ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Proyecto Fin de Carrera

Implementación de una aplicación para reconfiguración remota de dispositivos mediante servicios web

AUTOR: Darío Griñán Martínez

DIRECTOR: Dr. D. Fernando Losilla López

Julio/ 2014

Autor	Darío Griñán Martínez
E-mail del Autor	**dariogrimar@hotmail.com**
Director	Dr. D. Fernando Losilla López
E-mail del Director	fernando.losilla@upct.es
Codirector(es)	
Título del PFC	Implementación de una aplicación para configuración remota de dispositivos mediante servicios web.
Descriptores	

Resumen

En este proyecto se pretende diseñar una aplicación y un entorno que permitan la programación de dispositivos limitados en recursos, que usan el protocolo CoAP. El objetivo de la aplicación es traducir un código expresado en un lenguaje textual con acciones sencillas (encender, apagar, umbral superado, etc.) a un archivo binario de tamaño reducido y fácil procesamiento que será enviado a los nodos que se quieren programar. El programa usará, para cargar estos archivos en los dispositivos, clientes CoAP y HTTP (con el correspondiente proxy para traducir entre HTTP y CoAP) La configuración necesaria para encaminar los datagramas IPv6 hacia los dispositivos finales, por ejemplo una red de sensores, e introducirlos en la red destino, se ha obtenido analizando el tráfico de la red cuando los dispositivos se están comunicando entre ellos.

Titulación	I.T.T. Especialidad Telemática
Intensificación	
Departamento	Tecnologías de la Información y las Comunicaciones
Fecha de Presentación	Julio 2014

"Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo total es una victoria completa."

Mahatma Gandhi

AGRADECIMIENTOS

En primer lugar, quiero agradecer a mi familia por todo el apoyo que me han dado para hacer este proyecto.

Gracias también a mis compañeros, que siempre han estado ahí, ayudando y animando. Y por los momentos que hemos vivido juntos. Especialmente a Jorge, compañero de fatigas, con el que he compartido largos días de proyecto, al ser complementarios.

Agradecer también a mi director de proyecto, Fernando, por el esfuerzo en que saliese todo adelante, y su dedicación.

INDICE GENERAL

INDICE DE FIGURAS

1- INTRODUCCIÓN

1.1 Resumen y Objetivos

Las redes de sensores inalámbricas se han vuelto muy populares durante los últimos años. Están formadas por sensores autónomos que están distribuidos para monitorizar las condiciones físicas y ambientales y transferir de forma colaborativa sus datos a través de una red hasta un punto central. Éstas facilitan la creación y la implantación de redes de bajo coste en lugares donde emplear cableado es difícil y costoso. Algunos ámbitos en los que se podrían aplicar estas redes serían la domótica, la logística o la salud.

En este proyecto se va a crear una interfaz gráfica que permita programar el comportamiento de los nodos de una red de sensores. Para ello, se dará la opción de definir en los nodos ciertos umbrales que pueden ser sobrepasados. En este caso, el nodo elegido, o el propio nodo, hará saltar alarmas y realizarán una determinada acción, también definida en la aplicación.

Para poder llevar a cabo esta operación, la aplicación generará un fichero con la información necesaria para enviar al nodo elegido, que se almacenará en el directorio raíz de dicha aplicación. Este fichero tendrá un tamaño reducido para poder ser enviado y almacenado en los nodos.

A la hora de cargar el fichero en los nodos, se ha utilizado el servicio web REST, con las operaciones GET y PUT. Debido a que las operaciones de REST antes nombradas son propias del protocolo HTTP, se ha usado un proxy que se encargue de traducir las peticiones y respuestas entre HTTP y CoAP, protocolo con el que cuentan los nodos sensores.

El GUI desarrollado se va a emplear en aplicaciones domóticas. El concepto de domótica se puede definir como el conjunto de sistemas capaces de automatizar una vivienda, que pueden estar integrados por medio de redes interiores y exteriores de comunicación, cableadas o inalámbricas. Se podría definir como la integración de la tecnología en el diseño inteligente de un recinto cerrado.

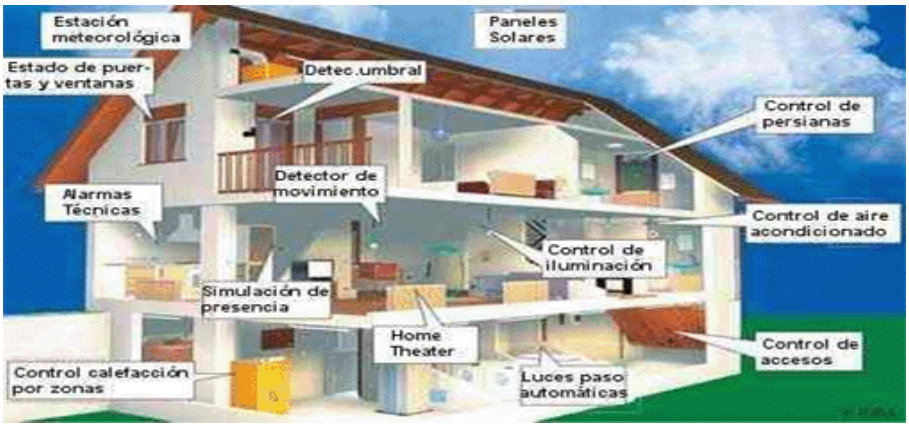

Figura 1.1: Vivienda automatizada

Así, a modo de ejemplo, se podrá configurar que un nodo, al sobrepasar el umbral de temperatura con el que se ha configurado, haga que dicho nodo u otro, encienda uno de sus leds.

En este proyecto, la interfaz gráfica se utilizará para programar una red de sensores que simule un despliegue real en un edificio y, gracias a sus leds, se pueda comprobar la ejecución de acciones.

2- REDES DE SENSORES

2.1 Introducción

Las redes de sensores inalámbricas o “Wireless Sensor Networks” (WSN) son redes formadas por dispositivos independientes, conectados de manera inalámbrica, llamados nodos, que en la mayoría de los casos son capaces de monitorizar el valor de una o varias magnitudes físicas, transmitir esta información a través de una red formada por estos mismos dispositivos, y realizar una tarea común.

La ausencia total de cableado permite que las redes de sensores sean capaces de reorganizarse y continuar funcionando, en el caso de que alguno de los nodos falle o sea eliminado, por lo que se dice que tienen gran escalabilidad y autoconfiguración. Además, el ser inalámbricas permite el ahorro de costes en infraestructuras y la ubicación en lugares de difícil acceso.

Las redes de sensores son capaces de formar una red ad-hoc sin infraestructura física prestablecida ni administración central. La expresión ad-hoc hace referencia a una red en la que no existe un nodo central, sino que todos los dispositivos están en igualdad de condiciones.

Una ventaja de las redes de sensores es la reutilización de frecuencias, esto es, dos nodos con áreas de coberturas disjuntas pueden emplear la misma banda de transmisión. Si la red de sensores contiene una cantidad de nodos lo suficientemente grande, este mecanismo permite establecer múltiples rutas para cada destino posible, contemplando rutas redundantes y empleando rutas alternativas para balancear el consumo entre nodos.

Para interconectar la red de sensores con la red de datos, y así poder monitorizar los valores captados por la red, y recopilar la información de los nodos, es necesario que uno de los motes esté conectado al PC. Éste será el gateway, y estará conectado mediante USB, RS-232 (para comunicación con el PC), Bluetooth, WLAN, Ethernet o incluso Wi-Fi (para conexión a un dispositivoremoto).

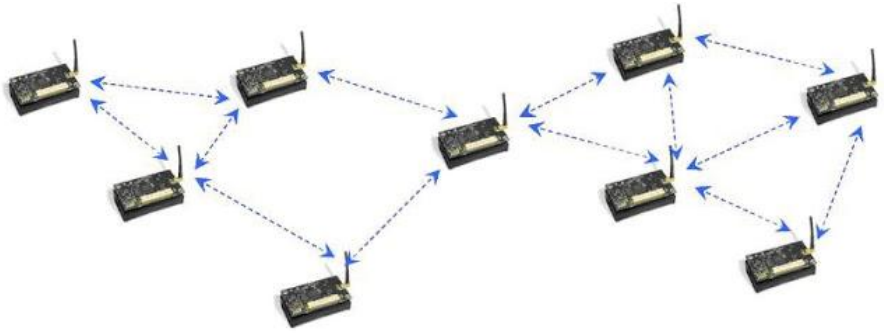

Figura 2.1: Ejemplo de red de sensores

2.2 Características de una red inalámbrica de sensores

Las redes de sensores pueden utilizar diversos nodos, con diferentes características, según la tarea que desempeñen y la aplicación que se les dé, pero hay ciertas características comunes. Algunas de esas características son las siguientes:

- Gran número de elementos. En una red de sensores, se puede usar una gran cantidad de nodos, diseminados de forma aleatoria por el área donde van a realizar su tarea. Pudiendo autoconfigurarse y crear las rutas para cada destino.

- Limitación de recursos. Además de su limitación de tamaño, los sensores dependen de baterías y de la energía del entorno para poder funcionar. Por ello, los nodos están limitados en memoria, potencia de cálculo y ancho de banda.

- Topología dinámica. El despliegue de una red de sensores, y el posicionamiento de sus nodos no es fija, puede ser arbitraria, e incluso, pueden existir movimientos y desaparición y adición de nodos. Al ser capaces de autoconfigurarse, las WSN permiten esta característica.

- Diversidad de aplicaciones. Al tener diferentes características, las redes de sensores tienen un alto nivel de aplicación en diferentes ámbitos. Profundizaremos más en esta característica en el siguiente apartado.

- Bajo coste.

2.3 Aplicaciones de las redes de sensores inalámbricas

En sus inicios las funcionalidades que ofrecían las redes de sensores eran muy básicas pero se apreciaban las múltiples posibilidades que ofrecían. Actualmente han evolucionado tanto que el rango de aplicaciones de estas redes está limitado por la porpia imaginación.

Las redes de sensores se pueden aplicar a numerosos ámbitos, y están en continuo crecimiento. Algunas aplicaciones que se pueden dar a estas redes son:

1. **Aplicaciones de seguridad y militares.** La necesidad del control del campo de batalla, así como el conocimiento del terreno, para facilitar la comunicación y la toma de decisiones. Algunas de las características de las redes de sensores hacen que esta opción sea atractiva en el ámbito militar. Pero no sólo el ámbito militar se aprovecha de éstas características. En situaciones que requieran alta seguridad, como prevención de ataques terroristas, también es necesario conocer el terreno y saber a qué enfrentarse. Gracias a que ciertos tipos de motes disponen de detector de presencia, las redes de sensores también son útiles en este ámbito.

Figura 2.2: Sistema militar de detección de posición con sensores.

2. **Aplicaciones en agricultura y medio ambiente.** La necesidad de llevar un control de la temperatura, humedad y cantidad de agua, entre otros, de los cultivos, hace que las redes de sensores en la agricultura, estén en continuo crecimiento y tengan una gran repercusión en este ámbito. Además, la continua observación del entorno, ya sea en tiempo real o no, es especialmente útil para la detección de inundaciones o controlar la explotación de animales en su hábitat natural.

Figura 2.3: Red de sensores aplicada a la agricultura.

3. **Aplicaciones en medicina.** Las redes de sensores ayudan en este ámbito a la hora de poder monitorizar, en tiempo real, diferentes parámetros vitales de los pacientes, que facilitan la deteccion de anomalías y la rápida intervención. También es útil en la supervisión de pacientes y ancianos en su casa, para monitorizar sus hábitos y llevar un control diario de su actividad.

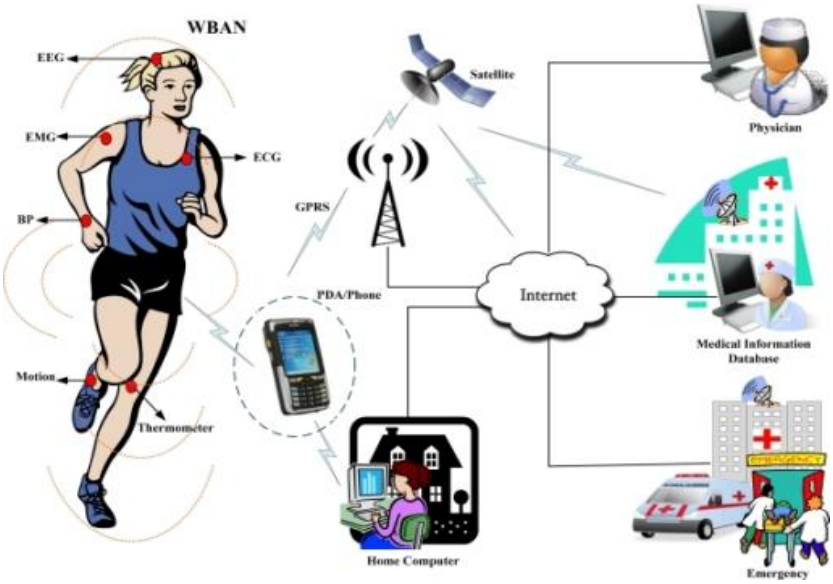

Figura 2.4: Red basada en sensores implantados en el cuerpo (WBAN).

4. **Aplicación en redes de vehículos.** En el sector automovilístico, cada vez más se está avanzando en la tecnología, y es necesario llevar un cierto control sobre todo lo que rodea el funcionamiento de un vehículo, como la gestión del motor, el confort o la seguridad. Pero no sólo se quiere tener un control exaustivo del propio vehículo, sino que el entorno también ha de ser monitorizado para una mayor facilidad y seguridad en la conducción. En la actualidad es posible la comunicación entre vehículos, para calcular distancias, detectar accidentes u obtener cualquier tipo de información como la climatología. Al estar en continuo movimiento, la red cambia de topología constantemente, por lo que es necesario que los nodos estén en adaptación continua.

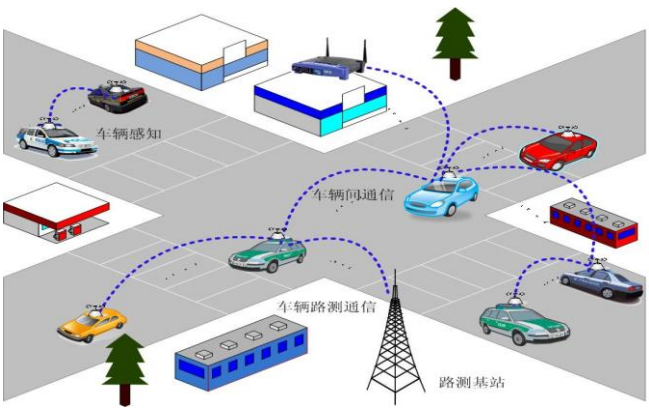

Figura 2.5: Ejemplo de red de sensores vehicular.

5. **Aplicación contra incendios forestales.** Relacionado con el cuidado y control del medio ambiente está la prevención de incendios forestales. Mediante la monitorización de los parámetros ambientales de una zona, como la temperatura, la humedad, nivel de precipitaciones o velocidad del viento, se pueden prevenir incendios, detectar su foco en una fase temprana del mismo o averiguar las causas de su origen.

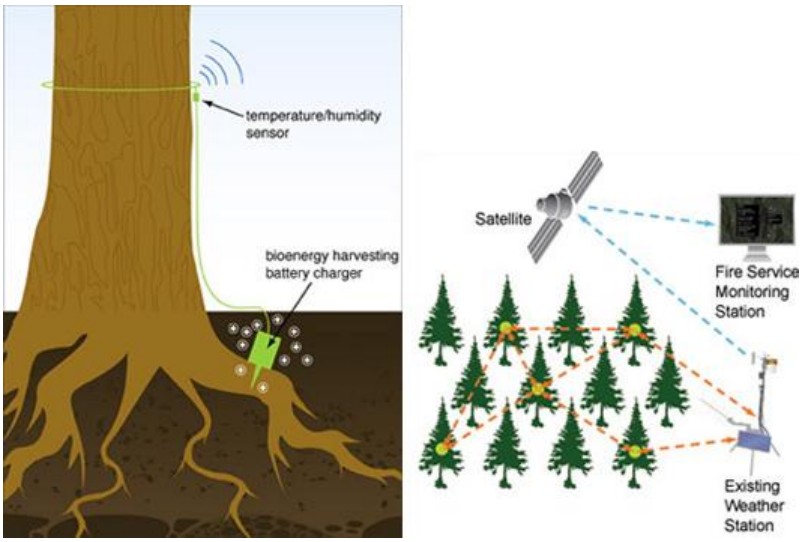

Figura 2.6: Red para la prevención de incendios forestales.

6. Aplicaciones de tracking. Control de paquetes que contienen nodos, para localizar su posición en todo momento. La topología tiene que ser muy dinámica debido al continuo movimiento de los nodos.

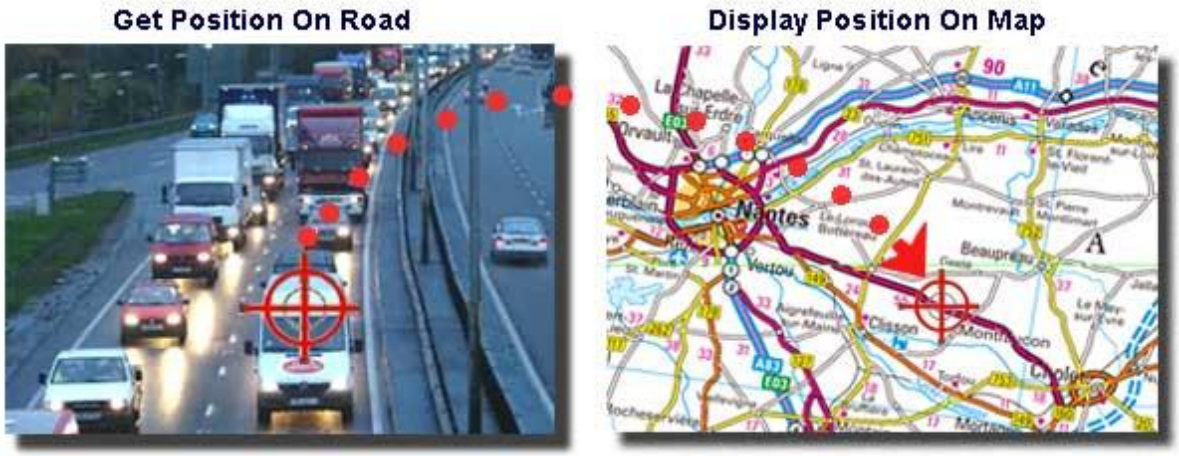

Figura 2.7: Sistema de localización con nodos sensores.

7. **Aplicaciones de aparcamiento.** Gracias a las redes de sensores, se facilita la gestión de aparcamiento de vehículos al tener control de ocupación de las plazas de aparcamiento.

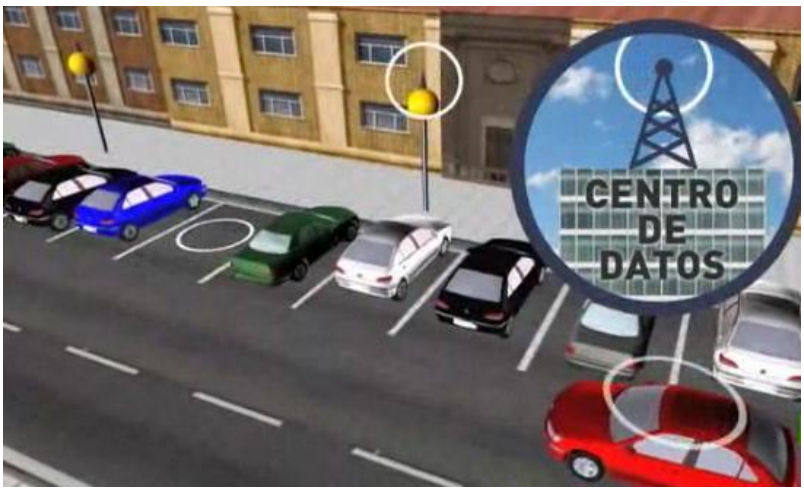

Figura 2.8: Ejemplo de gestión de aparcamiento mediante WSN.

8. **Aplicaciones domóticas.** Dentro del ámbito de la domótica, las redes de sensores tienen numerosas aplicaciones, como el confort, la seguridad y protección, el control de iluminación, entre otras. Algunos ejemplos como el apagar luces al no detectar presencia, o la apertura de cortinas al detectar luz solar, favorecen la optimización y el ahorro de energía.

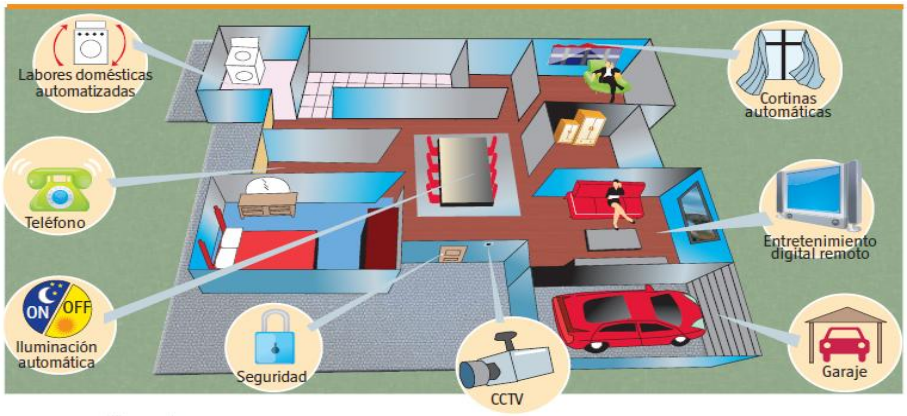

Figura 2.9: Vivienda domotizada. Entorno inteligente.

En resumen, las redes de sensores tienen numerosas aplicaciones, muy útiles para facilitar numerosos ámbitos de la sociedad, y que están en continuo el uso de este tipo de redes no es uno concreto, sino que sigue en proceso de estudio y aumentando sus posibilidades día a día, lo que promete una implantación futura en campos donde actualmente no están en uso.

2.4 Protocolos de comunicación en redes Inalámbricas

2.4.1 IEEE 802.15.4

Es el protocolo de nivel físico y MAC más extendido en redes de sensores. Fue producto de la necesidad de crear un nuevo estándar para redes inalámbricas de bajo consumo y de bajo costo (WSN), para aplicaciones domóticas e industriales debido a que se consideraba que las prestaciones de las redes inalámbricas como la 802.11 (WLAN) eran muy caras y excesivas para este tipo de aplicaciones.

Las características más importantes de este estándar son la flexibilidad de la red, bajo costo y bajo consumo de energía, y se puede utilizar para muchas aplicaciones como dómoticas o médicas, donde se requieren una baja tasa de transmisión de datos.

2.4.2Estándar 6LowPAN(IPv6 overLowPowerWireless Personal Area Network)

El uso de IPv6 en este tipo de redes impone ciertos requerimientos como el incremento de tamaño de las direcciones IPv6 y del MTU en 1280 bytes. Por tal motivo, se dio origen a 6LoWPAN con el fin de eliminar los inconvenientes que tenían los paquetes IPv6 para ser transportados sobre las redes inalámbricas de bajo consumo (LoWPAN), específicamente en las redes basadas en IEEE 802.15.4.

Para lograr este objetivo, se definió una capa de adaptación, que tratara los requerimientos impuestos por IPv6 como el incremento de tamaño de las direcciones IPv6 y del MTU en 1280 bytes y se crearon un conjunto de encabezados que permitieron una codificación eficiente de los encabezados y direcciones IPv6 dentro de encabezados comprimidos más pequeños hasta alcanzar en algunas ocasiones los 4 bytes.

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

De esta forma, nos podremos conectar con otros sensores que estén a cierta distancia y enviar datos.

Sus principales características son:

- Utiliza IPv6 lo que proporciona conectividad con otras redes IP e interoperabilidad.
- IPv6 proporciona un número de direcciones elevado, lo que es ideal para las redes con un gran número de sensores.
- Es un estándar, lo cual ya es una ventaja de por sí, pero además proporciona dos ventajas adicionales.

- La posibilidad de disponer de varios fabricantes de motes (Por ejemplo: Crossbowmote o moteiv entre otras)
- La posibilidad de disponer de diferentes sistemas operativos (*open-source*), aunque esto también implica al fabricante (Ejemplos son: tinyOS o Contiki)
- Los requerimientos de memoria son muy bajos debido a que el *stack* sólo es hasta el nivel IP (uIPv6 utiliza 11K de memoria ROM y 1.8 RAM)

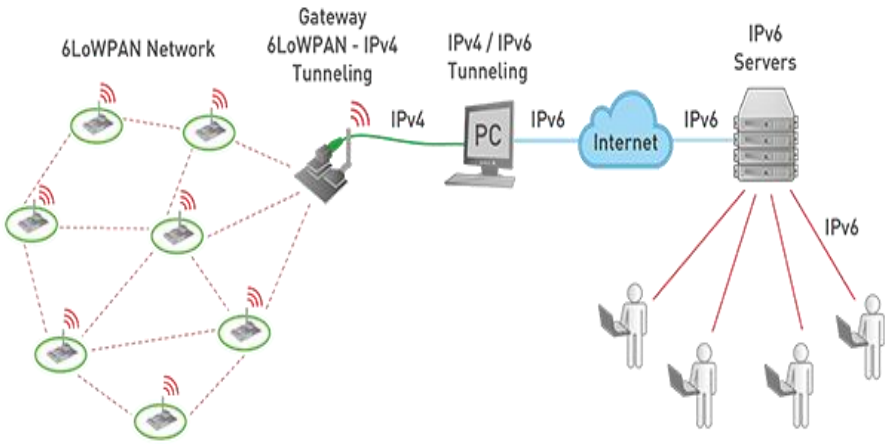

Figura 2.10: Esquema del protocolo 6LoWPAN.

2.4.3 Protocolo CoAP

ConstrainedApplicationProtocol (CoAP) es un software de protocolo creado para ser usado en dispositivos electrónicos muy sencillos, que les permite comunicarse entre ellos a través de Internet. Está particularmente destinado para sensores de baja potencia, switches, válvulas y componentes similares que necesitan estar controlados o supervisados remotamente, a través de redes standard de Internet.

CoAP es un protocolo de la capa de aplicación destinado al uso en redes de recursos de dispositivos de Internet, como las redes de sensores. Está diseñado para la traducción fácil a HTTP y la integración simplificada con la web, a la vez que especializarse en requisitos como soporte multicast, o simpleza, cosas importantes para el Internet de las cosas y dispositivos máquina a máquina, que tienden a estar profundamente incrustados y tienen mucha menos memoria y suministro de potencia que los dispositivos de Internet tradicionales. Por lo tanto, la eficiencia es muy importante. CoAP se puede ejecutar en la mayoría de dispositivos compatibles con UDP.

Se han propuesto las siguientes características para CoAP:

- El diseño del protocolo RESTful minimiza la complejidad de mapeo con HTTP.
- Baja sobrecarga de la cabecera y complejidad de análisis.
- Soporte para URI y tipo de contenido.
- Soporta el descubrimiento de recursos gracias a los servicios CoAP.

- Simple suscripción de un recurso, y notificaciones resultantes.

La unión de CoAP con HTTP también está definida, permitiendo que se usen proxys para proporcionar acceso a los recursos CoAP por medio de HTTP de una manera uniforme.

CoAP hace uso de dos tipos de mensajes, las peticiones y las respuestas, usando un formato de cabecera binario. La cabecera puede estar seguida por opciones en un formato Tipo-Longitud-Valor optimizado. CoAP por defecto está ligado a UDP y opcionalmente a DTLS, proporcionando una seguridad de alto nivel de comunicación.

Todos los bytes a partir de la cabecera del paquete son considerados el cuerpo del mensaje en caso de haberlos. La longitud del cuerpo del mensaje está marcada por la longitud del datagrama. Cuando está ligado a UDP, el mensaje entrante debe caber en un sólo datagrama. Cuando está usado con 6LoWPAN, los mensajes pueden caber en una trama IEEE 802.15.4 para minimizar la fragmentación.

Los métodos soportados en CoAP son *GET*, *POST*, *PUT* y *DELETE*.

2.4.4 REST

REST, REpresentationalState Transfer, es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP para la transmisión de datos sin la necesidad de contar con una capa adicional. Nos permite crear servicios y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda HTTP, por lo que es más simple y convencional que otras alternativas.

Las operaciones se solicitarán mediante *GET*, *POST*, *PUT* y *DELETE*, por lo que no requiere de implementaciones especiales para consumir estos servicios. Este protocolo es adecuado de utilizar cuando se busque mejorar el rendimiento, o cuando se disponga de escasos recursos, como las redes de sensores.

2.5 Interconexión con redes TCP/IP

La aplicación de redes de sensores en áreas como la videovigilancia, el rastreo de objetos, medición de parámetros ambientales, conducen cada vez más a la necesidad de interconectar estas redes a otros dominios ya sean redes de área local, o a Internet, permitiendo de esta manera que las tareas de gestión y control de los datos suministrados por los nodos sensores se puedan realizar de forma remota. Como una consecuencia de la operación aislada en que se encontraban las redes de sensores en el pasado, los fabricantes desarrollaron protocolos especializados y acordes con las limitaciones particulares de los dispositivos de este tipo de redes, esto trajo como consecuencia que hoy en día exista una incompatibilidad entre estos protocolos (como por ejemplo 6lowPAN) y los utilizados en la mayoría de las redes, como es el caso del protocolo TCP/IP utilizado como estándar en Internet.

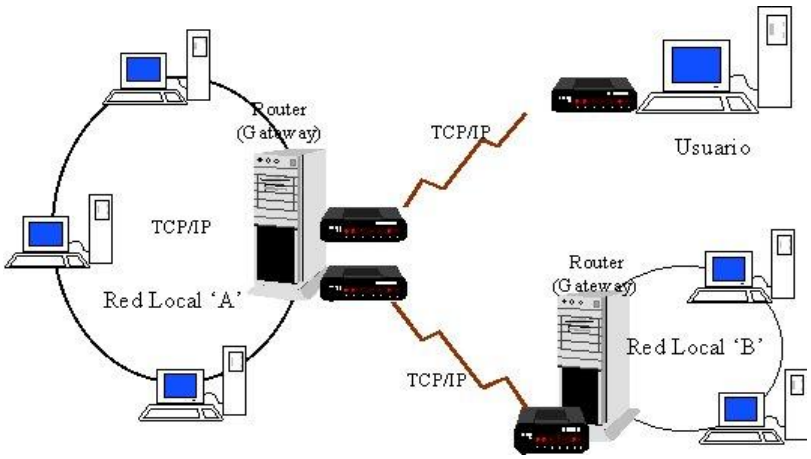

Figura 2.11: Ejemplo de red TCP/IP.

Para resolver estos inconvenientes se han propuesto varios mecanismos, nos centraremos en los mecanismos basados en Proxy, que son los usados en este proyecto.

2.5.1 Mecanismos basados en Proxy

Esta es la alternativa más sencilla y consiste en colocar entre la red de sensores y la red TCP/IP un servidor proxy el cual mantendrá la separación entre ambas redes y realizará las operaciones de traducción de un protocolo a otro. De esta manera ninguno de los dos tipos de redes necesita modificar su funcionamiento logrando así la interconexión entre redes heterogéneas.

A pesar de lo simple y fácil que puede resultar esta alternativa, cabe señalar que la ubicación del servidor proxy rompe la comunicación extremo a extremo entre ambas redes y por lo tanto trae como desventaja que se cree un solo punto de fallo, es decir, toda la comunicación entre ambas redes dependerá de lo robusto y estable que sea el funcionamiento del servidor proxy. Aunque existen alternativas que sugieren la implementación de servidores proxy de respaldo, estos adicionarían complejidad al sistema.

También se debe considerar que existe un servidor proxy específico para cada tipo de tarea a realizar o para un conjunto de protocolos en particular, por lo tanto dependiendo del número de aplicaciones que deseamos ejecutar, se incrementará la cantidad de servidores proxy que se necesitarán, lo que adiciona complejidad a esta alternativa.

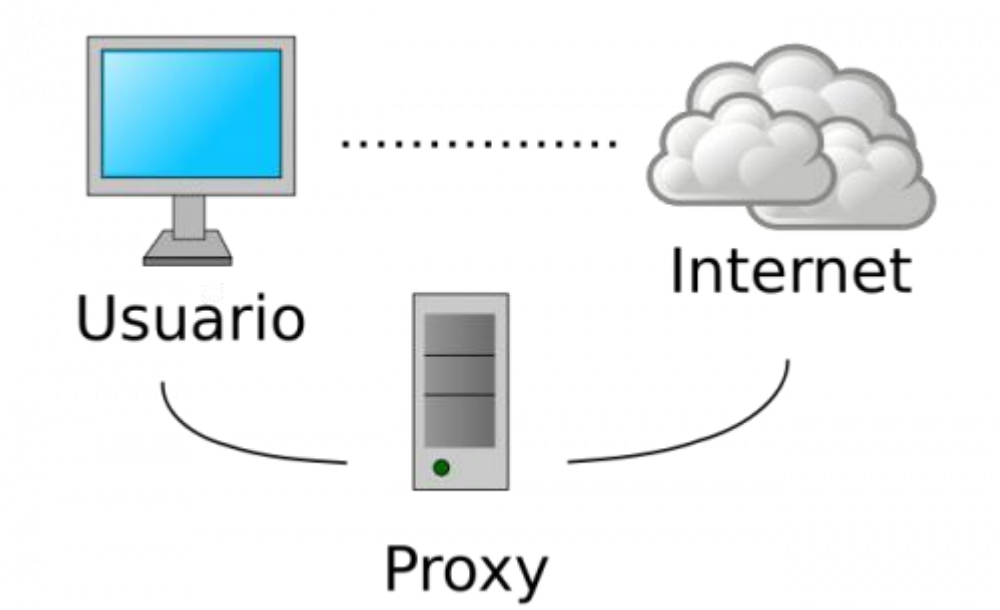

Figura 2.12: Pequeño esquema de una conexión Proxy.

3- ENTORNO DE TRABAJO

3.1 Introducción

A continuación se verán los elementos que se han utilizado a la hora de realizar el proyecto. Se incluirán tanto los dispositivos como el software utilizado.

3.2 Nodos Sensores

3.2.1 Definición

Los nodos sensores, o motes, son los componentes principales de las redes de sensores inalámbricas.

Figura 3.1: Dispositivo utilizado en este proyecto (tmote/moteiv).

Son dispositivos compuestos por un microprocesador con memoria, uno o varios sensores, radio de baja potencia y una batería. Estos nodos están diseñados para adquirir información del entorno, procesarla y transmitir los datos pertinentes a una estación base. Para poder adquirir la información del entorno, los nodos pueden incluir diferentes tipos de sensores, como de temperatura, humedad, luminosidad, voltaje, presencia, etc. Un nodo sensor integra, al menos, los siguientes componentes:

- Radio: el dispositivo de radio proporciona comunicación inalámbrica al nodo sensor, y es compatible con las propiedades específicas de comunicación de las WSN tales como bajo consumo de energía y velocidad de datos, y distancias cortas.

- Microcontrolador: Es el componente que proporciona la lógica computacional y de almacenamiento. Incluye microporcesador y memoria. Utilizado en tareas de procesamiento y manipulación de datos, cifrado, modulación y transmisión digital.

Además de la memoria proporcionada por el microcontrolador, hay modelos que disponen de una memoria externa adicional, por ejemplo, memoria flash.

- Fuente de energía: el suministro de energía debe alimentar al nodo durante bastante tiempo, dependiendo de la aplicación. Normalmente, la alimentación del nodo se lleva a cabo a través de USB o pilas AA. En la actualidad existen nodos sensores alimentados por baterías de litio pueden ser recargados a partir de una fuente de recarga o energía auxiliar.

- Unidad sensora, los nodos pueden disponer de una serie de sensores que se encargan de medir alguna magnitud física en el entorno donde estén posicionados. Entre los diferentes tipos de sensores nos encontramos el de temperatura, iluminación, presión, sonido, acústico, de presencia, etc.

3.2.2 TelosB

El nodo sensor TelosB es un dispositivo de bajo consumo y alta recolección de datos, que lleva integrados tanto los sensores como la radio (Chipcon CC2420) y el microcontrolador (MSP430), además de ser fácil de programar. Permanece a la espera durante la mayoría del tiempo, y se activa sólo para realizar la acción asignada, como enviar, y vuelve a ponerse a la espera. Posee un interfaz USB para conectar a un PC. Las características principales de este dispositivo son:

- Interactúa con otros dispositivos IEEE 802.15.4.
- Transmisor Chipcon inalámbrico de 250Kbps 2.4GHz IEEE 802.15.4.
- Compatibilidad global con las bandas ISM (2.4 hasta 2.4835 GHz).
- Velocidad de 250 kbps.
- Antena integrada en la placa base.
- Microcontrolador MSP430 (8 MHz con 10 kB de RAM y 48 Kb de Flash)
- Bajo consumo.
- Programación y recogida de datos por USB.
- Tiene sensores de luz, temperatura y humedad.
- Opera con TinyOS.

Gracias a su interfaz USB, es fácilmente programable a través del PC. Además es uno de los nodos que poseen un menor consumo en comparación con otros nodos sensores.

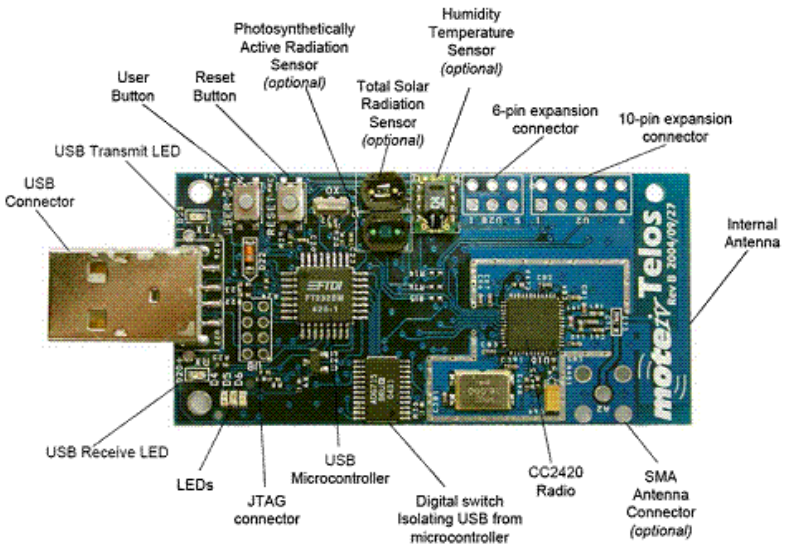

Figura 3.2:Diferentes componentes del mote empleado en el proyecto.

3.3 Sistema operativoTinyOS para redes de sensores

3.3: Logotipo del sistema operativo TinyOS, usado en el proyecto.

TinyOS es un sistema operativo útil para pequeños dispositivos como los motes, y que funciona a partir de eventos producidos que llamarán a funciones.

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.

Está escrito en el lenguaje de programación nesC como un conjunto de tareas y procesos que colaboran entre sí. Los programas están compuestos por componentes que se enlazan para formar un programa completo. Una pieza clave para la programación de los nodos de una red inalámbrica de sensores es usar un modelo de programación basado en eventos para poder soportar las características de concurrencia que estas redes necesitan en sus nodos.

En general, TinyOS puede verse actualmente como la plataforma estándar de implementación para WSN.

3.4 Lenguajes de programación

3.4.1 Generalidades

La programación de los motes es relativamente compleja, debido a su limitada capacidad de cálculo y la escasa cantidad de recursos. Además, para los microcontroladores usados en los motes, en la actualidad, no se encuentran disponibles entornos de depuración de código, con lo que puede llegar a ser muy complicado encontrar algún error en el código.

Entre los lenguajes de programación disponibles, nos encontramos con nesC, que está directamente relacionado con TinyOS. En este proyecto no se va a usar, pero se hablará de él ya que con él se trabaja sobre los nodos sensores y su programación.

3.4.2 NesC

NesC (Network EmbeddedSystems C) es un lenguaje de programación con una sintaxis similar a C, optimizado para las limitaciones de memoria y capacidad computacional presentes en las redes inalámbricas de sensores. NesC está orientado a componentes y especialmente diseñado para programar en redes de sensores bajo el sistema operativo TinyOS. Surgió de la necesidad de desarrollar un nuevo lenguaje de programación específico para redes de sensores inalámbricos,y viene motivada por el tipo de aplicaciones que se desarrolla, que pueden ser aplicaciones basadas en recolección, difusión y control de la información obtenida del sensor, tienen que reaccionar ante cambios en su entorno, aplicaciones que precisan de control de errores en el manejo de datos, aplicaciones en tiempo real, etc.

3.4.3 JAVA

En este proyecto se ha usado Java para la programación de la aplicación que configura los sensores. Es un lenguaje que se encuentra estrechamente relacionado con TinyOS(la mayoría de herramientas de apoyo de las aplicaciones de TinyOS están realizadas en Java).

Java es un lenguaje de programación desarrollado por Sun Microsystems. Es un lenguaje orientado a objetos y toma mucha parte de su sintaxis de C y C++ pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel.

Las aplicaciones Java están típicamente compiladas en un “bytecode” y en tiempo de ejecución, el “bytecode” es interpretado o compilado a código nativo para su ejecución.

Desde noviembre de 2006 Sun Microsystems liberó la mayor parte de sus tecnologías Java se encuentran bajo la licencia GNU GPL de forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java aún no lo es).

3.5 WIRESHARK

Wireshark, antes conocido como Ethereal, es un analizador de protocolos utilizado para realizar análisis y solucionar problemas en redes de comunicaciones, para desarrollo de software y protocolos, y como una herramienta didáctica.

Permite ver todo el tráfico que pasa a través de una red (usualmente una red Ethernet, aunque es compatible con algunas otras) estableciendo la configuración en modo promiscuo. Permite examinar datos de una red en tiempo real o de un archivo de captura guardado en disco. Se puede analizar la información capturada, a través de los detalles y sumarios por cada paquete. Wireshark incluye un completo lenguaje para filtrar lo que queremos ver y la habilidad de mostrar el flujo reconstruido de una sesión de TCP.

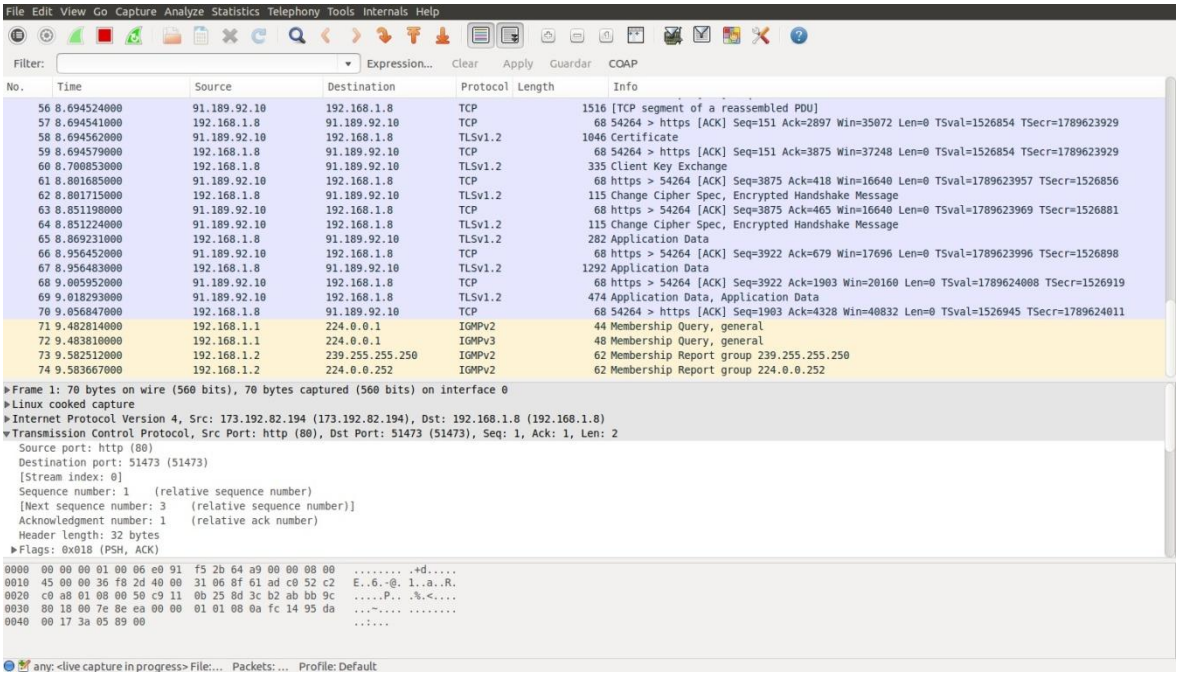

Figura 3.4: Ejemplo de captura de paquetes mediante Wireshark.

3.6 PROXY HTTP/COAP

Un proxy, en una red informática, es un programa o dispositivo que realiza una acción en representación de otro, esto es, si una hipotética máquina A solicita un recurso a una C, lo hará mediante una petición a B; C entonces no sabrá que la petición procedió originalmente de A. Esta situación estratégica de punto intermedio suele ser aprovechada para soportar una serie de funcionalidades: proporcionar caché, control de acceso, registro del tráfico, prohibir cierto tipo de tráfico, etc.

Su finalidad más habitual es la de servidor proxy, que consiste en interceptar las conexiones de red que un cliente hace a un servidor de destino, por varios motivos posibles como seguridad, rendimiento, anonimato, etc. Esta función de servidor proxy puede ser realizada por un programa o dispositivo. En nuestro caso, el Proxy será HTTP/CoAP que transformará la información a CoAP y viceversa.

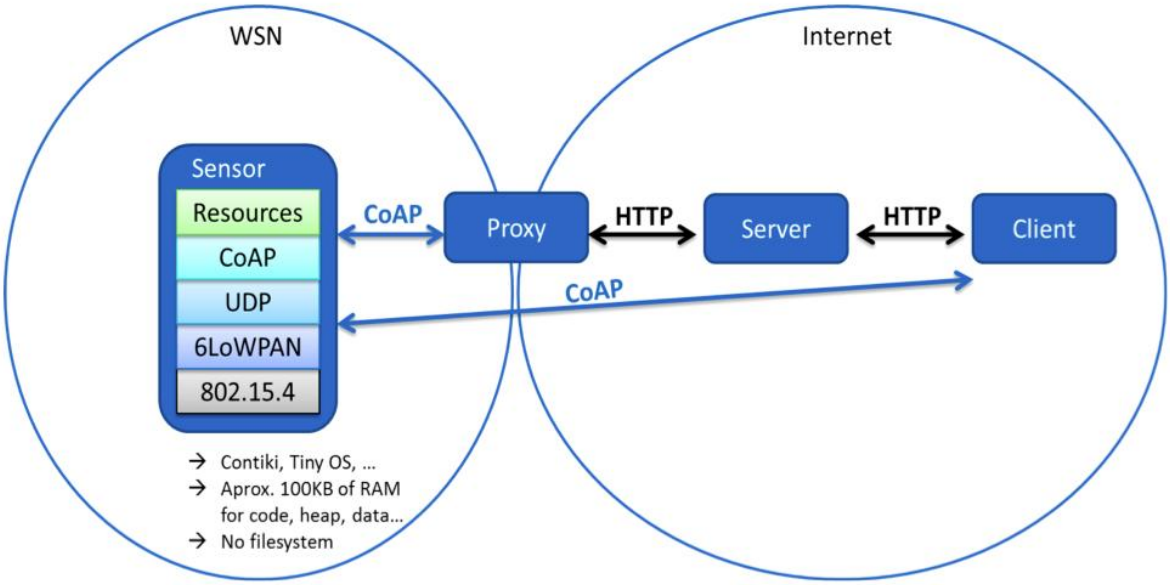

Figura 3.5: Esquema del uso del Proxy HTTP/COAP, empleado en el proyecto.

4- DESARROLLO DE LA APLICACION

4.1 Introducción

Después de analizar los componentes que han servido para realizar este proyecto, se pasará al desarrollo del mismo. Se explicará cómo se ha llegado a su finalización, así como los posibles errores que han ido surgiendo y su solución.

Se dispone de una interfaz gráfica que permitirá programar el comportamiento de los nodos de una red de sensores, aplicado al ámbito de la domótica. Para ello, dará la opción de definir en los nodos, ciertos umbrales que, al ser sobrepasados, harán que el nodo elegido, o el propio nodo que detecta ese umbral, haga saltar alarmas y realizar una determinada acción, también definida en la aplicación. Dicha aplicación se encargará de crear un fichero con la configuración y la acción a realizar por el nodo que se quiera, y enviarlo al nodo de la red inalámbrica que se haya establecido,transformando el fichero al formato adecuado a través de un proxy.

4.2 Conceptos

A continuación se van a definir los diferentes elementos que intervienen en este proyecto.

4.2.1. Ficheros

Una vez elegida la configuración que queremos que lleve nuestro mote, se creará un fichero en el PC, en lacarpeta "ficheros" del directorio raíz de la aplicación. Este fichero contendrá sólo una línea con toda la información necesaria para la configuración de los motes. El fichero tendrá la forma "recurso/umbral/acción/mote que realizará la acción", y, a la hora de usar el proxy, éste leerá su contenido y lo traducirá a CoAP, para luego cargarlo en el mote.

4.2.2. Recurso

En REST existen los recursos, que pueden ser accedidos utilizando un identificador global (URI). Los elementos de la red se intercambiarán *representaciones* de estos recursos. Así, una aplicación puede interactuar con un recurso conociendo el identificador del recurso y la acción requerida, no necesitando conocer si existen otros elementos.

En este proyecto se considerará recurso a las magnitudes detectadas por los motes, y no en el sentido de REST. Se trabajará con temperatura, humedad, voltaje y luz, que en el fichero se escribirá como t, h, v y l respectivamente.

4.2.2. Umbral

Gracias a la aplicación, podremos elegir el valor máximo o umbral de un sensor, que hará que, el mote elegido, realice la determinada acción. Ese umbral se comparará con el valor captado por los motes, si ese valor es mayor que el umbral, se llevará a cabo la acción. A la hora de escribir en el fichero, el umbral estará en formato ascii, para mayor facilidad a la hora de traducir con el proxy.

4.2.3. Acción

Será la tarea que ejecute el mote elegido, al sobrepasar el umbral establecido. En este caso, la acción será encender los leds del mote. Al haber tres leds distintos (rojo, azul y verde), hay ocho posibilidades diferentes de acción, que se escribirán en el fichero con la inicial del led o leds configurado por el usuario.

4.2.4. Mote que realizará la acción

Este campo tendrá la IPv6 del mote que realizará la acción establecida por el usuario. La terminación será el número de mote elegido.

4.2.5. REST

Arquitectura de desarrollo web que nos permite crear servicios y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda HTTP. Se ha utilizado para enviar el contenido del fichero al proxy, y para leer valores de los recursos. Se usarán operaciones de GET y PUT. Si se quiere leer cualquier recurso que detecte el mote, se usará la opción GET, y si lo que se quiere es enviar el contenido del fichero al proxy, se tendrá que elegir la opción PUT.

4.2.6. CoAP

El formato de tramas que emplean los motes es CoAP, es necesario realizar una conversión de formatos, que se lleva a cabo mediante un proxy HTTP/CoAP. Mirando los diferentes campos de las tramas CoAP a través de Wireshark, se configura el proxy con la información adecuada, para el correcto funcionamiento.

El proxy se encarga de obtener el contenido del fichero, y transformarlo de HTTP a CoAP, usando los diferentes parámetros que se han obtenido. Una vez transformado a CoAP, el contenido se envía al mote adecuado.

4.2 Estudio previo.

Antes de empezar con la aplicación, había que investigar qué tipo y formato de tramas enviaban los sensores entre sí. Para ello había que realizar una conexión básica desde terminal, siguiendo el tutorial de la página de CoAP, mientras con Wireshark se analizaba el tráfico de la red. Con esto, se averiguaron diferentes opciones y campos de configuración de las tramas CoAP, por lo que se dedujo que se necesitaba un Proxy que transformase el contenido del ficheroque se iba a enviar, con las opciones y los campos adecuados.

Uno de los problemas con los que se ha contado es la falta de información que hay sobre dicho Proxy, por lo que siguiendo la poca información de foros y ayudas de la página oficial, se ha conseguido hacer funcionar.

Con el Proxy funcionando y la información necesaria, comenzó la creación de la aplicación, que se explicará a continuación.

4.3 Aplicación.

La aplicación estará programada en lenguaje Java. Una vez arrancada la aplicación y pasada la página de presentación, se puede apreciar la ventana principal. En ésta, se encontrarán dos partes en dos zonas. En la zona de la izquierda se ve una imagen dividida por una rejilla. Sobre esta imagen,se puedencolocar los motes en el lugar que se quiera de la imagen, simulando el lugar de la casa/edificio donde se colocarían dichos sensores. Como apunte, hay que destacar, que en las posteriores listas de motes que mostrará la aplicación, éstos estarán en el orden de colocación en la imagen. En la segunda zona de la ventana principal, aparecen tres botones: el primero permite seleccionar la imagen que se quiera; la segunda, permite pasar a la configuración de los motes; y la tercera, pasar al envío u obtención de la configuración. A partir de aquí, se explicará cómo sería el uso normal de la aplicación, teniendo en cuenta que no hay ficheros de configuración previos.

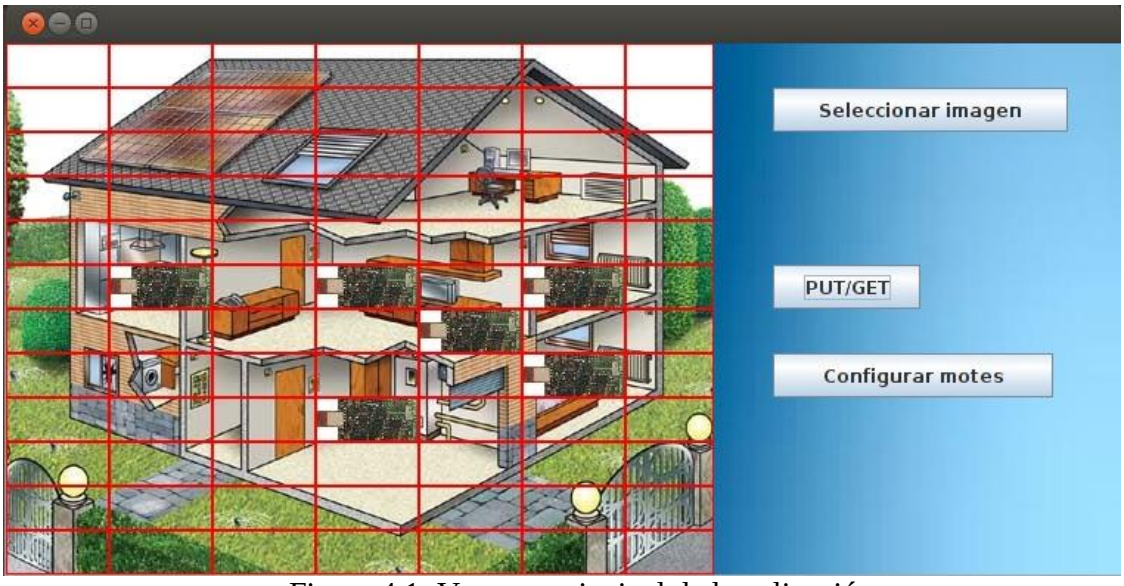

Figura 4.1: Ventana principal de la aplicación.

1. En la ventana principal se colocan los motes sobre la imagen elegida, y se pulsa en "configurar motes".Aparecerá la siguiente ventana:

Figura 4.2: Ventana “Configurar motes”. 1- Sirve para añadir un nuevo recurso. 2- Aparecerá el recurso elegido con el umbral. 3- Nombre del fichero. 4- Crear el fichero con nombre (3). 5- Abrir fichero con nombre (3).

2. Se selecciona la opción añadir recurso, y aparecerá otra ventana con diferentes opciones.

Figura 4.3: Ventana “Añadir recurso”.

3. Se selecciona el recurso que se quiera, y se escribe el umbral de dicho recurso. En la parte de abajo, se eligen los leds que encenderá el mote que contenga la dirección IPv6 escrita, al sobrepasar el umbral.

4. Al aceptar, se volverá a la ventana anterior, y se verá que se ha añadido al panel de texto la letra inicial del recurso añadido, un valor,las letras “r”, “v”, “a” según qué Leds queramos que se enciendan, y la IPv6 del mote que realizará la acción al sobrepasar el valor elegido. Eso será el contenido del fichero de configuración. Ya sólo quedaría escribir el nombre del fichero y darle a crear. Si se quiere comprobar qué contiene un fichero ya creado, sólo hay que escribir su nombre y darle a "Abrir existente".

Figura 4.4: Ejemplo de recurso añadido en un fichero “prueba”.

5. Al pulsar "crear fichero", se abrirá el fichero, y se comprobará que aparece la configuración elegida.

6. Una vez comprobado el fichero, se procederá a enviarlo. En la ventana principal, se elegirá "put/get". Saldrá la ventana que se ve a continuación:

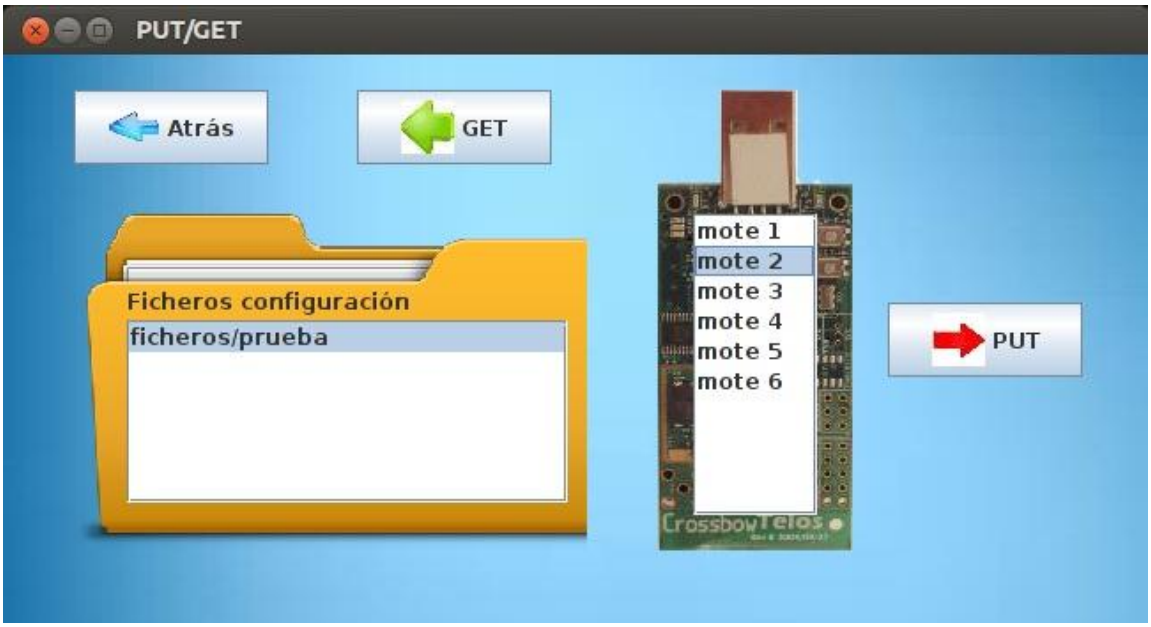

Figura 4.5: Ventana “PUT/GET”.

En esta ventana, aparecen dos listas, la de la izquierda mostrará todos los ficheros de configuración existentes. La derecha, la lista de motes que se han colocado. Para enviar la configuración al mote, sólo hay que seleccionar el fichero con la configuración que se

quiera, y el mote al que se quiere enviar, y darle a "PUT".

En caso de que se quiera obtener algún valor de algún recurso, pulsar "GET", y en la siguiente ventana, seleccionar el mote del que se quiera obtener el valor, y el recurso que se quiera saber.

Figura 4.6: Ventana "GET". Lista de motes y recursos.

8. En la parte en la que se hace PUT, la aplicación configurará el Proxy, y enviará la información, para transformarla a formato CoAP, y enviarla al sensor elegido. Como sería complicado comprobar la temperatura, voltaje, etc., y depende de otro proyecto complementario, para comprobar el correcto funcionamiento de la aplicación, mientras ésta se usa, se tendrá Wireshark abierto, y se comprobará qué tipo de tramas se envían al sensor. Éstas deberían ser del tipo CoAP, y en la parte de Datos, comprobar que aparece la trama que se va a enviar al mote. De ser así, se confirmará que se ha hecho correctamente. A partir de aquí, se supondrá que el sensor tiene la configuración elegida y, en este caso, encenderá el led del mote que se haya elegido, cuando pase el umbral indicado.

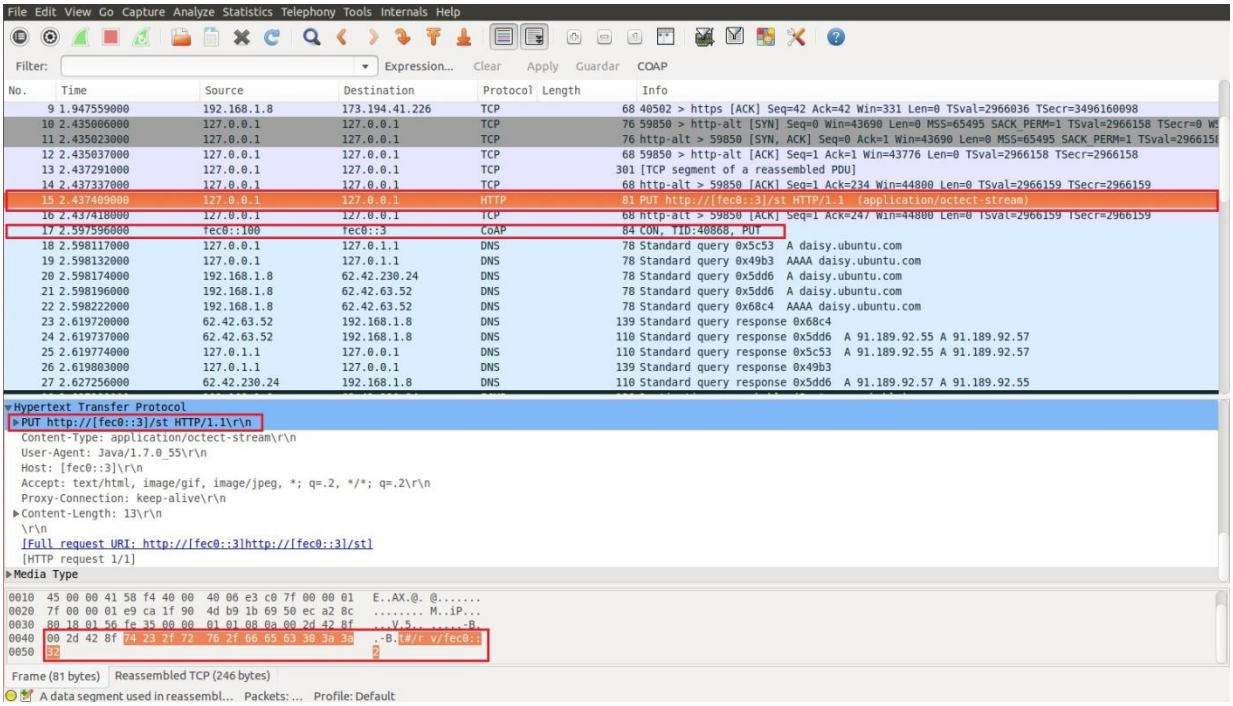

Figura 4.7: Captura Wireshark con el PUT hecho. Se puede apreciar la trama HTTP inicial, con la dirección del mote. La siguiente trama sería la CoAP, ya transformada por el proxy. Por último, se ha recuadrado la zona de abajo de Wireshark, para comprobar que la trama que se ha enviado aparece en el campo de datos.

5- CONCLUSIÓN Y LÍNEAS FUTURAS

Con la realización de este proyecto se ha pretendido estudiar la configuración de una red de sensores, para su uso a la hora de analizar el entorno.

En este proyecto se ha creado una interfaz gráfica que permite definir en los nodos, ciertos umbrales que, al ser sobrepasados, harán que el nodo elegido, o el propio nodo que detecta ese umbral, haga saltar alarmas y realizar una determinada acción, también definida en la aplicación. En este caso, dicha acción es el encendido de los leds, pero podría ser cualquier aplicación domótica conectada a la red de sensores.

La aplicación generará un fichero con la información necesaria para enviar al nodo elegido, que se almacenará en el directorio raíz de dicha aplicación. Este fichero tendrá un tamaño reducido para poder ser enviado y almacenado en los nodos.

A la hora de cargar el fichero en los nodos, se ha utilizado un servicio web REST, con las operaciones de GET y PUT. Para la tarea de traducción del contenido de los ficheros a CoAP, se ha usado un proxy que se encarga de transformar las peticiones y respuestas entre HTTP y CoAP.

En este proyecto, se ha empezado analizando las redes de sensores y el estándar que se ha usado para su comunicación (6LoWPAN). Se ha explicado el uso de un Proxy, necesario para poder transformar las tramas HTTP a formato CoAP y viceversa.

Se ha acabado explicando el funcionamiento de la aplicación, para configurar una red de sensores en una casa o edificio y las acciones que realizarían los motes. Hasta acabar con una comprobación de que la configuración se ha completado con éxito.

De cara a futuras mejoras referentes a este proyecto, se pueden añadir algunas opciones, o mejorar otras, como se verá a continuación.

Se podría usar direccionamiento privado IPv6 en la red de sensores, y que las peticiones HTTP se dirigiesen a la IP pública del gateway, donde, por medio del proxy, se transformasen las URIs a las que se accede.

Otra mejora, podría ser una opción para saber qué configuración tiene un mote. Para ello, de una lista con los motes que hay en la red, se elige uno y la aplicación mostraría el recurso que tiene activado y el umbral.

Se podrían extender las funcionalidades de la aplicación, permitiendo que se mostrasen alarmas. Los motes, cuando se sobrepasase un umbral, informarían a la aplicación. Ésta estaría escuchando continuamente, y cada vez que un mote informase, mostraría por pantalla el mote y el valor que ha detectado.

Una mejora, que podría ser interesante, sería la de hacer el port de la aplicación a Android o IOS. Una aplicación de este tipo en un dispositivo móvil o tablet, facilitaría mucho las cosas a la hora de configurar los sensores desde cualquier parte, sin tener que depender del ordenador, y de la poca movilidad de éste.

Otra ampliación podría ser la de añadir una opción con ficheros de configuración, para evitar estar colocando los motes cada vez que se inicia la aplicación.

Se podría hacer que saltasen las alarmas de los motes, no sólo cuando se sobrepasa el umbral, sino que se podría dar también que salten cuando las lecturas de los sensores bajen de cierto valor.

BIBLIOGRAFÍA

[1] Documentación CoAP.
http://tinyos.stanford.edu/tinyos-wiki/index.php/CoAP_-03

[2] DatasheetTelosB.
http://moss.csc.ncsu.edu/~mueller/rt/rt11/readings/projects/g4/datasheet.pdf

[3] TinyOSDocumentation Wiki.
http://docs.tinyos.net/tinywiki/index.php/Main_Page

[4] Chong C.Y., Kumar S.P., "Sensor Networks: Evolution, Opportunities, and Challenges".

[5] Sohraby K., Minoli D., Znati T., "Wireless Sensor Networks: Technology, Protocols, and Applications" Wiley-Interscience.

[6] Alcaraz Calero J.M., "Tutorial de TinyOS". Universidad de Murcia.

[7] Hill J.L., "SystemArchitectureforWireless Sensor Networks".

[8] Bharathidasan A., Ponduru V.A. S., "Sensor Networks: AnOverview", Department of ComputerScience, University of California, CA 95616, USA.

[9] Akyildiz I.F., Su W., Sankarasubramaniam Y., Cayirci E., "Wireless sensor networks: survey", Broadband and WirelessNetworkingLaboratory, School of Electrical and ComputerEngineering, Georgia Institute of Technology, Atlanta, GA 30332, USA.

[10] Sorabi, K., Gao, J., Ailawadhu, V., and Pottie, J. "ProtocolsforSelf-Organization of a Wireless Sensor Network" ElectricalEngineeringDepartment, UCLA Box 951594, Los Angeles, California, 90095-1594, USA.

[11] Levis P., Madden S., Polastre J., Szewczyk R., Whitehouse K., Woo A., Gay D., Hill J., Welsh M., Brewer E., Culler D., "TinyOS: AnOperatingSystemfor Sensor Networks".

[12] Karl, Holger and Willig, Andreas. "Protocols and ArchitecturesforWireless Sensor Networks".

[13] K.Kim, G.Montenegro y S.Yoo. "6LoWPAN Ad Hoc On-DemandDistance Vector (LOAD)".

[14] Z. Shelby K. Hartke. ObservingResources in CoAP, draft-ietf-core-observe-02.
http://tools.ietf.org/html/draft-ietf-core-observe-02