulación se puede modificar el modelo de propagación dependiendo del emplazamiento donde ubicar los sensores. Se podría hacer un estudio de varios emplazamientos distintos (en campo abierto, en la ciudad, en el pueblo, ...) para ver cómo se comportan los

sensores en ellos. Teniendo varios modelos de propagación, la herramienta estaría mucho más completa.

El método de acceso al medio utilizado en este proyecto ha sido por contienda. Debido a éste método de acceso, para poder obtener resultados aceptables, los nodos deben consumir mucha energía debido al uso de grandes períodos de escucha. Para mejorar notablemente los resultados, se podría emplear sincronismo en el acceso al medio, aunque ésto dé lugar a un aumento del procesamiento en los nodos.

Una mejora que ya está implementada, pero que no se ha utilizado debido a que con el método de acceso se conseguían malos resultados, es el uso de retransmisiones. Las retransmisiones permiten mejorar la fiabilidad de la red debido al reenvío de los mensajes que se pierden o no se escuchan. Estos mensajes podrán ser recibidos correctamente por los nodos que anteriormente no pudieron hacerlo.

Por último cabe destacar que la implementación del protocolo de encaminamiento AODV no ha utilizado todas las ventajas que dicho protocolo ofrece (números de secuencia del destino y actualización de rutas). Se podría comparar el protocolo implementado con una nueva implementación del protocolo con todas sus funciones y ver los beneficios de la nueva implementación.

APÉNDICES

APÉNDICE A

PRESUPUESTO DEL PROYECTO

En este apéndice se presentan justificados los costes globales de la realización de este Proyecto Fin de Carrera. Tales costes, imputables a gastos de personal y de material, se pueden deducir de las Tablas A.1 y A.2.

En la Tabla A.1 se muestran las fases del proyecto y el tiempo aproximado para cada una de ellas. Así pues, se desprende que el tiempo total dedicado por el proyectando ha sido de 1.190 horas, de las cuales aproximadamente un 30 % han sido compartidas con el tutor del proyecto, por lo que el total asciende a 1.547 horas. Teniendo en cuenta que la tabla de honorarios del Colegio Oficial de Ingenieros de Telecomunicación establece unas tarifas de 60 €/hora, el coste de personal se sitúa en 92.820 €.

En la Tabla A.2 se recogen los costes de material desglosados en equipo informático, local de trabajo, documentación y gastos varios no atribuibles (material fungible, llamadas telefónicas, desplazamientos...). Ascienden, pues, a un total de 3.640 €.

A partir de estos datos, el presupuesto total es el mostrado en la Tabla A.3.

Tabla A.1: *Fases del Proyecto*

Fase 1	*Documentación*	350 *horas*
Fase 2	*Desarrollo del software*	200 *horas*
Fase 3	*Análisis de los datos*	390 *horas*
Fase 4	*Redacción de la memoria del proyecto*	250 *horas*

Tabla A.2: *Costes de material*

Ordenador de gama media	1.300 €
Local (durante 12 meses, con un coste de 120 €/mes)	1.440 €
Documentación	200 €
Gastos varios	700 €

Tabla A.3: *Presupuesto*

Concepto	**Importe**
Costes personal	78.000 €
Costes material	3.640 €
Base imponible	96.460 €
I.V.A. (16 %)	15.433,6 €
TOTAL	111.893,6 €

Bibliografía

[1] I. 802.15.4-2006. *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)*, chapter 15.4. 2006.

[2] I. 802.15.4a 2007. *Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, chapter 15.4. 2007.

[3] N. Aakvaag and J.-E. Frey. Redes de sensores inalámbricos. *Revista ABB*, (2):39–42, 2006.

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *IEEE Communications Magazine*, 40(8):102–114, 2002.

[5] J. N. Al-Karaki. Data aggregation in wireless sensor networks - exact and approximate algorithms. *IEEE Wksp. High Perf. Switching and Routing*, Abril 2004.

[6] J. N. Al-Karaki and A. E. Kamal. Routing techniques in wireless sensor network: a survey. *Wireless Communications, IEEE*, 1:6–28, Diciembre 2004.

[7] C. Alcaraz. Kms crisis guidelines web application. Website, 2008. http://www.isac.uma.es/CRISIS/tools.html.

[8] C. Alcaraz, R. Román, and J. López. Análisis de la aplicabilidad de las redes de sensores para la protección de infraestructuras de información críticas. In *VII Jornadas de Ingeniería Telemática, del 16 al 18 de Septiembre de 2008*, Septiembre 2008.

[9] S. Bandyopadhyay and E. Coyle. An energy efficient hierarchical clustering algorithm for wireless sensor networks. *INFOCOM*, 2003.

[10] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. *1st Wksp. Sensor Networks and Apps.*, Octubre 2002.

[11] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. *Int´l. Conf. Distrib. Comp. Sys.*, Noviembre 2003.

[12] N. Bulusu. Scalable coordination for wireless sensor networks: Self-configuring localization systems. *6th Int´l. Symp. Commun. Theory and Apps.*, Julio 2001.

[13] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost out door localization for very small devices. Technical report, Comp. Sci. Dept., 2000.

[14] S. B. Calpe. Optimización de un protocolo de encaminamiento en redes de sensores ieee 802.15.4, Julio 2006.

[15] S. Capkun, M. Hamdi, and J. Hubaux. Gps-free positioning in mobile ad-hoc networks. *34th Annual Hawaii Int´l. Conf. Sys. Sci.*, 2001.

[16] P. C.E. *Ad Hoc Networking.* Addison-Wesley, 2001.

[17] P. C.E. and B. P. Highly dynamic destination-sequenced distance-vector routign (dsdv) for mobile computers. *Comp. Commun*, 1994.

[18] I. D. Chakeres and E. M. Belding-Royer. Aodv routing protocol implementation design. *In AODV Routing Protocol Implementation Design*, Marzo 2004.

[19] J. Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *Adv. Telecommun. and Info. Distrib. Research Prog.*, Marzo 2000.

[20] B. Chen. Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Wireless Networks*, Mayo 2002.

[21] K. Choi and J.-I. Song. Investigation of feasible cryptographic algorithms for wireless sensor network. Technical report, Proceedings of the 8th International Conference on Advanced Communication Technology (ICACT 2006), 2006.

[22] M. Chu, H. Haussecker, and F. Zhao. Scalable information driven sensor querying and routing for ad hoc heterogeneous sensor networks. *J. High Perf Comp. Apps.*, Agosto 2002.

[23] C. con clase. Curso de c++. capítulo 36: Herencia. Website, 2001. http://www.conclase.net/c/curso/index.php?cap=036.

[24] S. Corson and J. Macker. *Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations*, chapter RFC 2501. 1999.

[25] D. E. Culler and J. Hui. 6lowpan tutorial. Technical report, University of California, Berkeley, 2007.

[26] J. D.B. and M. D.A. Dynamic source routing in ad-hoc wireless networks. *Mobile Computing*, 1996.

[27] M. B. Delgado, J. M. G. Pardo, A. M. Sala, E. E. López, F. D. Jiménez, and L. J. Llacer. Caracterización de la propagación a 868mhz en una red estática de sensores. *Departamento de Tecnologías de la Información y las Comunicaciones, Universidad Politécnica de Cartagena*, 2005.

[28] L. Doherty, W. Lindsay, and J. Simon. Channel-specific wireless sensor network path data. *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, pages 89–94, 2007.

[29] S. Dulman. Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. *WCNC Wksp.*, Marzo 2003.

[30] M. O. Echeverria. Caracterización del enrutamiento en redes inalámbricas de geosensores. Technical report, Universidad Distrital Francisco José de Caldas, 2007.

[31] P. Esposito. Future networks. Website, 2008. http://www.sensorsmag.com/sensors/Feature+ARTICLEs/Future-Networks/ARTICLEStandard/ARTICLE/detail/494983.

[32] D. Estrin. Next century challenges: scalable coordination in sensor networks. In *5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, 1999.

[33] Q. Fang, F. Zhao, and L. Guibas. Lightweight sensing and communication protocols for target enumeration and aggregation. *4th ACM MOBIHOC*, 2003.

[34] J.-E. Frey, A. Kreitz, and G. Scheible. Desenchufado pero conectado, parte 1: Redefinición de lo inalámbrico. *Revista ABB*, (3), 2005.

[35] C. Gómez, P. Salvatella, and J. Paradells. Diseño y evaluación en un entorno real de un protocolo de routing para redes de sensores mesh ieee 802.15.4. Technical report, Universidad Politécnica de Cataluña, 2006.

[36] D. Goodman. *Wireless Personal Communications Systems.* Addison-Wesley, 1997.

[37] M. Hatler, C. Chi, and Ph.D. *Wireless Sensor Networks: Growing Markets, Accelerating Demand.* On World, 2005.

[38] T. He. Speed: A stateless protocol for real-time communication in sensor networks. *Int´l. Conf. Distrib. Comp. Sys.*, Mayo 2003.

[39] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *Sys. Sci*, Enero 2000.

[40] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. *ACM/IEEE Mobicom*, Agosto 1999.

[41] E. C. R. Ibarra. *Protocolo de enrutamiento para redes de sensores y actuadores inalámbricos.* PhD thesis, Centro de Investigación Científica y de Educación Superior de Ensenada, BC, Agosto 2006.

[42] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. *ACM Mobi-Com*, 2000.

[43] S. Jayasheree. A battery-aware mac protocol for ad hoc wireless networks. Technical report, Department of Computer Science and Engineering, Indian Institute of Technology, Madras, India, Octubre 2003.

[44] D. B. Johnson, D. A. Maltz, and J. Broch. *The dynamic source routing protocol for multihop wireless ad hoc networks*, chapter 5. Addison-Wesley, 2001.

[45] H. Karl and A. Willig. A short survey of wireless sensor networks. Technical report, Technical University Berlin, Octubre 2003.

[46] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. *Ad Hoc Networks*, 2003.

[47] F. Kuhn, R. Wattenhofer, and A. Zollinger. Worst-case optimal and average-case efficient geometric ad hoc routing. *4th ACM Int´l. Conf. Mobile Comp. and Net.*, 2003.

[48] J. Kulik, W. R. Heinzelman, and H. Balakrishnan. Negotiation-based protocols for disseminating information in wireless sensor networks. *Wireless Networks*, 2002.

[49] N. Kushalnagar and G. Montenegro. 6lowpan: Overview, assumptions, problem statement and goals. *IETF Internet Draft (Work in progress)*, Octubre 2005.

[50] Y. Law, J. Doumen, and P. Hartel. Survey and benchmark of block ciphers for wireless sensor networks. *ACM Transactions on Sensor Networks*, 2(1):65–93, 2006.

[51] L. Li and J. Y. Halpern. Minimum-energy mobile wireless networks revisited. *IEEE ICC*, 2001.

[52] Q. Li, J. Aslam, and D. Rus. Hierarchical power-aware routing in sensor networks. *DIMACS Wksp. Pervasive Net.*, Mayo 2001.

[53] J. Lindqvist and J. Lindqvist. Counting to infinity, 2004.

[54] S. Lindsey and C. Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. *IEEE Aerospace Conf.*, 2002.

[55] A. Liu and P. Ning. A configurable library for elliptic curve cryptography in wireless sensor networks. Technical report, North Carolina State University, Department of Computer Science, 2007.

[56] C. Liu and J. Kaiser. A survey of mobile ad hoc network routing protocols. Technical report, University of Magdeburg, Octubre 2005.

[57] J. Liu and S. Singh. Wireless sensor networks characteristics. *IEEE JOURNAL on Selected Areas in Communications*, 19(7):1300–1315, 2001.

[58] F. Loureiro, S. Nogueira, and B.Linnyer. Redes de sensores sem fio. Technical report, Congresso da SBC, Julio 2002.

[59] E. F. López and O. G. Bermejo. Redes de sensores autoconfigurables, Julio 2006.

[60] J. L. López and S. M. Doce. Desarrollo de un demostrador para evaluar técnicas cross-layer en sistemas de comunicaciones inalámbricos, Marzo 2008.

[61] A. Manjeshwar and D. P. Agarwal. Apteen: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. *Int´l. Parallel and Distrib. Proc. Symp.*

[62] A. Manjeshwar and D. P. Agarwal. Teen: a routing protocol for enhanced efficiency in wireless sensor networks. *1st Int´l. Wksp. on Parallel and Distrib. Comp. Issues in Wireless Networks and Mobile Comp.*, Abril 2001.

[63] D. G. Martos. Redes de sensores. Technical report, Universidad de Sevilla, Mayo 2008.

[64] S. Middleton. Wireless personal area network low rate project authorization request. *IEEE P802.15-00/248r4*, 2000.

[65] R. Morris. Exceeding the standard for wireless sensor networks. Technical report, EE Times-Asia, Mayo 2008.

[66] A. M. Márquez, J. J. P. Solano, and J. P. Sebastiá. Red de sensores inalámbricos para monitorización de terrenos mediante tecnología ieee 802.15.4. *XX Simposio Nacional de la URSI*, Septiembre 2005.

[67] F. M. A. Papacetzi. *Wireless Personal Area Network (WPAN) & Home Networking. Cap. 4: El estándar IEEE 802.15.4.* PhD thesis, Universidad de las Américas Puebla, Diciembre 2003.

[68] Z. W. Paper. Zigbee and wireless radio frequency coexistence. Technical report, ZigBee Alliance, Junio 2007.

[69] F. J. G. Peñalvo. Herencia en *C++. Departamento de Informática y Automática, Universidad de Salamanca*, 2002.

[70] W. Pereira. Estudio comparativo de tecnologías de redes inalámbricas autoconfigurables para enrutamiento dinámico. Technical report, Universidad Andrés Bello, 2007.

[71] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ip flooding in ad hoc mobile networks. Technical report, IETF Internet Draft, 2001.

[72] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on-demand distance vector (aodv) routing. RFC Experimental 3561, Internet Engineering Task Force, July 2003.

[73] C. E. Perkins and E. M. Royer. Adhoc ondemand distance vector routing. *In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[74] A. Perrig. Spins: Security protocols for sensor networks. *Wireless Networks*, 2000.

[75] J. R. Polastre. *A Unifying Link Abstraction for Wireless Sensor Networks*. PhD thesis, University of California, Berkeley, 2005.

[76] C. Rahul and J. Rabaey. Energy aware routing for low energy ad hoc sensor networks. *IEEE WCNC*, Marzo 2002.

[77] D. Reche Martínez, A. García Linares, and J. Richarte Reina. Redes ubicuas un nuevo paradigma en sanidad. Technical report, Dpto.I+D+I. Novasoft Sanidad, Abril 2003.

[78] V. Rodoplu and T. H. Meng. Minimum energy mobile wireless networks. *IEEE JSAC*, Agosto 1999.

[79] R. Roman, J. Lopez, and S. Gritzalis. Situation awareness mechanisms for wireless sensor networks. *IEEE Communications Magazine*, 46(4):102–107, 2008.

[80] M. S. and G.-L.-A. J.J. An efficient routing protocol for wireless networks. *ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks*, 1996.

[81] N. Sadagopan. The acquire mechanism for efficient querying in sensor networks. *Wksp. Sensor Network Protocol and Apps*, Mayo 2003.

[82] J. S. Sanchis. Redes de sensores inalámbricas. Technical report, Universidad de Valencia, 2007.

[83] A. Savvides, C.-C. Han, and M. Srivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. *7th ACM MobiCom*, 2001.

[84] C. Schurgers and M. Srivastava. Energy efficient routing in wireless sensor networks. *MILCOM Proc. Commun. for Network-Centric Ops.: Creating the Info*, 2001.

[85] S. Servetto and G. Barrenechea. Constrained random walks on random graphs: Routing algorithms for large scale wireless sensor networks. *Wksp. Wireless Sensor Networks and Apps.*, 2002.

[86] R. C. Shah and J. Rabaey. Energy aware routing for low energy ad hoc sensor networks. *IEEE WCNC*, Marzo 2002.

[87] K. Sohrabi and J. Pottie. Protocols for self-organization of a wireless sensor network. *EEE Pers. Commun.*, 2000.

[88] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, M. Nixon, and W. Pratt. Wirelesshart: Applying wireless technology in real-time industrial process control. *IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 377–386, 2008.

[89] I. Stojmenovic and X. Lin. Gedir: Loop-free location based routing in wireless networks. *Conf. Parallel and Distrib. Comp. and Sys.*, Noviembre 1999.

[90] L. Subramanian and R. H. Katz. An architecture for building self configurable systems. *IEEE/ACM Wksp. Mobile Ad Hoc Net. and Comp.*, Agosto 2000.

[91] S. J. Talabac. Sensor webs, an emerging concept for future earth observing systems. Technical report, UNASA/GSFC, 2003.

[92] N. I. o. S. U.S. Department of Commerce and I. T. L. Technology. Specification for the advanced encryption standard (aes). *Federal Information Processing Standard Publication (FIPS PUB) 197*, Noviembre 2001.

[93] J. A. G. Velasco and L. V. C. Ormaza. Redes de sensores. Technical report, Escuela Politécnica Nacional Quito-Ecuador, Marzo 2006.

[94] J. Walters, W. S. Z. Liang, and V. Chaudhary. Wireless sensor network security: A survey, security in distributed, grid, and pervasive computing. Technical report, CRC Press, 2006.

[95] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad-hoc routing. *7th Annual ACM/IEEE Int´l. Conf. Mobile Comp.*, 2001.

[96] Y. Yao and J. Gehrke. The cougar approach to innetwork query processing in sensor networks. *SIGMOD Record*, Septiembre 2002.

[97] F. Ye. A scalable solution to minimum cost forwarding in large sensor networks. *Comp. Commun. and Networks*, 2001.

[98] F. Ye. A two-tier data dissemination model for large-scale wireless sensor networks. *ACM/IEEE MOBICOM*, 2002.

[99] Y. Yu, D. Estrin, and R. Govindan. Geographical and energy-aware routing: A recursive data dissemination protocol for wireless sensor networks. Technical report, UCLA Comp. Sci. Dept., Mayo 2001.

[100] J. L. T. Zurita. *Tutorial de redes de sensores ad hoc con eficiencia en energía*, chapter Capítulo 2, Redes Inalámbricas de Sensores Ad Hoc. Universidad de las Américas Puebla, Mayo 2006.