Escuela Politécnica Superior de Linares

Grado en Ingeniería Telemática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Linares

Trabajo Fin de Grado

PLATAFORMA DE PARKING EXTERIOR INTELIGENTE MEDIANTE RED DE SENSORES INALÁMBRICOS

Alumno: **Pedro Fernández Garrido**

Tutores: Prof. D. José Ángel Fernández Prieto
Prof. D. Joaquín Cañada Bago
Depto.: Ingeniería de Telecomunicación

Octubre, 2015

Resumen

Este Trabajo Fin de Grado consiste en el diseño, programación e implementación de una plataforma de parking exterior inteligente mediante red de sensores inalámbricos. La red estará formada por nodos sensores Waspmote, equipados con sensores de parking, que mediante un módulo de comunicación mandan el estado de la plaza de aparcamiento a una base de datos situada en un router Meshlium. También se diseñará una aplicación web que haga uso de la citada base de datos para mostrar el estado de los sensores de parking.

Se diseñará una red de sensores inalámbricos para cada tipo de módulo de comunicación (802.15.4 y DigiMesh).

Waspmote es un dispositivo sensorial, diseñado para ser usado en ciudades inteligentes, que puede usar diferentes modos de comunicación como Bluetooth, ZigBee o WiFi, abarcando un amplio rango de frecuencias. Además a este dispositivo de carácter básico se le pueden acoplar todo tipo de sensores, de lectura de parámetros ambientales o en este caso un módulo de sensor de parking. Estos dispositivos se programan de una forma sencilla y utilizando un entorno de desarrollo de código abierto. Se configurarán Waspmote tanto de la versión 1.1 como la versión 1.2 para los diferentes tipos de comunicaciones.

Meshlium es un router Linux multiprotocolo con distintas interfaces de comunicación, como los sensores Waspmote, en este caso hará la función de receptor de información además de gestor de la misma. Además será donde se aloje la aplicación web desde donde se podrá el estado de las plazas de aparcamiento.

Se implementará una aplicación web en la que se mostrarán en un mapa el estado de las plazas de aparcamiento, a través de un código de colores (verde libre, rojo ocupado), de manera que sea lo más sencillo de consultar al usuario final. También se diseñará un sistema de búsqueda por localización para que directamente se proporcione el número de plazas libres en una determinada zona.

Índice General

Capítulo 1
INTRODUCCIÓN

En este capítulo se expondrá el porqué de este trabajo fin de grado, desde la problemática existente, a la solución tomada, pasando por los diferentes modos de comunicación estudiados durante el desarrollo del mismo.

La búsqueda de aparcamiento en las grandes ciudades se está convirtiendo en un grave problema, ya que hay estudios que informan que la media mundial de tiempo gastado en búsqueda de aparcamiento es de 20 minutos, aumentando en algunas ciudades españolas como Madrid al doble del tiempo. Además del evidente problema de contaminación ambiental por polución y contaminación acústica, se suma el dato de que la búsqueda de aparcamiento produce un 30% de los atascos en ciudades. [6]

Cada vez más ciudades están intentando combatir esta problemática través del uso de las ciudades inteligentes o Smart Cities, concepto que se refiere a un tipo de desarrollo urbano basado en la sostenibilidad que es capaz de responder adecuadamente a las necesidades básicas de instituciones, empresas, y de los propios habitantes, tanto en el plano económico, como en los aspectos operativos, sociales y ambientales.

Como ejemplo se puede poner a ciudad de Santander, la cual, dentro de su proyecto SmartSantander, ha implementado un sistema de detección de plazas de aparcamiento, como muestra la figura 1.1, utilizando dispositivos de la marca española Libelium. El sistema se trata de una red de sensores inalámbrica formada por nodos sensores Waspmote con función de detección de parking y routers Meshlium.

Figura 1.1 Esquema detección plaza de aparcamiento.
Fuente: Libelium Comunicaciones Distribuidas. [1]

El sistema implementado en Santander funciona de la siguiente manera. Los sensores Waspmote equipados con la placa de detección de aparcamiento se introducen dentro de una capsula estanca, mostrada en la figura 1.2, y se entierran en el asfalto añadiendo por encima un material especial que es detectable a simple vista.

Figura1.2 Waspmote instalado en Santander.
Fuente: Libelium Comunicaciones Distribuidas. [1]

El sensor que se encuentra en estado de reposo (durmiendo), se activa cada cinco minutos y manda al router el estado de la plaza de aparcamiento. Las comunicaciones se realizan a través del uso del protocolo 802.15.4 y la información se almacena en la base da datos de cada router Meshlium, esta es mostrada a los conductores a través de unos paneles situados a lo largo de las calles en la que se encuentran los sensores instalados, como se ve en la figura1.3.

Figura 1.3 Paneles de visualización de plazas libres.
Fuente: Libelium Comunicaciones Distribuidas. [1]

En este Trabajo Fin de Grado se implementará una red de sensores inalámbricos, para el envío del estado de las plazas de aparcamiento, al igual que se hace en Santander con 802.15.4 y además con el nuevo modo de comunicación DigiMesh. Ambos modos de comunicación se implementarán en sensores de la versión 1.1 y de la versión 1.2. El modo de consultar las plazas disponibles será a través de una aplicación web en la que se mostrará el estado de cada plaza y también se podrá consultar las plazas libres en una determinada zona de la ciudad. Los modos de comunicación empleados y las topologías diseñadas se detallan a continuación.

1.1 Modos de comunicación.

A continuación se expondrá las distintas tecnologías de comunicación estudiadas para la transmisión de los datos recogidos por los sensores hasta el servidor.

1.1.1 802.15.4

Según el Instituto de Ingeniería Eléctrica y Electrónica (IEEE), el estándar 802.15.4 define la capa física y de control de acceso al medio para conexiones inalámbricas con una tasa de transmisión de datos baja, para dispositivos tanto móviles como fijos con un consumo de baja potencia o un uso de baterías limitado, en redes de corto alcance. La norma prioriza una baja complejidad con un bajo costo de implementación. [2]

La velocidad de transmisión de datos entre dispositivos es los suficientemente alta (250 Kb/s) para satisfacer un conjunto de aplicaciones, pero también es escalable a las necesidades de transmisión que necesiten los sensores, ya que la tasa de transmisión entre sensores es comúnmente 20 Kb/s.

Dentro de las capas físicas soportadas las más usadas son las destinadas a los dispositivos que operan bajo licencia libre en 868-868.6 MHz, 902-928 MHz y 2400-2483.5MHz. En la banda de 2.4GHz se especifican hasta 16 canales con una separación de 5Mhz, partiendo de la frecuencia de 2.405 GHz y de la enumeración del canal numero 11 hasta llegar al canal número 26 con frecuencia 2.48 GHz. Como se observa en la figura 1.4.

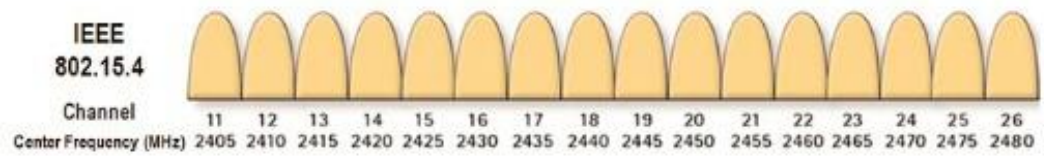


Figura 1.4 Canales 802.15.4 a 2.4 GHz. [3]

Las posibles topologías a utilizar con este estándar son la topología estrella y la punto a punto. A continuación veremos que para este Trabajo Fin de Grado se elegirá la tipología en estrella.

1.1.2 DigiMesh

DigiMesh es un firmware desarrollado por la empresa Digi para los módulos XBee-802.15.4 y XBee-900. Este firmware permite a los módulos dormir, sincronizarse entre ellos y trabajar en igualdad de condiciones, evitando el uso de nodos routers o coordinadores que tienen que estar constantemente en funcionamiento. [4]

Las características ofrecidas por el protocolo implementado son las siguientes:

- Self Healing: cualquier nodo puede unirse a la red o abandonarla en cualquier momento.
-Todos los nodos son iguales: no existen relaciones padre-hijo.
-Silent protocol: reduce la cabecera de enrutamiento debido al uso de un protocolo reactivo similar a AODV (Ad hoc On-Demand Vector Routing).
-Descubrimiento de rutas: en lugar de tener una tabla de rutas, las rutas se descubren cuando se necesitan.
-Mensajes de respuesta (ACKs) selectivos: solo el destinatario responde a los mensajes de ruta.
-Fiabilidad: el uso de ACKs garantiza seguridad de transmisión de datos.
- Modos de reposo: modos de bajo consumo de energía con sincronización para despertar todos al mismo tiempo.

La topología clásica de este tipo de comunicación es de malla, ya que los nodos pueden establecer comunicaciones punto a punto con nodos hermanos mediante el uso de parámetros como la dirección de capa física o la dirección de red, así como realizar conexiones multi-salto, aunque también es posible una topología punto a punto o en estrella.

1.2 Topologías de red

En este apartado se expondrán las distintas tipologías utilizadas en este Trabajo Fin de Grado.

1.2.1 Estrella de un salto

La topología en estrella será la utilizada para las comunicaciones entre los módulos de 802.15.4, ya que el objetivo en cuanto a comunicaciones se refiere, es el envío de daos dense los nodos sensores hasta el receptor que gestione los datos de parking, en este caso un router Meshlium. El router que tendrá función de coordinador se situará en una posición central de forma de que este en la zona de transmisión de todos los nodos sensores, como se muestra en la figura 1.5.

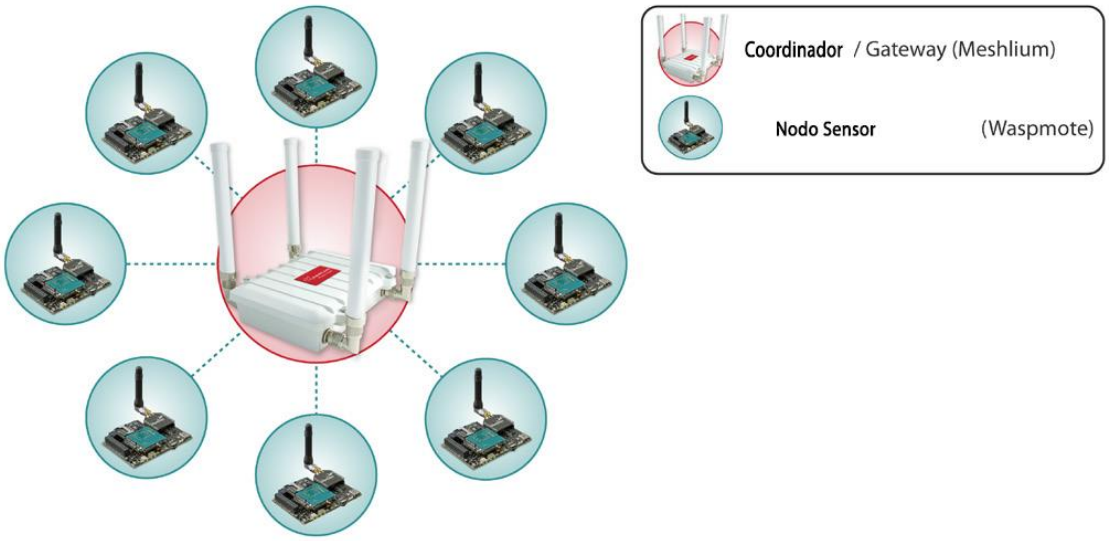


Figura 1.5 Esquema topología en estrella.

Fuente: Libelium Comunicaciones Distribuidas. [4]

En cuanto a la implementación de este sistema en la vía pública cada uno de los nodos sensores se relacionaría e instalaría con su plaza de aparcamiento correspondiente y el coordinador en una posición que estuviera en la zona de alcance de todos los nodos que tenga relacionados. Se puede ver el esquema de red resultante en la figura 1.6.

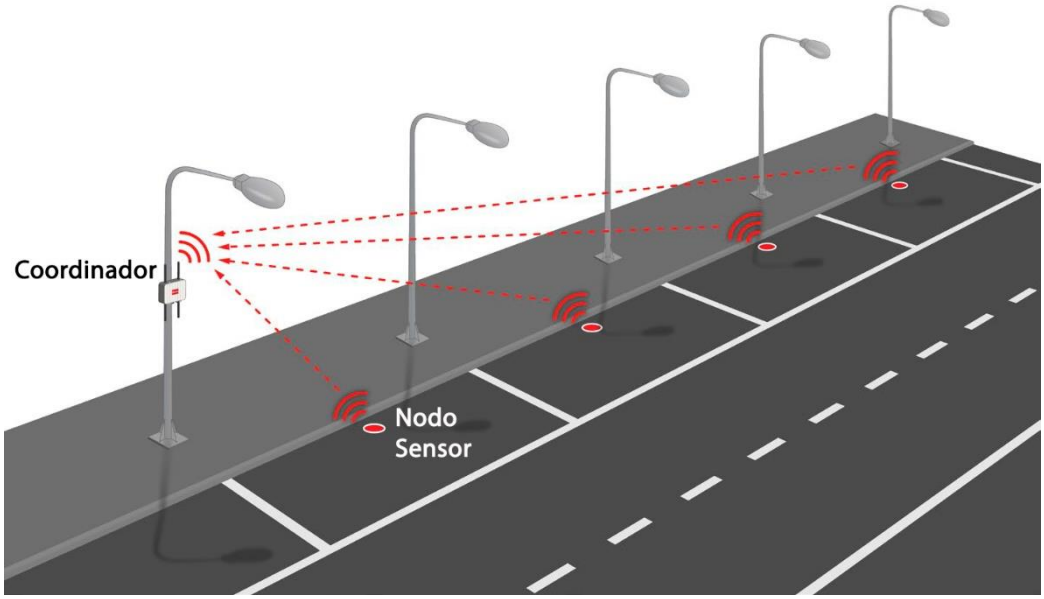


Figura 1.6 Esquema tipología en estrella en la vía pública.

Fuente: Libelium Comunicaciones Distribuidas. [5]

1.2.2 Mallada

La topología mallada será la elegida para las comunicaciones con DigiMesh, ya que con su protocolo de comunicación permite que el modulo receptor no tenga que estar en dentro de la cobertura de todos los nodos sensores, al permitir los módulos el reenvío de paquetes entre ellos. El esquema de red quedaría como se muestra en la figura 1.7. De esta forma cualquier nodo sensor puede enviar sus datos al receptor sin ni siquiera conocer la ubicación de este, lo que se debe cumplir es que todos los nodos formen parte de la misma red y que ninguno quede aislado.

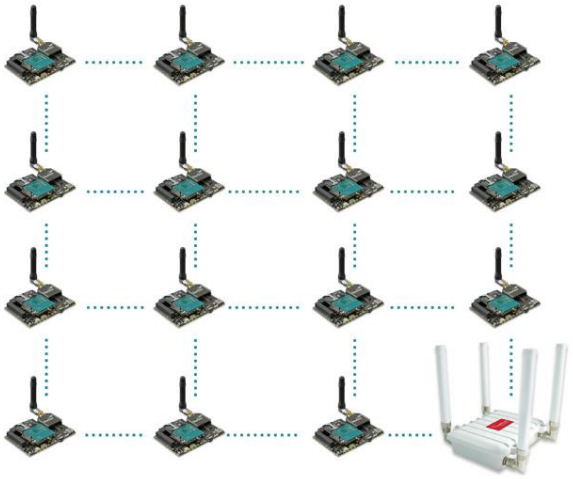


Figura 1.7 Topología mallada.

Fuente: Libelium Comunicaciones Distribuidas. [4]

El escenario de la red mallada en la vía pública se puede explicar a través del esquema de la figura 1.8. En esta situación el receptor no tiene comunicación directa ni con el nodo 3 ni con el nodo 2 pero estos al utilizar DigiMesh pueden llegar a su objetivo a través del nodo 2, por lo que la comunicación seria completa en la red.

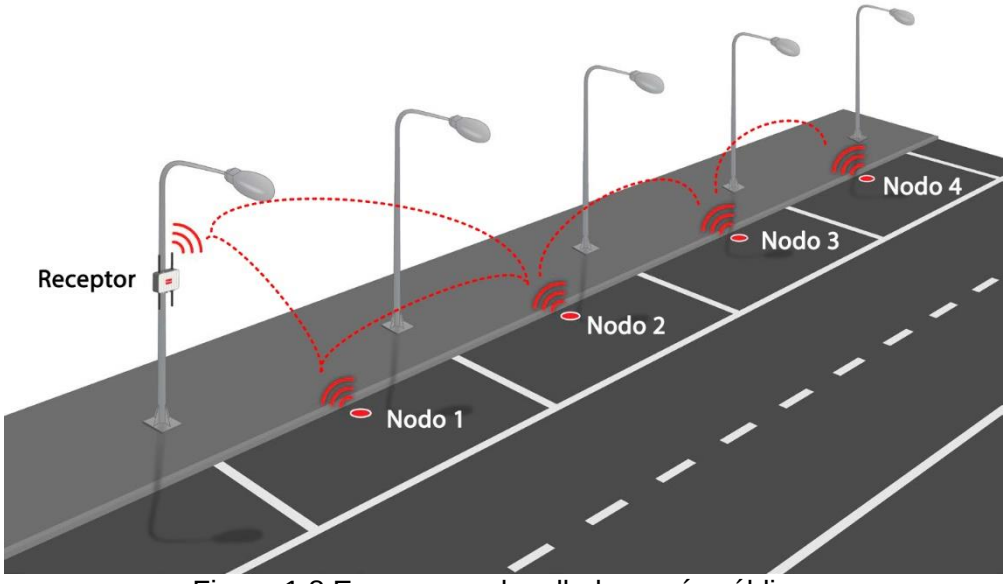


Figura 1.8 Esquema red mallada en vía pública.

Fuente: Libelium Comunicaciones Distribuidas. [5]

Capítulo 2
OBJETIVOS

El objetivo de este trabajo fin de grado es el diseño, programación e implementación de una plataforma de parking exterior inteligente mediante red de sensores inalámbricos. Para llevar a cabo este objetivo se han de realizar los siguientes puntos:

-Se programarán los sensores Waspmote para que realicen las mediciones pertinentes en cuanto al estado de su plaza de aparcamiento correspondiente y envíen los datos resultantes de las mediciones. Esta programación se hará para las dos versiones de los Waspmote 1.1 y 1.2.

-Se estudiarán diferentes modos de comunicación con tal de encontrar el más eficiente tanto económica como energéticamente.

-Se diseñará y se pondrá en funcionamiento una red de sensores inalámbricos para cada tipo de comunicación elegido.

-Se diseñará una base de datos en la cual se darán de alta los sensores de parking y se guardarán sus coordenadas, identificadores y estado de la plaza de aparcamiento.

-Se diseñará y pondrá en funcionamiento una aplicación web para que el usuario final pueda consultar tanto el estado de todas las plazas de aparcamiento monitorizadas como las plazas libres disponibles por zonas.

Capítulo 3
MATERIALES Y MÉTODOS

3.1 Sensor de parking

El sensor de parking, llamado por la empresa distribuidora Libelium “Smart Parking”, permite detectar plazas de aparcamiento disponibles colocando el nodo bajo el pavimento. Se trabaja con un sensor magnético que detecta cuando un vehículo está presente o no. La placa con el sensor de parking, que aparece en la figura 3.1, opera con la variación de campo magnético encima de la misma causada por el chasis del vehículo situado en la plaza de aparcamiento en la que el sensor se ha colocado. Este cambio se detecta utilizando un sensor de campo magnético de permalloy (aleación magnética formada por hierro y níquel), de aquí en adelante MFS, un material cuya resistencia eléctrica varia con el campo magnético que fluye a través de él, esta es medida y procesada para dotar al sistema de una respuesta que se puede utilizar para determinar el estado de la plaza de aparcamiento. [5]

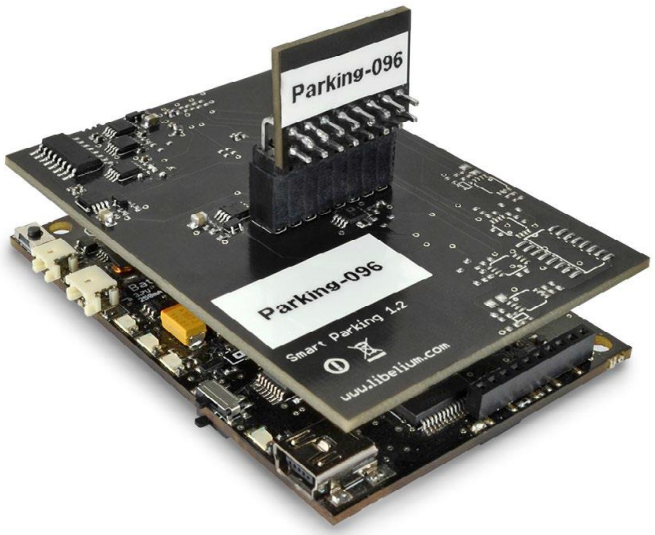


Figura 3.1 Waspmote equipado con placa de Parking. [5]

Fuente: Libelium Comunicaciones Distribuidas.

El MFS incluye un módulo fijado perpendicularmente a la placa, se utiliza para controlar el eje vertical (eje Z) del campo magnético. Cada placa está preparada para funcionar con un solo modulo, por lo que es importante mantener la correspondencia entre ellos. Todos los tableros, módulos y Waspmote están etiquetados con su correspondiente identificador, se puede ver en la imagen 3.2.

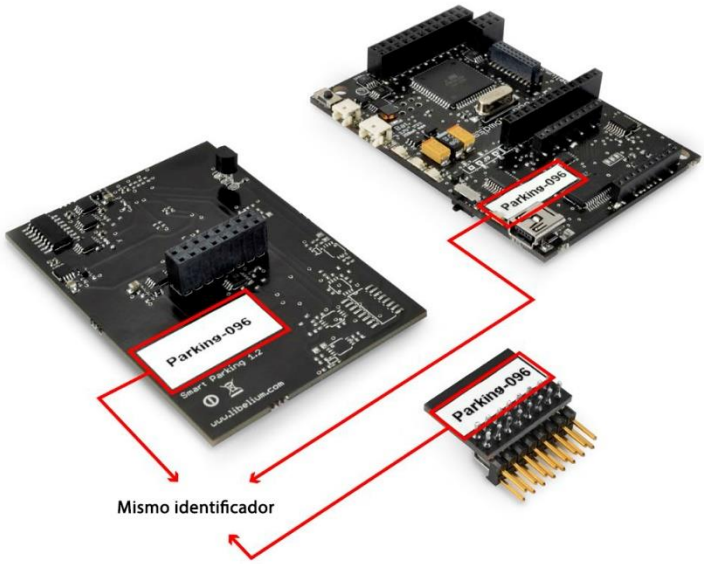


Figura 3.2 Muestra del identificador.

Fuente: Libelium Comunicaciones Distribuidas. [5]

3.1.1 Versiones

Libelium ofrece 2 líneas diferentes de sensor de placa de parking para las redes de sensores inalámbricos: Waspmote OEM y Plug & Sense. El primero se usa para crear aplicaciones para redes o sensores que no son de la marca Libelium y el segundo es de uso exclusivo para la marca y tiene unas nuevas funcionalidades añadidas.

Las nuevas funcionalidades que introduce la placa Plug & Sense son:

-Reseteo magnético: el Waspmote asociado esta modificado para poder resetear el sensor pasando un imán especifico por encima del encapsulado.

-Gel: el set incorpora un gel que se añade cuando se instala el sensor en el encapsulamiento, lo que le otorga resistencia a la humedad.

-Encapsulamiento y espuma de relleno también incluidos en el set.

Sin embargo la segunda versión aunque mejorada también tiene algunas limitaciones. La primera es los modos de transmisión permitidos para el sensor, que solo son los recomendados por el fabricante, en este caso 802.15.4 y DigiMesh. Además la segunda limitación es la batería, ya que al Waspmote asociado no se le permite recargar baterías, debido a que viene pre configurado para el uso de baterías no recargables.

Para la realización de este Trabajo Fin de Grado se tienen disponible las dos versiones, la primera asociada a un Waspmote de la versión 1.1 y la Plug & Sense con un Waspmote de la versión 1.2 asociado.

3.1.2 Proceso de medición

El MFS consiste básicamente en una película delgada de permalloy cuya resistencia depende del campo magnético a través de él (intensidad y orientación). Esta película está integrada en un puente de resistencias que oscila entre 600Ω y 1200Ω (típicamente 850Ω), por lo tanto entre los dos terminales de salida del sensor se tiene una tensión en el orden de la 3,5mV / gauss con una tensión de alimentación de 5V.

El MFS es un sensor de tres ejes que permite supervisar el campo magnético en cualquier dirección espacial. La salida de cada uno de los ejes se amplifica y se filtra para evitar interferencias causadas por campos magnéticos externos. Las salidas de los tres ejes se leen directamente usando el conversor del Waspmote de analógico a digital en los pines de entrada analógicas ANALOG1, ANALOG2 y ANALOG5.

Para leer las salidas basta con ejecutar las funciones “readParking” o “readParkingSetReset” de la biblioteca de la API, en la que se captura la tensión analógica en la entrada del conversor de analógico a digital. En la segunda de estas funciones un pequeño impulso de corriente (500 mA de pico, un microsegundo de ancho) se aplica a un circuito interno antes de leer el sensor de modo que las moléculas de la película de permalloy se realinean con el campo magnético, lo que elimina los efectos no deseados debidos a la histéresis del material.

La placa también incorpora un sensor de temperatura con el fin de compensar los efectos que este parámetro tiene en el sensor. La salida del sensor en el pin ANALOG7 se puede leer usando la función “readTemperature”, asignado su salida a una variable entera. En cuanto a la compensación de temperatura, Libelium facilita los sensores calibrados para un rango de operación entre 0º y 45º, con los parámetros necesarios para calcular la referencia en función de la nueva temperatura almacenada en la memoria EEPROM del sensor. [5]

En este proyecto se hará solamente uso de la función “readParking”, que devuelve el estado de la plaza de aparcamiento, aunque todas los demás procesos descritos se utilizan en el apartado de set up para calibrar el sensor.

3.1.3 Instalación

El sensor se ha de instalar en un punto de implementación óptimo. Este punto será la posición en la que la probabilidad de detección sea máxima, lo que implica reducir al mínimo las probabilidades de detección falsa, causadas por otros vehículos en plazas

colindantes o por campos magnéticos ajenos al sensor, y de falso rechazo, debido a una baja variación del campo magnético encima del sensor con un vehículo situado en la plaza asociada.

Este lugar dependerá del tipo de aparcamiento que vamos a monitorear. En el caso del estacionamiento en serie, figura 3.3, el sensor debe de ser colocado debajo de uno de los lados del coche. En el caso de estacionamiento en batería, figura 3.4, el lugar más adecuado será el más cercano al motor o a la parte trasera del vehículo.

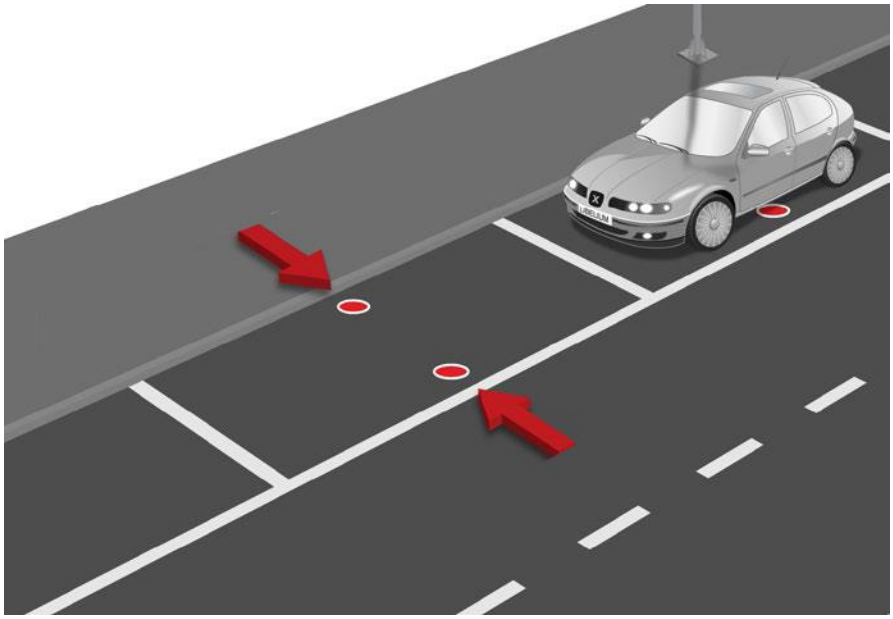


Figura 3.3 Estacionamiento en serie.
Fuente: Libelium Comunicaciones Distribuidas. [5]

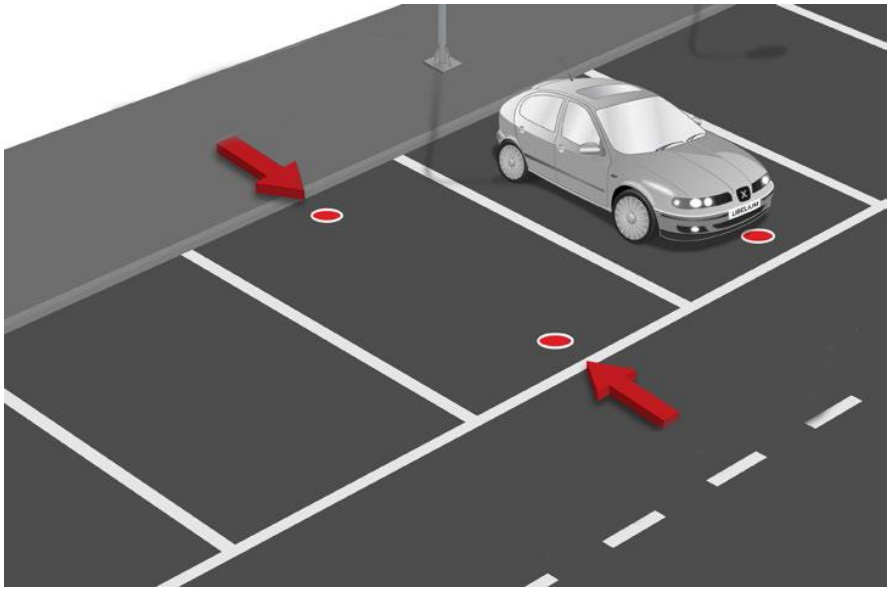


Figura 3.4 Estacionamiento en batería.
Fuente: Libelium Comunicaciones Distribuidas. [5]

Una vez elegido el punto de disposición de los sensores se pasará a la instalación de los mismos en la vía. Para ello primero se ha de instalar el sensor y la placa dentro del encapsulamiento y se necesitan los elementos mostrados en la figura 3.5, que son los siguientes:

-(1) Waspmote mas placa de sensor de parking y módulo de comunicaciones.

-(2) Batería no recargable.

-(3) Encapsulado de PVC.

-(4) Set de geles.

-(5) Espuma de relleno.

Figura3.5 elementos para la instalación. [5]

Para introducir el Waspmote dentro del encapsulamiento la batería debe de ser colocada en el agujero dedicado en el relleno de espuma de color rosa, teniendo cuidado que el sensor descansa en posición horizontal y con la antena en posición vertical. El sensor magnético de reseteo debe permanecer lo más cerca posible de la parte superior del recinto, pero permitiendo que este se pueda cerrar correctamente. Una distancia apropiada sería dejar un centímetro de margen entre el sensor magnético y la parte superior del encapsulamiento. Quedando como se puede ver en la figura 3.6. Además, es recomendable fijar el sensor magnético con un poco de cinta adhesiva para que no se pueda mover.

Figura 3.6 Situación sensor magnético dentro del encapsulado.
Fuente: Libelium Comunicaciones Distribuidas. [5]

A continuación, se ha de preparar el gel para introducirlo dentro del encapsulado. Se prepara mezclando los dos tipos de componentes mostrados en la figura 3.5, a partes iguales. Para cada sensor que se vallan a instalar se necesitan 650 ml de mezcla, por lo que se necesitan 325 ml de cada tipo de gel. La mezcla se debe de agitar el tiempo que sea necesario para asegurar que los dos componentes estén mezclados completamente, ya que de no ser así la mezcla se convertiría a estado sólido.

Ya está todo preparado para colocar el sensor en el lugar elegido para su colocación, así que lo siguiente será realizar el agujero donde introducir el encapsulado de PVC, como se puede observar en las figuras 3.7 y 3.8. Este debe de ser los suficientemente grande para que el conjunto se encuentre totalmente enterrado, pero teniendo en cuenta que sea lo más superficial posible para que el módulo de transmisión tenga las menores perdidas posibles.

Figura 3.7 Obras para la introducir el encapsulado en el suelo.

Fuente: Libelium Comunicaciones Distribuidas. [5]

Figura 3.8 Agujero final para la instalación.

Fuente: Libelium Comunicaciones Distribuidas. [5]

A continuación se ha de colocar el encapsulado dentro del agujero antes de encender el sensor, ya que al encenderse toma las variables X, Y y Z para obtener una correcta calibración del sensor de parking, así que una vez encendido no se puede mover. Seguidamente, se vierte el resultado de la mezcla de geles hasta que cubran todos los componentes electrónicos salvo el sensor magnético de reseteo y la parte superior de la antena perteneciente al módulo de comunicaciones. El interior del encapsulado deberá quedar como se muestra en el esquema de la figura 3.9.

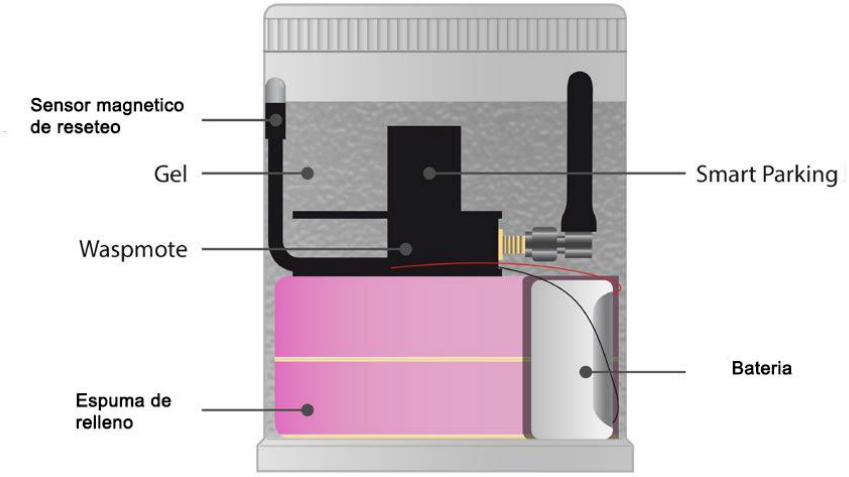


Figura 3.9 Esquema del interior del encapsulado.

Fuente: Libelium Comunicaciones Distribuidas. [5]

Seguidamente se ha de cerrar la capsula y comprobar el correcto funcionamiento del sensor haciendo un reseteo del mismo y comprobando si envía la información de la plaza de aparcamiento correctamente. Después el hueco que queda entre el asfalto y en encapsulado se rellenará con una resina epoxi y debe de rellenar todo el hueco y recubrir la parte superior del encapsulamiento, haciendo que el agua de lluvia y la humedad nunca pude llegar hasta el sensor. El esquema de la figura 3.10 muestra como de quedar la instalación.

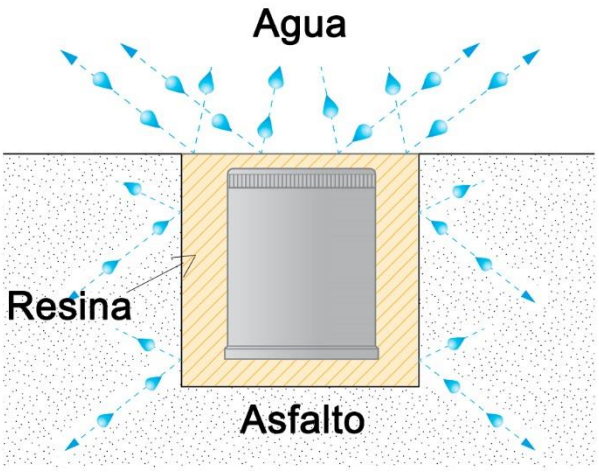


Figura 3.10 Esquema instalación en asfalto.
Fuente: Libelium Comunicaciones Distribuidas. [5]

3.2 Comunicaciones

En este apartado se explicará cómo se configuran los modos de transmisión, se expondrá las características de los dos tipos de comunicaciones estudiados, las topologías diseñadas y una comparación entre ambos.

3.2.1 Configuración módulos de transmisión.

Todas las configuraciones que se deban de realizar en los módulos de transmisión se harán del programa XCTU, proporcionado por la empresa fabricante de los módulos que se van a utilizar, la marca Digi. Para conectar los módulos al ordenador y asi poder usar el programa, se utilizará una estación base fabricada por la marca Libelium, mostrada en la figura 3.11.

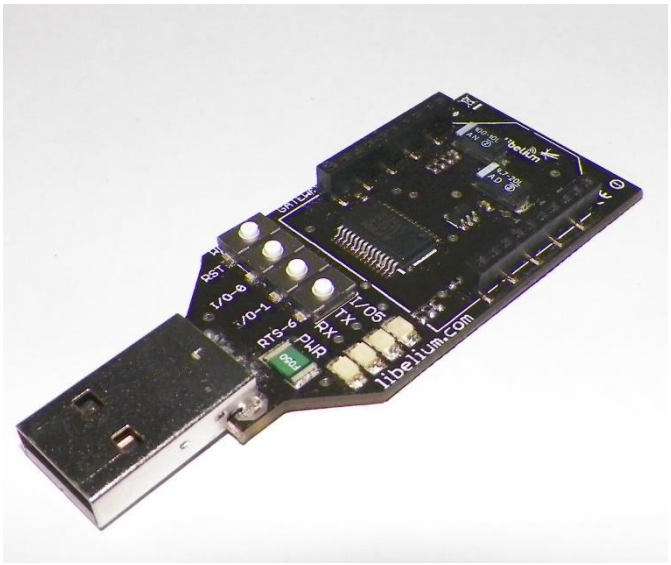


Figura 3.11 Estación base para módulos de comunicación.

A continuación se explicará los pasos que hay que realizar para poder instalar y utilizar el programa citado para configurar los módulos.

El programa es totalmente gratuito y se puede descargar desde la página de la misma empresa, en la siguiente dirección:

http://www.digi.com/support/productdetail?pid=3352.

Después se seleccionará la versión necesaria, en este caso la correspondiente a Windows de 64 bit, este paso se muestra en la figura 3.12, y se descargará.

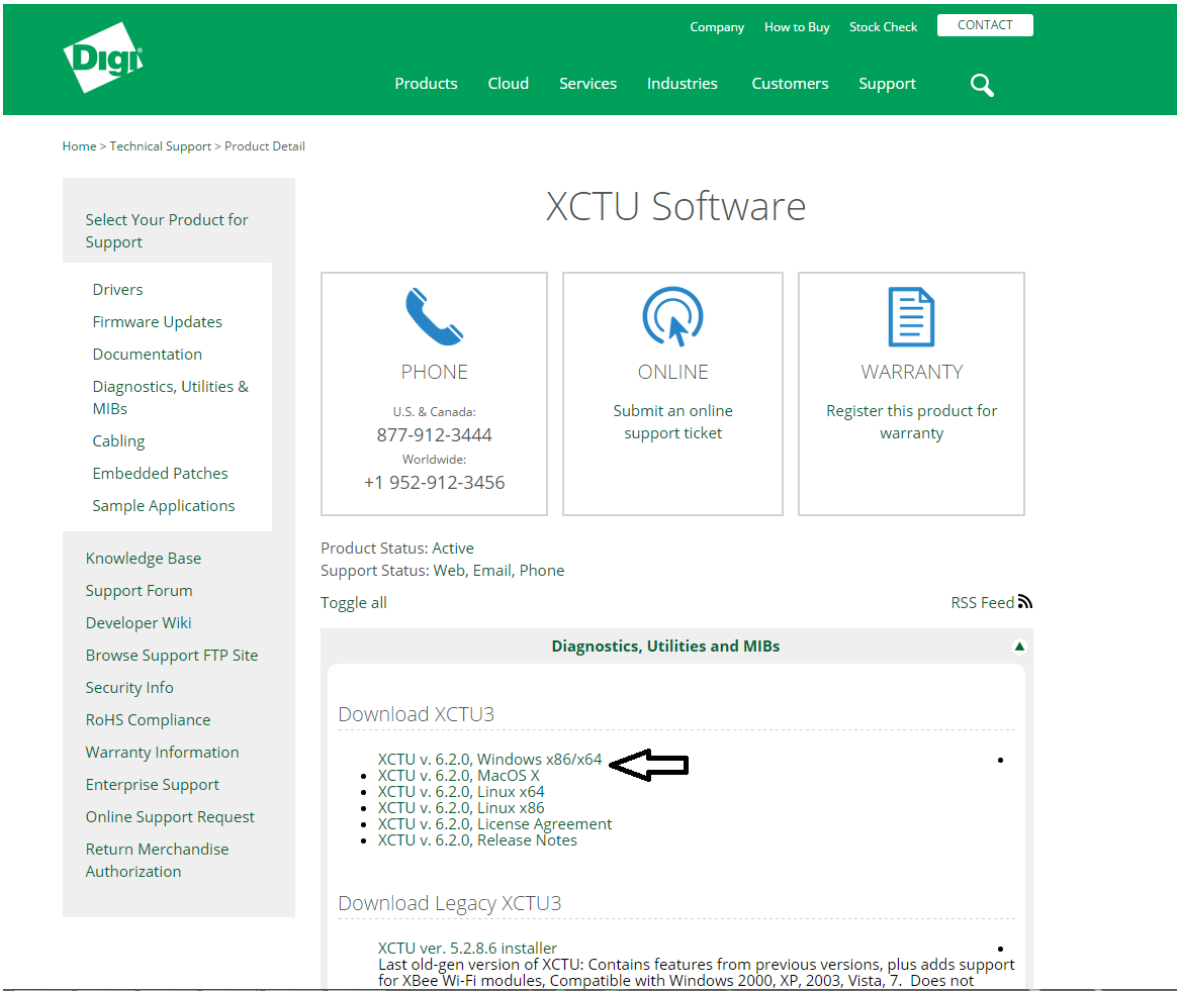


Figura 3.12 Selección de versión a descargar.

Una vez descargado se ha de ejecutar el programa y se abrirá el instalador del mismo. En primera instancia el ayudante de instalación mostrará el dialogo que se muestra en la figura 3.13.

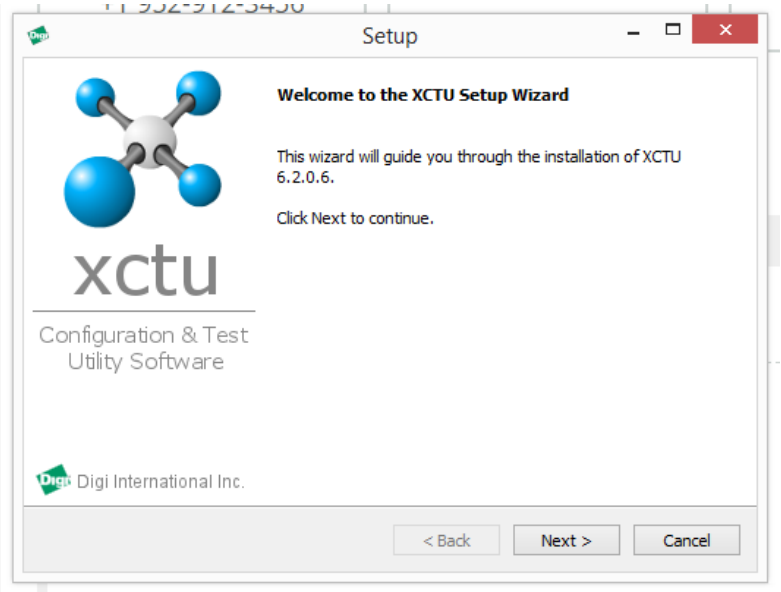


Figura 3.13 Asistente de instalación XCTU.

Se tendrá que pulsar en siguiente, en las páginas que aparecen a continuación se pedirá que se acepte la licencia del programa y que se seleccione la carpeta de instalación. Una vez introducidos estos datos comenzará la instalación del programa, como se muestra en la figura 3.14.

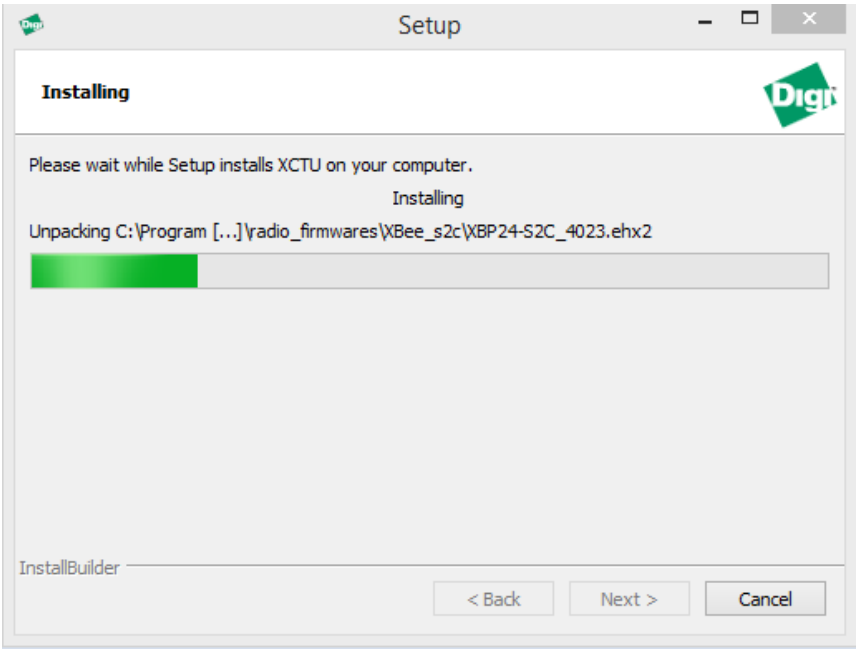


Figura 3.14 Instalación del programa XCTU.

Una vez realizada la instalación ya se estará en disposición de ejecutar el programa, y se mostrará como aparece en la figura 3.15.

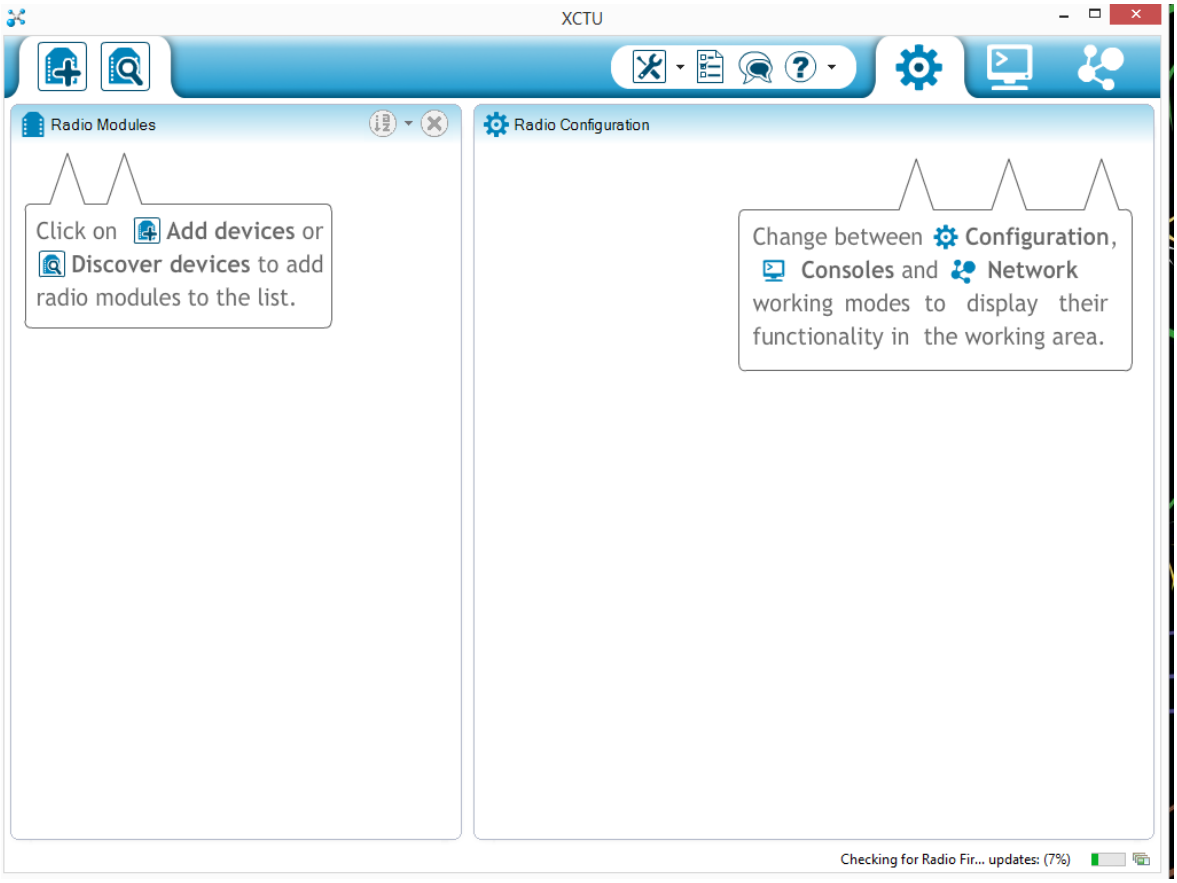


Figura 3.15 Primera visualización del programa XCTU.

Si se pulsa en el apartado de añadir un nuevo módulo radio se desplegará un formulario, como se ven en la figura 3.16, en el que habrá que seleccionar el puerto serie donde se encuentra conectado el módulo, la velocidad de transmisión en Baudios y algunos parámetros que se dejarán por defecto.

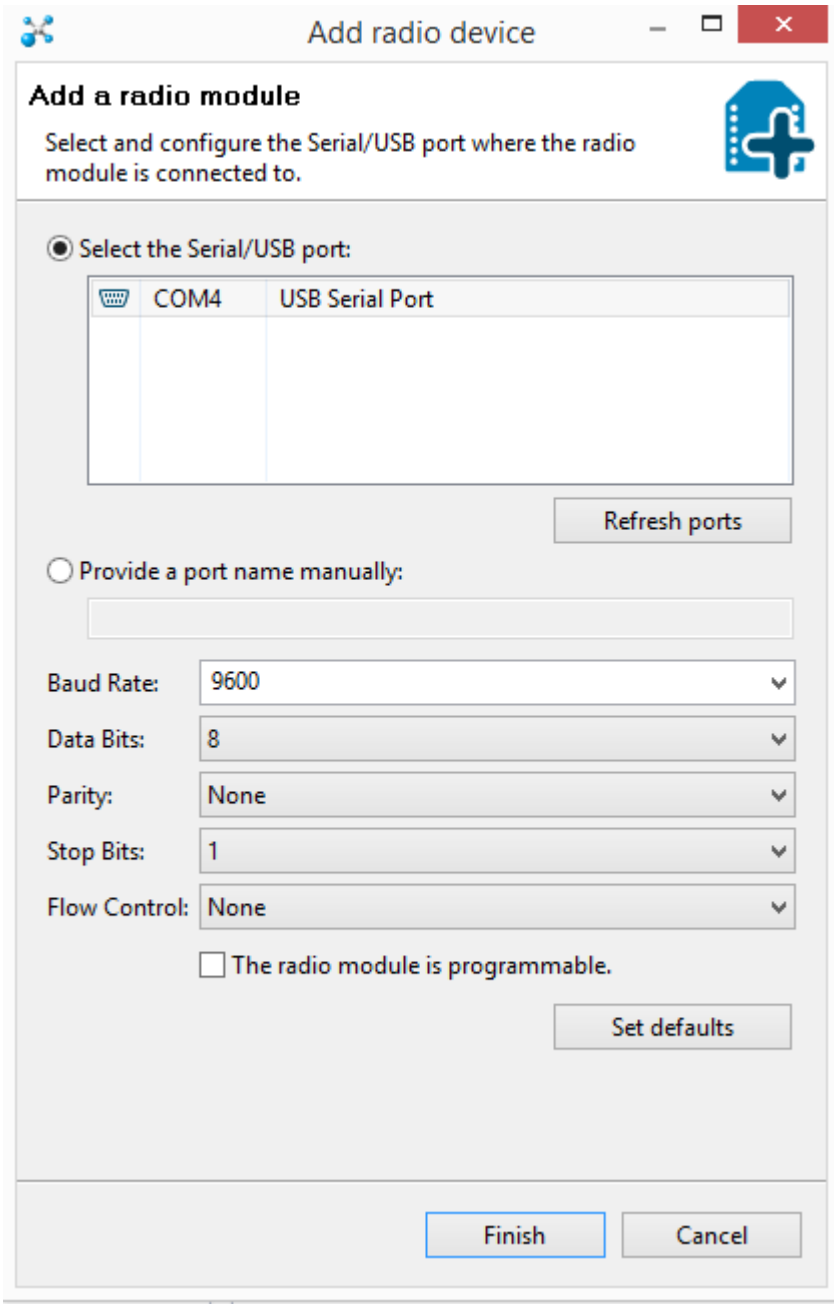


Figura 3.16 Ventana se búsqueda de módulos de radio conectados.

Una vez realizada la búsqueda se mostrará el listado con los módulos de radio encontrados. Se puede ver en la figura 3.17. En el caso de no tener ninguno conectado o haber escogido en puerto equivocado, se dará un aviso de que no se ha encontrado ninguno.

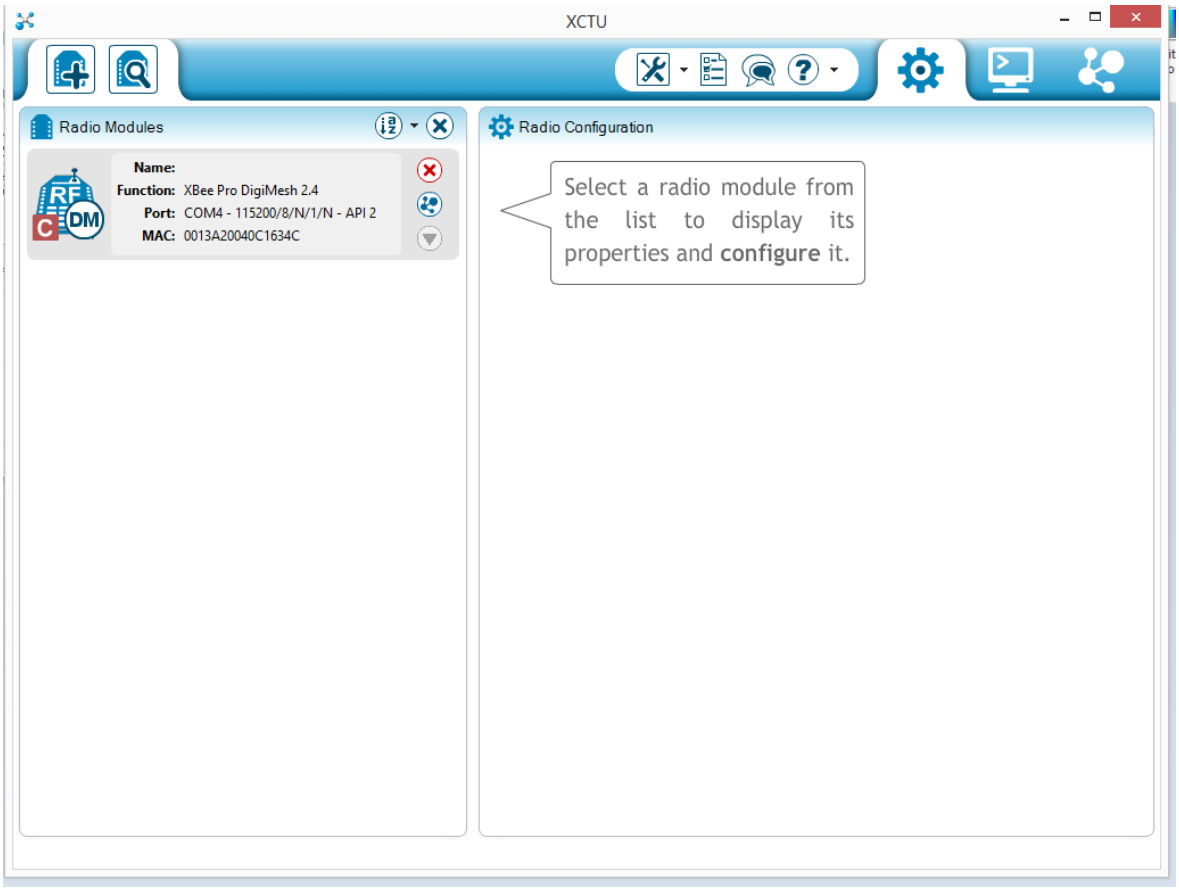


Figura 3.17 Muestra de los módulos de radio encontrados por el buscador.

Si se pulsa encima de uno de los módulos de la lista en el apartado Radio Configurations, aparecerán tanto los datos de la versión de Firmware que posee el modulo, como todos los parámetros que tiene configurados y que se pueden cambiar. Como muestra la figura 3.18. En este apartado será donde se configuren todos los parámetros necesarios para el correcto funcionamiento de las comunicaciones.

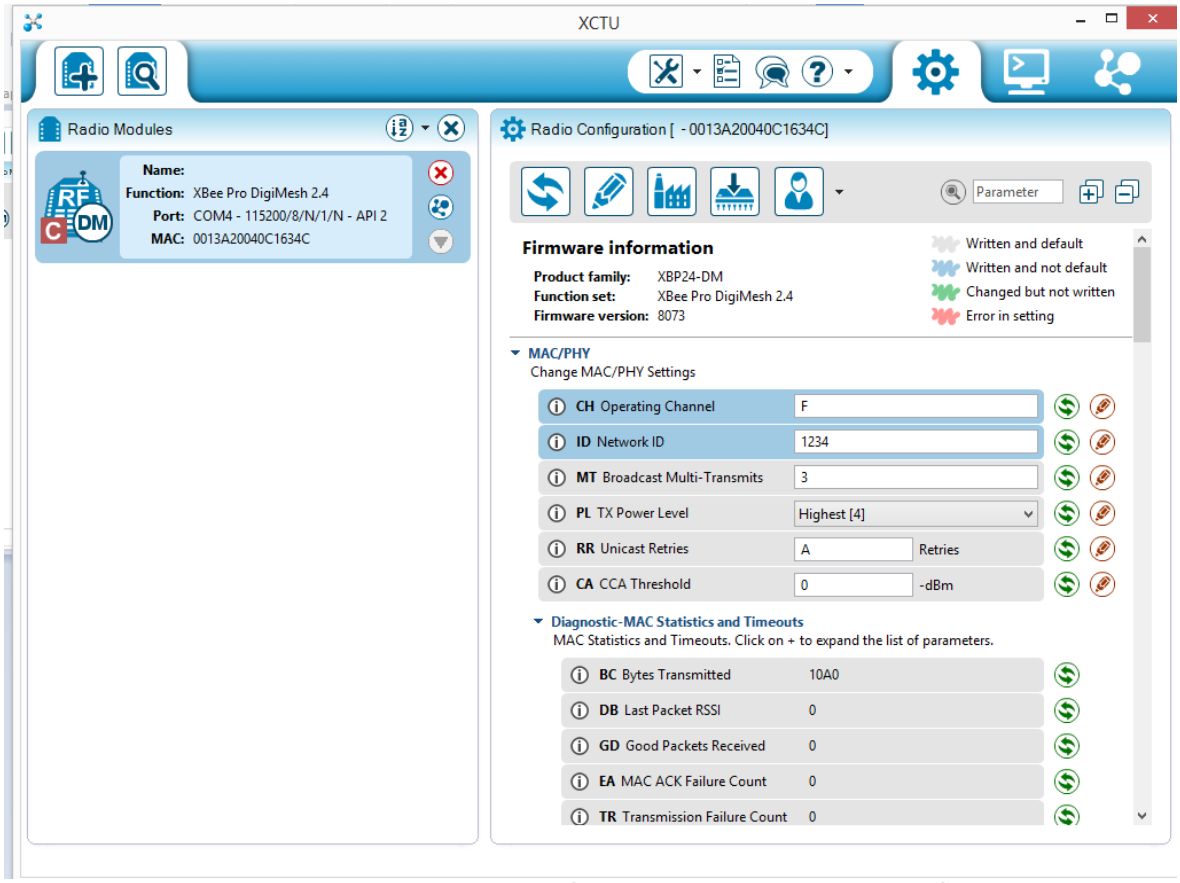


Figura 3.18 Muestra de los parámetros configurables del módulo.

Si se pulsa en el apartado de la consola situado en la parte superior derecha, como se ve en la figura 3.19, se accederá a una ventana donde se puede establecer conexión con el módulo y se pueden ver todos los mensajes que envía y recibe el mismo.

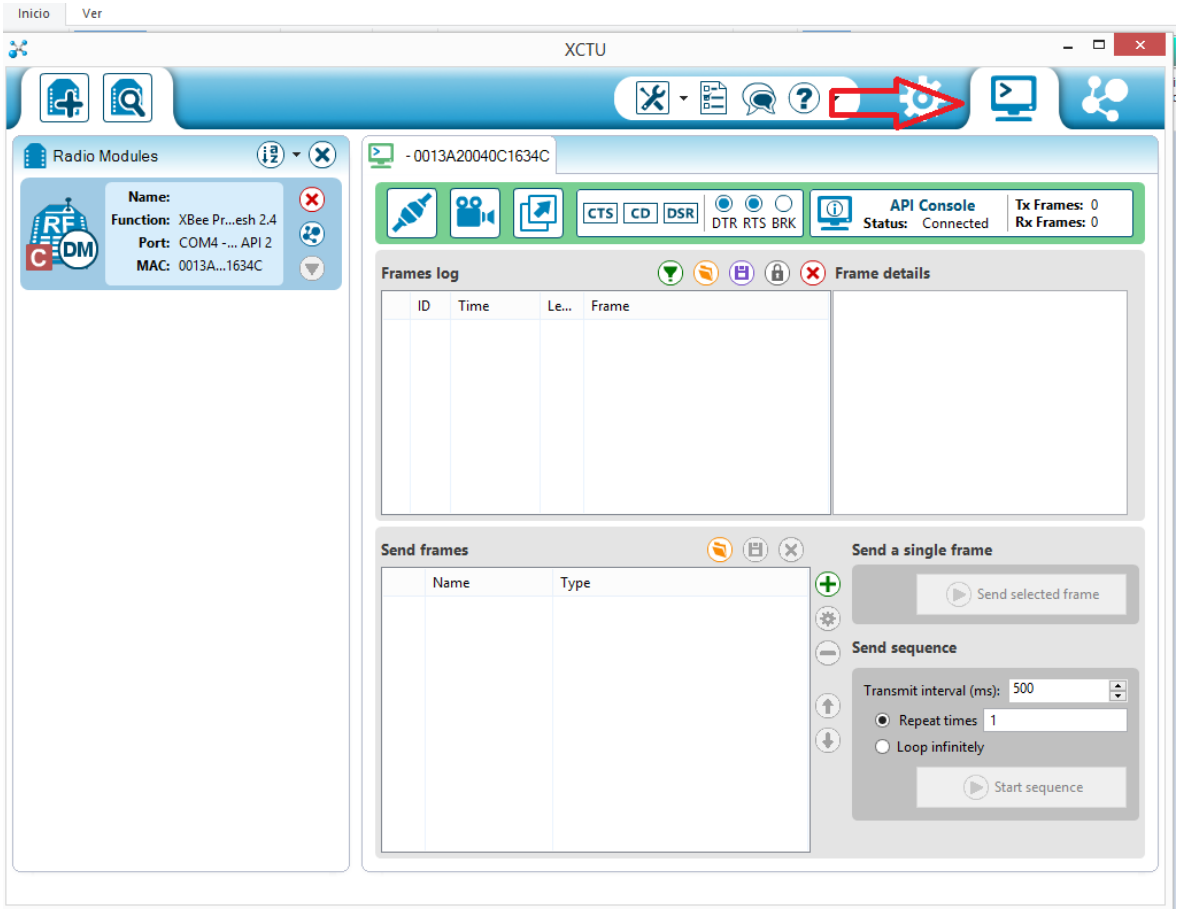


Figura 3.19 Ventana de consola.

3.2.2 802.15.4

Esta tecnología ya fue nombrada e introducida en el capítulo 1 y en este se detallarán la configuración de tipología e instalación utilizada.

Para hacer uso de esta tecnología se utilizarán módulos de transmisión de la marca Digi. Se trata del modelo XBee series 1 con una frecuencia de transmisión de 2.4 GHZ, mostrado en la figura 3.20. Cada uno de estos módulos cuenta con una dirección física única que servirá para identificarlos y para poder establecer las comunicaciones necesarias.

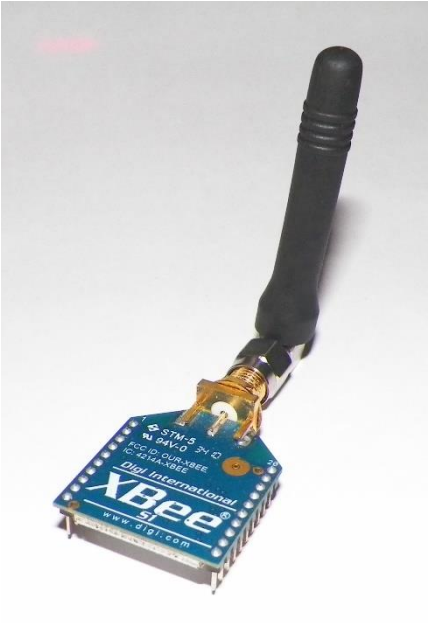


Figura 3.20 Módulo de comunicación XBee s1.

La red estará formada por sensores Waspmote con los citados módulos de transmisión que mandarán la información de la plaza de aparcamiento a un router Meshlium. Se utilizará la ya nombrada topología en estrella en el apartado 1.2.1, así que se tendrá una comunicación unicast entre los sensores y el router. Para que las comunicaciones sean satisfactorias se deberá elegir el mismo canal de transmisión y la misma id de la red para todos los módulos de comunicación. En este caso se ha elegido el canal C y el identificador de red 3332, en los parámetros CH e ID respectivamente, en el programa de configuración anteriormente citado.

A la hora de situar el router Meshlium se ha de tener en cuenta a la distancia máxima que pueden trasmitir estos módulos que según el fabricante es 90 metros para comunicaciones punto a punto. Así que en este caso el sensor más alejado no podrá a estar a más de 90 metros del router. En el caso de que se necesite que algún sensor este más alejado, se pude configurar un sensor intermedio que siempre este activo como router para reenvío de paquetes desde los nodos más alejados hasta el Meshlium. Se debe intentar situar el router en la posición más centrada que sea posible.

La potencia de transmisión también se pude regular, ya que tiene cinco niveles, siendo la distancia anterior la máxima distancia a la potencia de transmisión. Si se tiene nodos muy cercanos al router se podrá bajar la potencia con el objetivo de ahorrar energía.

El consumo de este tipo de módulo a máxima potencia de transmisión es de 45 miliAmperios.

3.2.3 DigiMesh

Al igual que el apartado anterior esta tecnología también fue introducida en el capítulo uno mencionado sus principales características. En este apartado se detallarán la configuración de tipología e instalación utilizada.

Para hacer uso de esta tecnología se utilizarán módulos de transmisión modelo XBee Pro series 1, con una frecuencia de trasmisión de 2.4GHz, mostrado en la figura 3.21. Al igual de los módulos de 802.15.4 estos cuentan con una dirección física para identificarlos y efectuar las comunicaciones.

Figura 3.21 Módulo de comunicación DigiMesh.

La red en este caso estaría formada igual que en apartado anterior pero en este caso la topología seria la mallada citada en el apartado 1.2.2. La comunicación configurada para los nodos sensores será unicast con dirección de destino el router, para este caso se cuenta con un router Meshlium compatible con DigiMesh, en su defecto se utilizara la estación base vista en el apartado anterior, con el programa XCTU un el modo consola para visualizar los datos recibidos. Se configurarán todos los módulos con la id 3332 y el canal C para efectuar las comunicaciones.

En este caso la situación de la estación base no tiene que ser tan especial como con el 802.15.4, ya que se configurarán todos los módulos como “Standar Router” para que puedan facilitar rutas a los nodos que no encuentran el destino y reenviar los datos por ellos para finalizar con éxito la comunicación. Para que el reenvío entre módulos sea

posible solamente es necesario que los nodos intermedios estén en funcionamiento, ya que la funcionalidad que proporciona DigiMesh funciona automáticamente.

Con estos módulos la distancia máxima a la que se puede transmitir, según el fabricante, es 1.6 kilómetros, por lo que en principio debido a su gran alcance no se debería que tener que poner la estación base en un sitio en especial, solo con que estuviera al alcance de un nodo sería suficiente porque todos los demás se podrían comunicar a través de él. En esta topología funcionarán siempre las comunicaciones, a no ser, que algún nodo se quede aislado sin poder comunicarse con ninguno de los demás que forman la red.

En este caso es mucho más interesante el regular la potencia de transmisión, ya que la distancia de alcance es mucho más grande y el consumo aumenta hasta los 250 miliamperios, si las plazas de aparcamiento son colindantes, la red funcionaria correctamente a mínima potencia.

3.2.4 802.15.4 frente a DigiMesh

En este apartado se compararán ambas tecnologías comparando sus características y por lo tanto sus diferencias y se comentará cual es más adecuada para cada situación. Para esto la decisión se basará en el consumo, potencia de transmisión, distancia de transmisión y topología.

En cuanto a la distancia de transmisión, DigiMesh tiene ventaja ya que la distancia máxima es más de 10 veces superior a la ofrecida por el 80.15.4, pero esto hace que la potencia consumida sea también mayor en el caso de DigiMesh, aunque como estamos tratando con un sistema de parking se puede presuponer que no habrá una gran distancia entre nodos y por lo tanto se podrá reducir la potencia de transmisión, por lo que bajara el consumo considerablemente.

Si se comparan las topologías de red, DigiMesh también tendría ventaja ya que no se ha de poner un gran empeño en buscar una colocación óptima para que todos los nodos puedan trasmitir correctamente al receptor, ya que la capacidad de renvío entre nodos permite abarca mucha más zona con un solo router que el 802.15.4, que necesitaría nodos configurados exclusivamente como routers.

Como añadido a la comparación, se puede decir que en cuanto al precio los módulos de XBee pro compatibles con DigiMesh son casi un 70% más caros que los módulos XBee 802.15.4.

Con estos datos se puede decir que sería conveniente utilizar transmisiones 802.15.4 para redes de sensores pequeñas, ya que tiene menos potencia y por lo menos consumo. Para una red de sensores más extensa y con menos estaciones de recepción

de datos disponibles, se debería utilizar DigiMesh, ya que su gran ventaja es el reenvío automático de datos entre nodos junto a su mayor alcance de transmisión.

3.3 Waspmote versión 1.1

En esta sección se expondrán las diferentes configuraciones realizadas para los sensores Waspmote de la empresa Libelium, en este caso la versión 1.1, se muestra en la figura 3.22. Pasando por la instalación del entorno de desarrollo, la inclusión de librerías para el correcto funcionamiento y las medidas tomadas para los diferentes modos de comunicación.

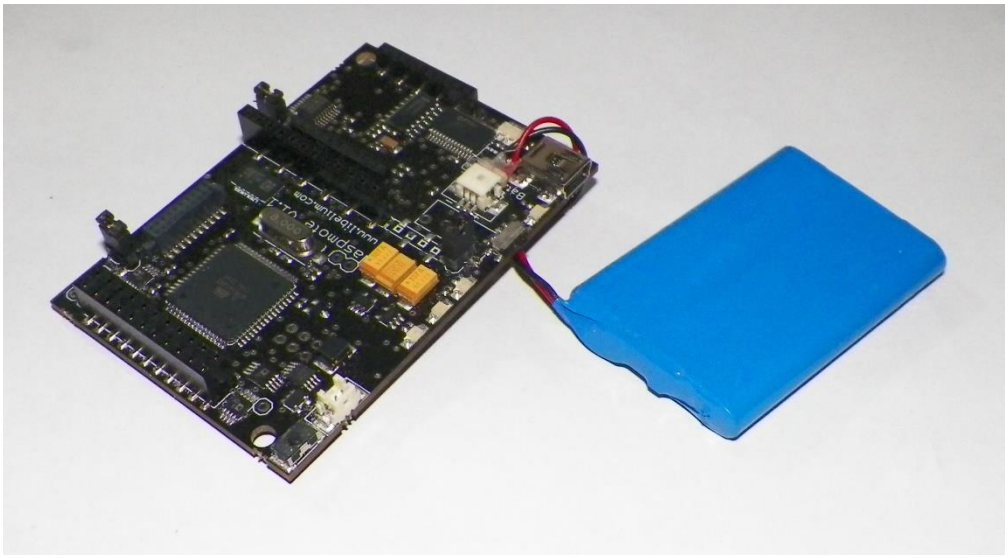


Figura 3.22 Waspmote versión 1.1

3.3.1 Instalación de IDE e inclusión de librerías.

Para poder desarrollar aplicaciones y cargarlos en los Waspmote es necesario descargar el entorno de desarrollo proporcionado por la empresa desarrolladora de los sensores. Este se encuentra en la página web de Libelium, www.libelium.com. Entrando en el apartado “Development”, como se trata de la versión anterior de Waspmote deberemos entrar en el apartado exclusivo para este modelo, como se muestra en la figura 3.23.

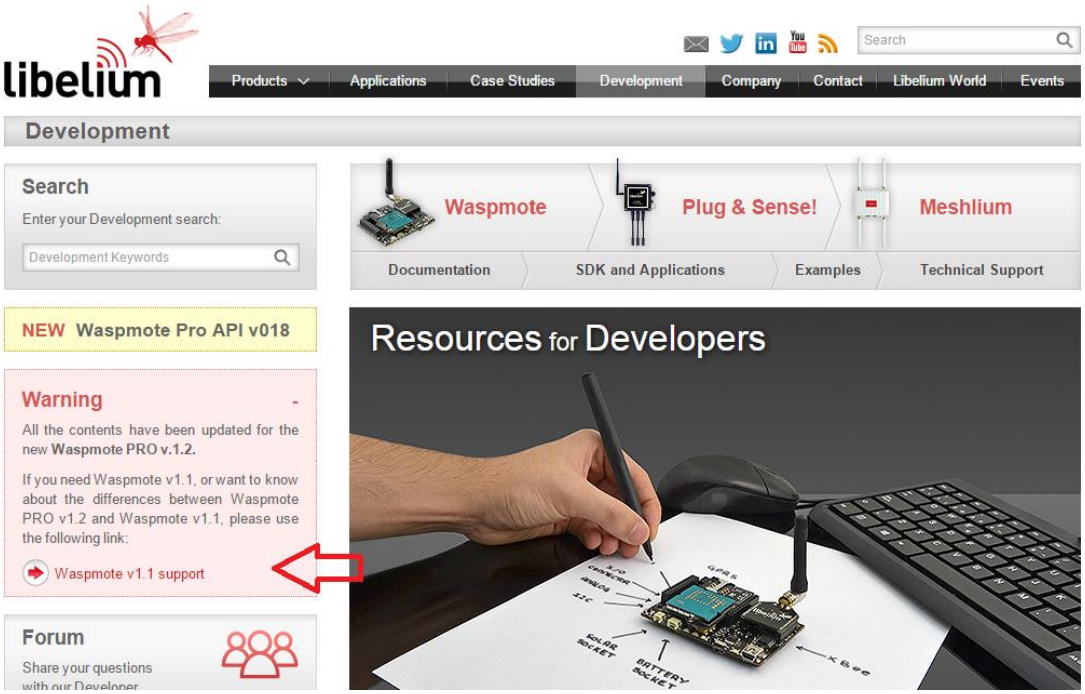


Figura 3.23 apartado development acceso a apartado de versión 1.1

Seguidamente, se pasara a descargar el entorno de desarrollo y la última API proporcionada para esta versión de sensor, se indica donde hay que acceder en la figura 3.23. Se escogerá la versión de Windows 64 bits para el IDE y la API versión 33 que es la última disponible.

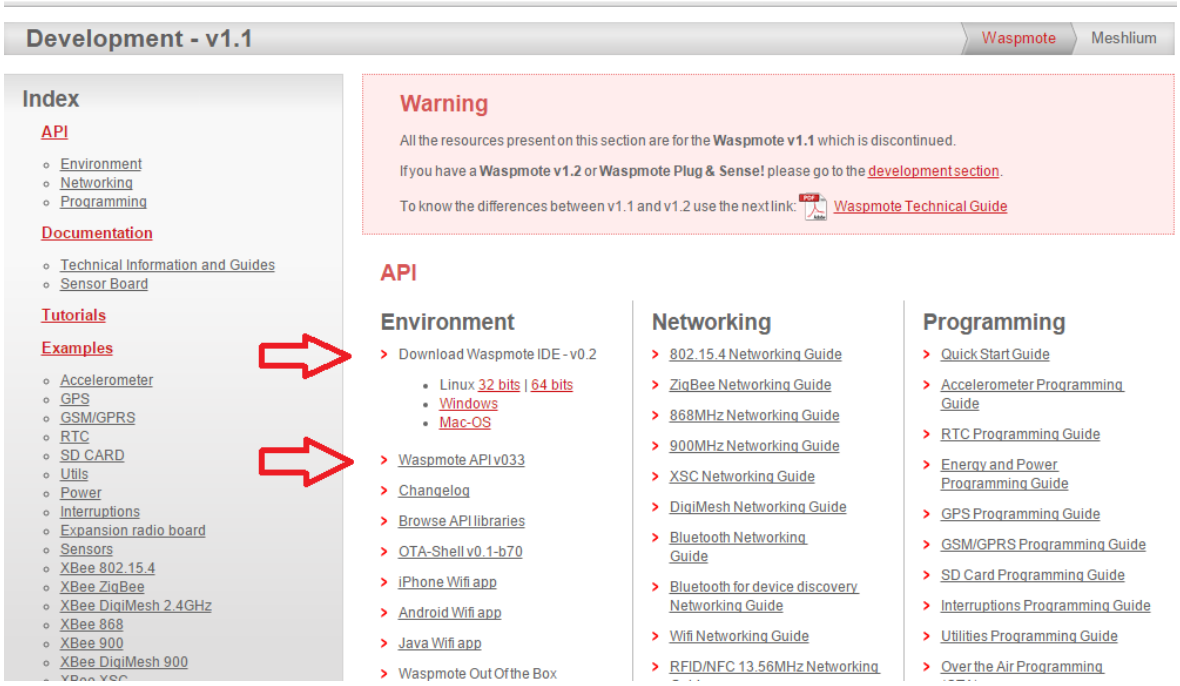


Figura 3.24 Pagina de descarga del IDE y la API.

Después de descargar ambos archivos y descomprimirlos se debe situar ambas carpetas en el mismo directorio para poder relacionar las librerías con el entorno de desarrollo. Se deberá abrir el IDE llamado como Waspmote dentro de la carpeta descargada. Dentro de este se nos dan las opciones de verificar, parar, crear nuevo fichero, abrir, guardar, cargar a la placa y monitorización de puerto serie asociado al Waspmote conectado, como muestra la figura 3.25.

Figura 3.25 Menú de IDE Waspmote 1.1

Para seleccionar la librería que queremos utilizar se tendrá que acceder al apartado Tools y dentro de este seleccionar Board, ahí aparecerá las APIs que este en el mismo directorio que el entorno de desarrollo. Como se muestra en la figura 3.26.

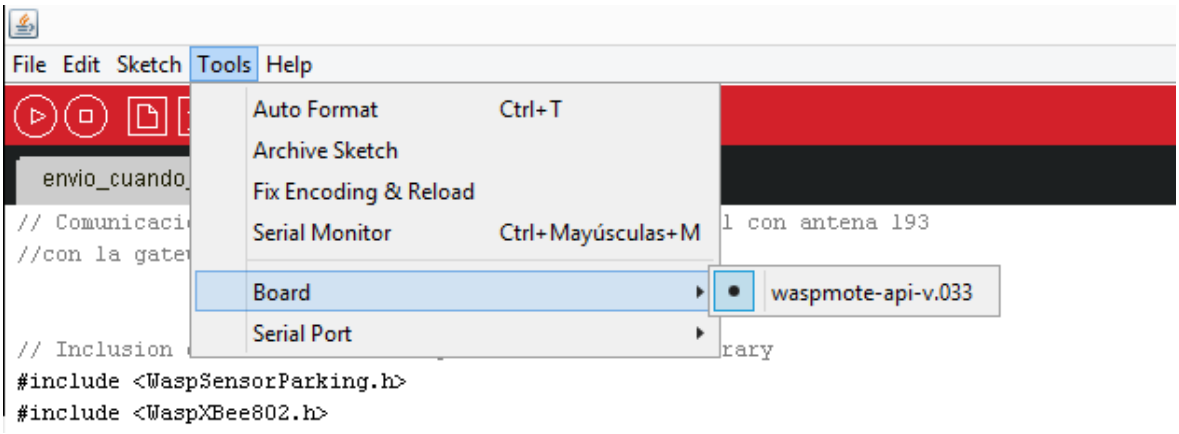


Figura 3.26 Elección de la API.

Para seleccionar el puerto serie al que está conectado el Waspmote al cual se le va a subir la aplicación, se deberá acceder también al menú Tools pero este caso al apartado Serial Port, como se ve en la figura 3.27.

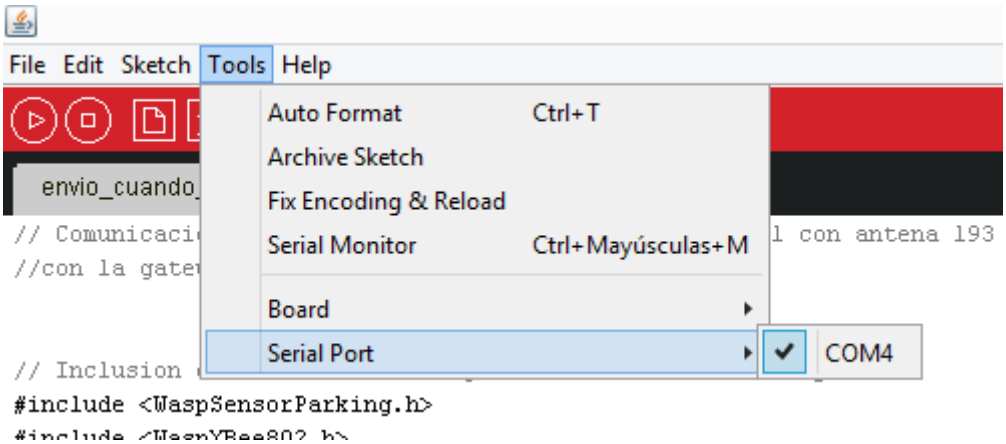


Figura 3.27 Elección del puerto serie.

3.3.2 802.15.4

En esta sección se expondrán todas las configuraciones realizadas para el correcto funcionamiento del Waspmote de la versión 1.1 con la placa de sensor de parking y módulo de comunicaciones 802.15.4.

Se empezara comentando la configuración de los módulos, la cual se hará con el programa XCTU mostrado en la sección 3.2.1. Dentro del programa en el apartado de configuración de radio y se pondrán los siguientes valores:

- CH (Operating Cannel) = C
- ID (Network ID) = 3332

- CE (Routing/Messaging Mode) = 2 (End device)
- BD (Baud Rate) = 5 (38400)
- AP (API Enable) = 2 (API mode with Escapes)

Para este caso de estudio no será necesario alterar ningún parámetro más, aunque en este apartado también se puede cambiar en caso de que sea necesario la potencia de transmisión y la inclusión de una clave de cifrado que en este caso no se ha implementado, pero sería tan sencillo como definir la misma clave para todos los modos de transmisión.

El apartado CE se puede configurar como un router para configurar el sensor para que reenvíe datos, pero en esta configuración no será necesario.

El parámetro de tasa de velocidad en baudios se configura a ese valor porque es el adecuado para que sea legible lo que aparece por el puerto serie para la versión 1.1, en la versión 1.2 será diferente ya que esta tasa ha aumentado.

En cuanto el apartado AP este debe de estar activado con valor 2 para que la comunicación unicast sea posible, si este toma valor 0 o 1 solo se permitirá la comunicación por broadcast con los demás elementos de la red.

Código implementado.

El código implementado para este tipo de comunicación y sensor tratará de lo siguiente. Primero se definirán las variables a usar, en especial la dirección MAC de destino. Durante la fase de inicialización se pondrá en marcha el sensor de parking y se calibrará tomando los parámetros iniciales, seguidamente también se pondrá en funcionamiento el módulo XBee. Seguidamente el sensor entrará en la fase de bucle, en la cual se medirá el estado de la plaza de parking, se comprobará si es igual al anterior estado, si no lo es construirá un mensaje en el cual se envía el identificador del Waspmote y el estado de la plaza (0 si esta libre o 1 si está ocupada), si el estado de la plaza es el mismo que el anterior pasara al siguiente paso directamente. En el siguiente paso se desconecta el sensor durante 5 minutos, pasados estos se activa el sensor y acaba el bucle.

Continuará constantemente con la realización del bucle, es importante que no se mueva de sitio el sensor una vez encendido, porque ya tomo los valores de referencia en la primera ejecución y si se desplaza no funcionará correctamente.

3.3.3 DigiMesh

En este apartado se expondrá las configuraciones realizadas para que funcionen las comunicaciones del Waspmote de la versión 1.1 con la placa de sensor de parking y módulo de comunicaciones DigiMesh.

En la configuración del módulo de comunicación, que también se hará con el programa XCTU nombrado anteriormente se configurarán los siguientes valores:

- CH (Operating Cannel) = C
- ID (Network ID) = 3332
- CE (Routing/Messaging Mode) = 0 (Standar Router)
- BD (Baud Rate) = 5 (38400)
- AP (API Enable) = 2 (API mode with Escapes)
- SM (Sleep Mode) = 4 (Async. Cyclic Sleep)

Los parámetros a configurar y los valores que toman son los mismos que en el apartado anterior 3.3.2, pero esta vez se pone en el parámetro CE el valor 0 ya que en DigiMesh se pueden reenviar los datos automáticamente sin tener que realizar una configuración específica, también se incluye el parámetro SM que corresponde con la capacidad de dormir el sensor y el módulo de comunicaciones que incorpora DigiMesh, con la que se pone el sensor en un estado de absoluto reposo en el cual casi no se consume batería.

En este caso el modo 4 permite que se duerma y despierte el sensor de una forma cíclica pero no regulada, ya que la función cíclica se reserva a la nueva versión de los Waspmote la 1.2. De este modo en el código se definirá el tiempo que permanece el sensor despierto y cuanto durmiendo. Al igual que para los módulos de 802.15.4 el modo AP tiene que ser el 2 para que funcionen las comunicaciones unicast,

Código implementado.

El código implementado tendrá el mismo funcionamiento que el visto en el anterior apartado 3.2.2, pero cambiando la parte en la cual se desactiva el sensor por la nueva funcionalidad de dormir que se realiza mediante la función “PWR.sleep(SENS_OFF);”, pasado el tiempo de dormir el sensor despertara y seguirá con su funcionamiento.

3.4 Waspmote versión 1.2

En esta sección se expondrán las diferentes configuraciones realizadas para los sensores Waspmote de la empresa Libelium, en este caso la versión 1.2, se muestra en la figura 3.28. Pasando por la instalación del entorno de desarrollo, la inclusión de librerías para el correcto funcionamiento y las medidas tomadas para los diferentes modos de comunicación.

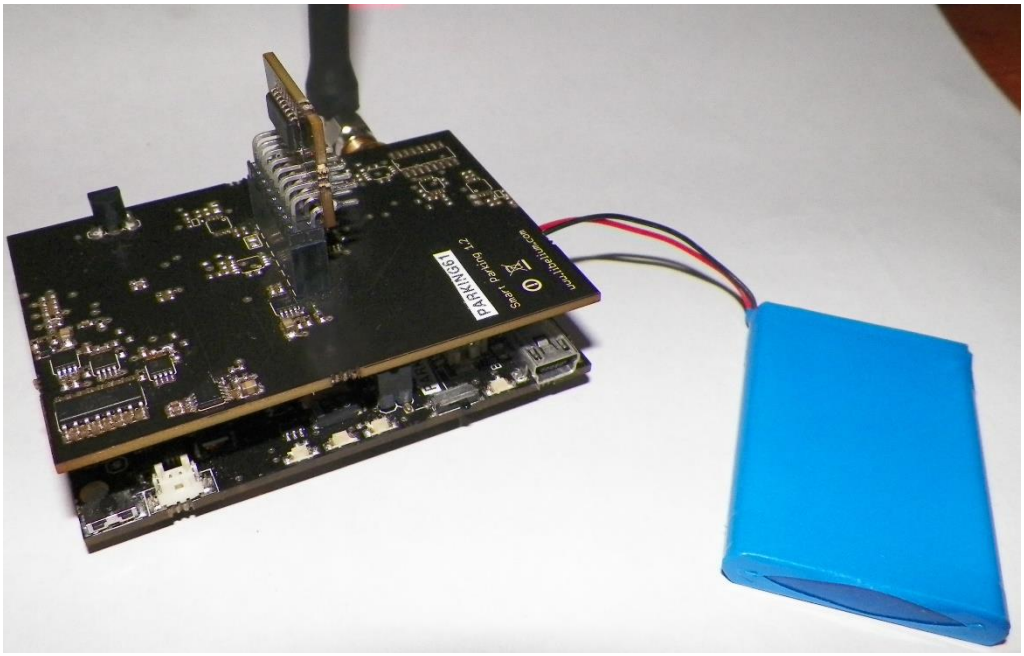


Figura 3.28 Waspmote 1.2 con placa de aparcamiento.

3.4.1 Instalación de IDE e inclusión de librerías.

La instalación del entorno de desarrollo y las librerías se harán de la misma forma que para la versión 1.1 mostrado en el apartado 3.3.1, pero con pequeñas diferencias siendo el IDE mejorado respecto al anterior. Este también es gratuito y de libre uso y se puede descargar al igual que el anterior de la página de la empresa Libelium, www.libelium.com, pero esta vez accediendo al apartado “Development” y esta vez pulsar sobre el apartado donde aparece la Waspmote Pro API, como se ve en la figura 3.29.

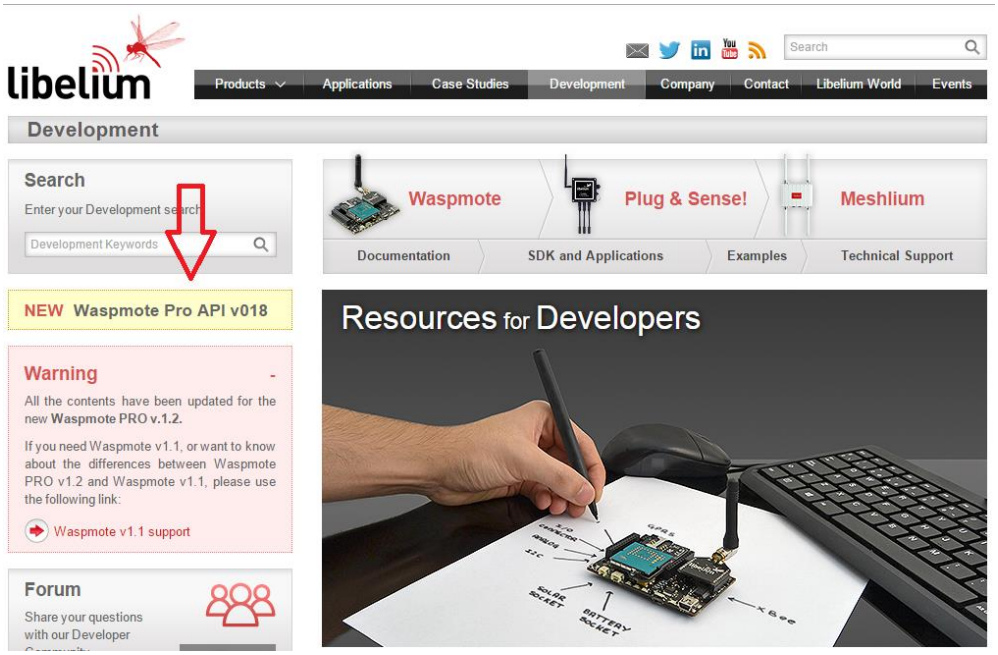


Figura 3.29 Apartado Development de la web de Libelium.

A continuación aparecerán los ficheros disponibles para descargar, y se tendrá que descargar el Pro IDE y la última API que se encuentre, como se ve en la figura 3.30, para tener todas las funciones actualizadas.

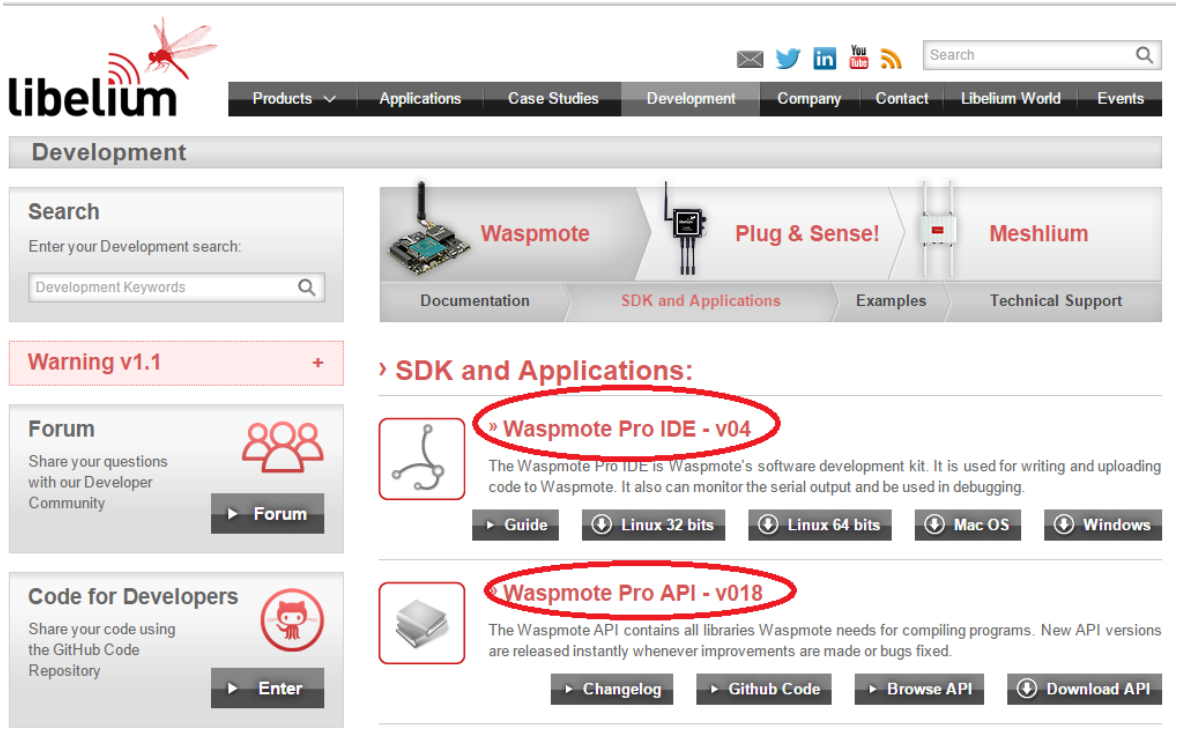


Figura 3.30 Descargas desde la web de Libelium.

Después de descargar ambos archivos y descomprimirlos se debe situar ambas carpetas en el mismo directorio para poder relacionar las librerías con el entorno de desarrollo. Una vez abierto el entorno de desarrollo nos encontraremos con las siguientes funcionalidades: verificar, cargar, nuevo, abrir, guardar y monitor del puerto serie., como se puede ver en la figura 3.31.

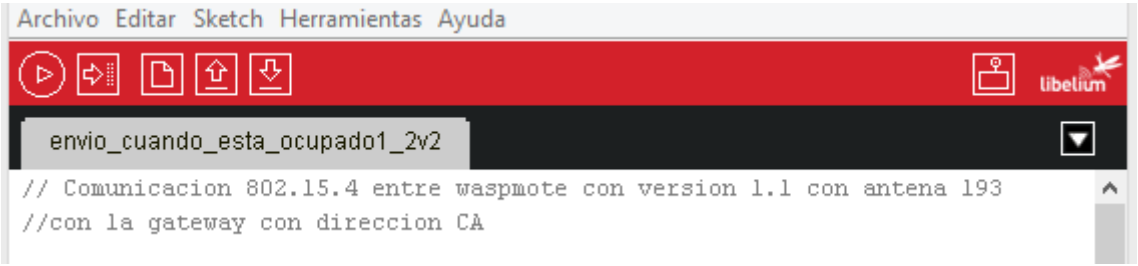


Figura 3.31 Menú pro IDE para Waspmote 1.2.

Para seleccionar la librería que queremos utilizar se tendrá que acceder al apartado herramientas y dentro de este seleccionar tarjeta, ahí aparecerá las APIs que este en el mismo directorio que el entorno de desarrollo. Como se muestra en la figura 3.32.

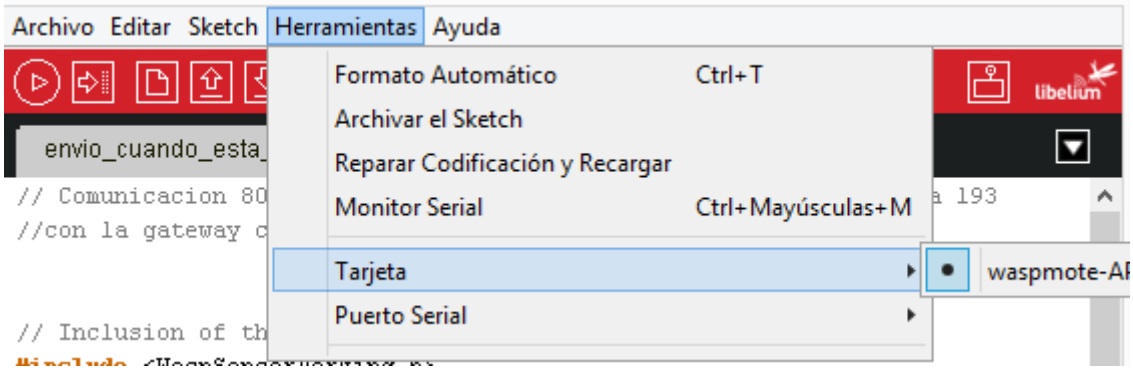


Figura 3.32 Selección de tarjeta.

Para seleccionar el puerto serie al que está conectado el Waspmote al cual se le va a subir la aplicación se deberá acceder también al menú herramientas pero este caso al apartado puerto serie, como se ve en la figura 3.33.

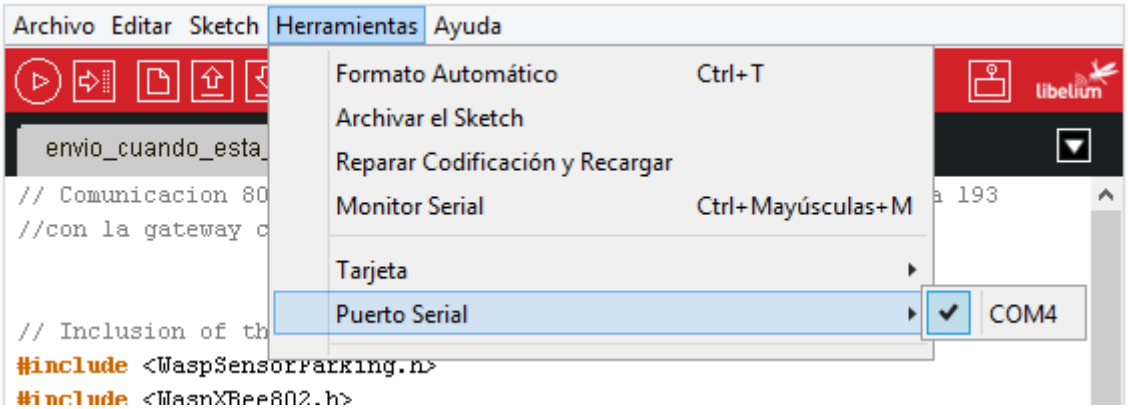


Figura 3.33 Selección del puerto serie.

3.4.2 802.15.4

En esta sección se expondrán todas las configuraciones realizadas para el correcto funcionamiento del Waspmote de la versión 1.2 con la placa de sensor de parking y módulo de comunicaciones 802.15.4.

La configuración de las antenas y el funcionamiento del código será el mismo que el mostrado en el apartado 3.3.2.

Código implementado.

La funcionalidad el código será la misa pero en este caso se utilizarán las funciones pertenecientes a la nueva API y la librería “WaspFrame.h”, en la que la generación de la trama a enviar se hace de forma automática y no manual como en el código para el Waspmote 1.1, esta librería no se incluía en las APIs anteriores. Funciona de manera de que se crea una trama con la función “frame.createFrame (ASCII);” y se añaden tantos

sensores junto con su valor, como se crea oportuno. Estas funciones generan automáticamente la trama de datos a enviar y mediante otra función el router Meshlium es capaz de almacenarla en la base de datos automáticamente.

A continuación se detallará los pasos a realizar para construir la trama a enviar. Entre paréntesis aparecerá el código utilizado para realizar los pasos indicados. En primer lugar y tras incluir la librería citada, se ha de definir el identificador que se le ha asociado al sensor (`char WASPMOTE_ID[] = "Parking_1.2";`). Seguidamente se ha de asociar el identificador definido a las tramas a enviar (`frame.setID( WASPMOTE_ID );`), con esta función se asocia el valor Parking_1.2 como identificador de sensor a todas las tramas que se envíen. Después, en el caso de que exista envío, se definirá la trama (`frame.createFrame(ASCII);`), en este caso se envía una trama ASCII, aunque también se puede crear una trama binaria para realizar el envío. Lo siguiente sería añadir los sensores que se quieren enviar en este caso el de parking junto con su estado (`frame.addSensor(SENSOR_PS,status);`), donde status es el valor que devuelve la función de leer el sensor de parking y PS es el nombre asociado al sensor, este viene definido por Libelium para que pueda ser interpretado correctamente por la función existente en el router Meshlium. Después se pararía al envío de la trama.

3.4.3 DigiMesh

En este apartado se expondrá las configuraciones realizadas para que funcionen las comunicaciones del Waspmote de la versión 1.2 con la placa de sensor de parking y módulo de comunicaciones DigiMesh.

En la configuración del módulo de comunicación, que también se hará con el programa XCTU nombrado anteriormente se configuraran los siguientes valores:

- CH (Operating Cannel) = C
- ID (Network ID) = 3332
- CE (Routing/Messaging Mode) = 0 (Standar Router)
- BD (Baud Rate) = 5 (38400)
- AP (API Enable) = 2 (API mode with Escapes)
- SM (Sleep Mode) = 8 (Synchronized Cyclic Sleep)

Los parámetros a configurar y los valores que toman son los mismos que en el apartado anterior 3.3.3, pero esta vez el SM estará configurado con valor 8 de sueño cíclico sincronizado.

El parámetro SM define el modo de sueño y puede tomar valores de 0 a 8:

- 0 Normal: no está permitida la función de sueño.

- 1 Asynchronous Pin Sleep: permite que el módulo duerma y despierte en función del estado de la entrada del Pin 9.
- 2, 3 y 6: modos no definidos.
- 4 Asynchronous Cyclic Sleep: el sueño es cíclico y sigue los parámetros definidos en el módulo.
- 5 Asynchronous Cyclic Sleep Pin Awake: igual que el modo 4, pero se despierta si recibe señal por el pin 9.
- 7 Synchronous Sleep Support Mode: el nodo se sincroniza con una red que duerme, pero este no lo hace, es útil para ser coordinador.
- 8 Synchronized Cyclic Sleep: modo de sueño cíclico sincronizado con coordinador de sueño.

3.4.3.1 Synchronized Cyclic Sleep.

En este apartado se expondrá como se ha de configurar los módulos DigiMesh para poder poner en marcha este tipo de sueño. Para empezar estamos ante un modo de sueño cíclico sincronizado.

Para poder poner en marcha este modo se ha de configurar un módulo DigiMesh como coordinador de sueño y los demás en modo seguidor, esto se hace cambiando el valor del parámetro SO (Sleep Options), mediante el programa XCTU, poniendo el valor 0 para indicar que el coordinador de sueño predefinido y 1 para que ese nodo nunca se convierta en coordinador.

Después se habrá de introducir una serie de funciones en los códigos, en el coordinador en la parte de set up se ha de definir el tiempo que se está despierto y durmiendo tanto el mismo sensor como todos los nodos que coordina. En el apartado de set up además de su código de medida de plaza de parking y el envío, se introducirán los las funciones "xbeeDM.setAwakeTime(awake)" y "xbeeDM.setSleepTime(asleep);" que servirán para establecer tanto el tiempo de dormir como el de sueño y lo retransmitirá todos los nodos no coordinadores de la red. Junto a esto, tanto el coordinador como todos los nodos deben de tener el código mostrado en la figura 3.34 para poder interrumpir el sueño.

```
if( intFlag & XBEE_INT )
{
  Utils.setLED(LED1,LED_ON);
  delay(1000);
  Utils.setLED(LED1 ,LED_OFF);

  USB.println(F("--------------------"));
  USB.println(F("XBee interruption captured!!"));
  USB.println(F("--------------------"));
  intFlag &= ~(XBEE_INT); // Clear flag
}
```

Figura 3.34 Código de detección de interrupciones de sueño.

El tiempo para estar durmiendo debe de ser el máximo que tarde el código de los Waspmote en ejecutarse incluido el envío.

De esta forma todos los sensores de la red despiertan y están activos al mismo tiempo por si necesitan realizar reenvíos dentro de la red.

El funcionamiento de esta implementación se puede explicar a través del esquema mostrado en la figura 3.35. Existen dos tipos de nodos en esta configuración: el coordinador y los seguidores. El coordinador es el encargado de definir los tiempos de sueño y de estar despierto y se los ha de comunicar a todos los nodos seguidores. Además en cada ciclo de sueño ha de mandar una señal, para que todos los demás nodos despierten, hagan sus funciones y vuelvan a dormir. De esta forma los nodos seguidores ejecutarán su código, dormirán y volverán a despertarse solo cuando el coordinador lo indique a través de una interrupción una vez pasado el tiempo de sueño.

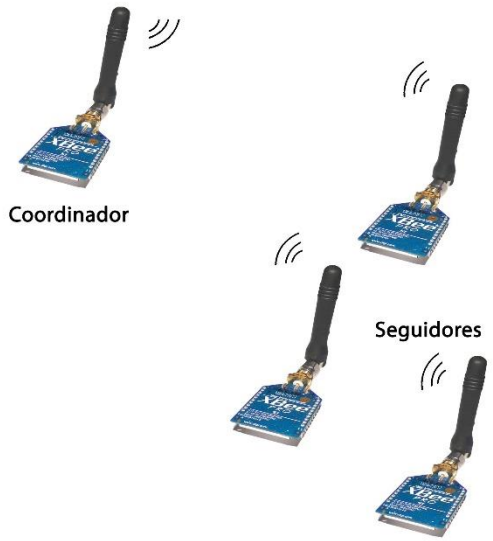


Figura 3.35 Esquema nodos sueño cíclico.

Código implementado.

El código implementado tendrá el mismo funcionamiento que el visto en el anterior apartado 3.3.2, pero aplicando los cambios nombrados en el apartado 3.4.3.1.

3.5 Meshlium

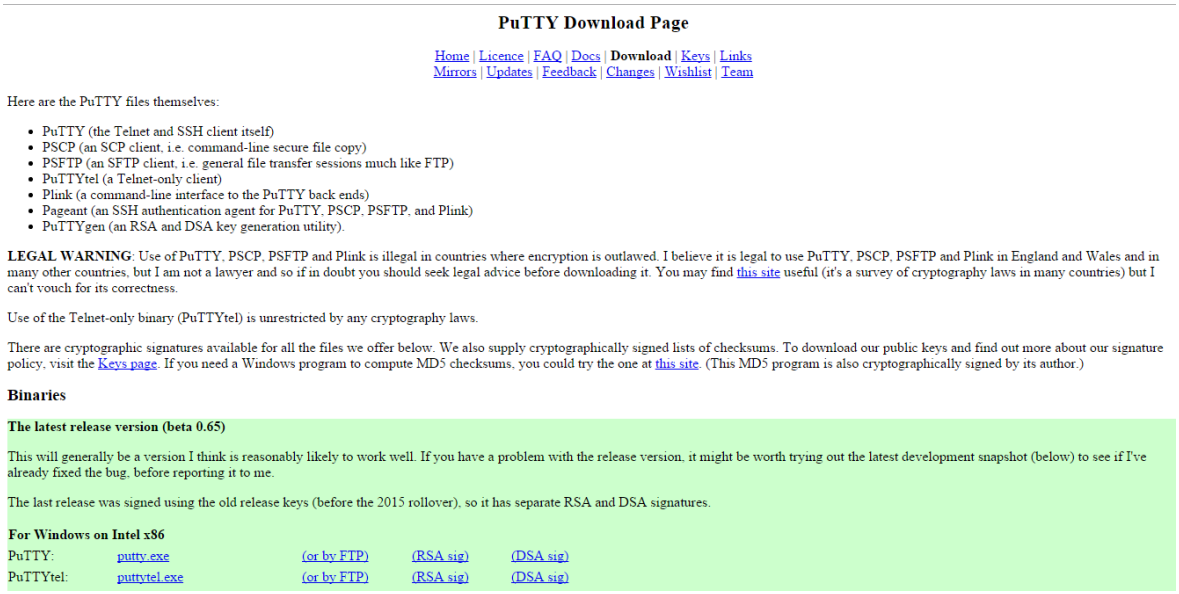
En esta sección se expondrá las configuraciones realizadas en el router Meshlium de la empresa Libelium, pasando por el modo de acceso, control, recepción de datos y la aplicación web implementada para el control de las plazas de aparcamiento.

3.5.1 Acceso

En este apartado se comentará los pasos que hay que seguir para poder acceder al Meshlium y hacer modificaciones dentro de su almacenamiento. Para acceder al almacenamiento se deberá de hacer mediante un cliente SFTP, pero este solo nos dará acceso a lectura. Para poder modificar datos tendremos que realizar una conexión SSH para poder cambiar el estado del Meshlium, a lectura escritura, para así poder subir la aplicación web al sistema de almacenamiento y que funcione como servidor.

Instalación y conexión mediante cliente SSH.

Para establecer la conexión SSH vamos a instalar el cliente Putty, que es uno de los clientes más conocidos y fácil de utilizar para realizar conexiones seguras. Se trata de un Software libre por lo que se puede descargar directamente de la página oficial http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html mostrada en la figura 3.36.



PuTTY Download Page

Home | Licence | FAQ | Docs | **Download** | Keys | Links
Mirrors | Updates | Feedback | Changes | Wishlist | Team

Here are the PuTTY files themselves:

- PuTTY (the Telnet and SSH client itself)
- PSCP (an SCP client, i.e. command-line secure file copy)
- PSFTP (an SFTP client, i.e. general file transfer sessions much like FTP)
- PuTTYtel (a Telnet-only client)
- Plink (a command-line interface to the PuTTY back ends)
- Pageant (an SSH authentication agent for PuTTY, PSCP, PSFTP, and Plink)
- PuTTYgen (an RSA and DSA key generation utility).

LEGAL WARNING: Use of PuTTY, PSCP, PSFTP and Plink is illegal in countries where encryption is outlawed. I believe it is legal to use PuTTY, PSCP, PSFTP and Plink in England and Wales and in many other countries, but I am not a lawyer and so if in doubt you should seek legal advice before downloading it. You may find this site useful (it's a survey of cryptography laws in many countries) but I can't vouch for its correctness.

Use of the Telnet-only binary (PuTTYtel) is unrestricted by any cryptography laws.

There are cryptographic signatures available for all the files we offer below. We also supply cryptographically signed lists of checksums. To download our public keys and find out more about our signature policy, visit the Keys page. If you need a Windows program to compute MD5 checksums, you could try the one at this site. (This MD5 program is also cryptographically signed by its author.)

Binaries

The latest release version (beta 0.65)

This will generally be a version I think is reasonably likely to work well. If you have a problem with the release version, it might be worth trying out the latest development snapshot (below) to see if I've already fixed the bug, before reporting it to me.

The last release was signed using the old release keys (before the 2015 rollover), so it has separate RSA and DSA signatures.

For Windows on Intel x86

PuTTY:	putty.exe	(or by FTP)	(RSA sig)	(DSA sig)
PuTTYtel:	puttytel.exe	(or by FTP)	(RSA sig)	(DSA sig)

Figura 3.36 Pagina de descarga de Putty.

En este caso se descargará el ejecutable para Windows. Una vez descargado habrá que ejecutarlo y nos saldrá la pantalla de inicio, mostrada en la figura 3.37.

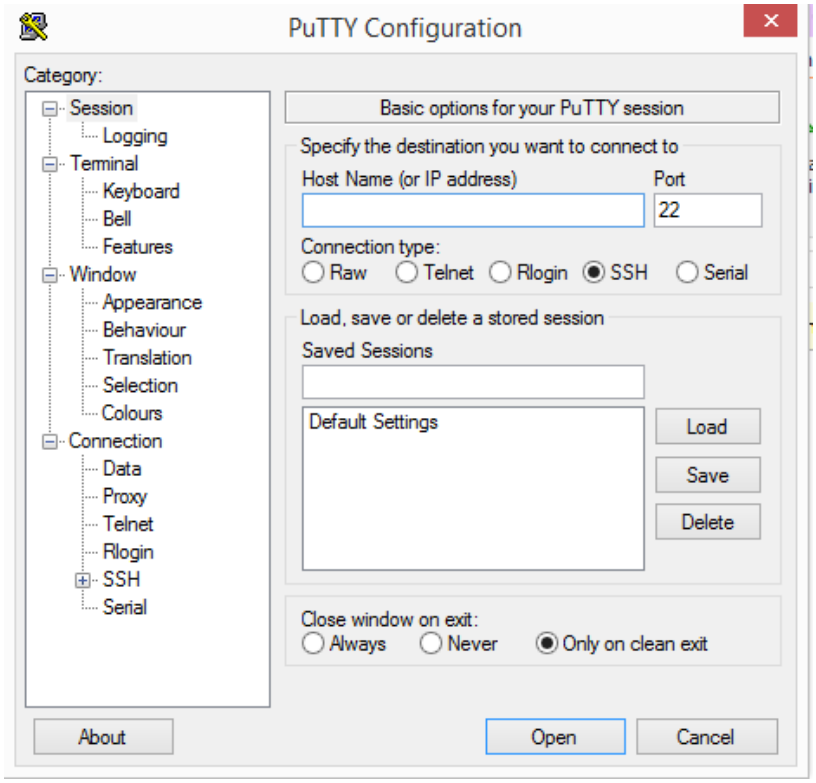


Figura 3.37 Pantalla de inicio Putty.

En esta pantalla de ha de introducir la dirección IP que tenga el Meshlium y seleccionar el tipo de conexión SSH.

Seguidamente se entrará en una consola de comandos en la que se pedirá usuario y contraseña para poder acceder, el usuario a introducir será “root” y contraseña “libelium”. Después, se podrá introducir el comando que permite la lectura y escritura. Este proceso se puede ver en la figura 3.38.

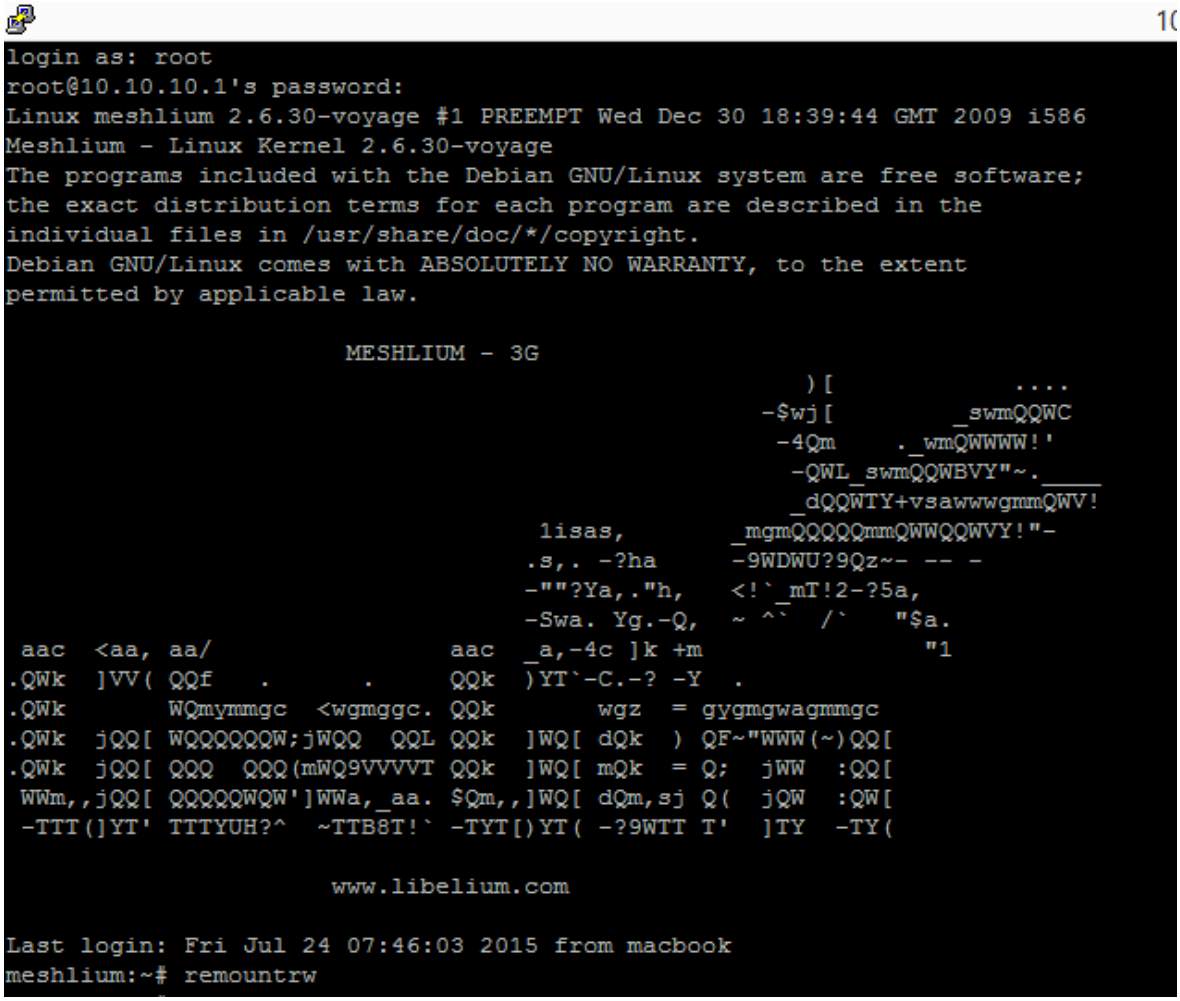


Figura 3.38 Estado de conexión SSH.

Instalación y conexión mediante cliente SFTP.

El acceso al almacenamiento interno del Meshlium debe de ser a través de un cliente SFTP ya que es el tipo de servidor que tiene instalado el router. Se utilizará el cliente Filezilla que es de software libre y se puede descargar directamente desde la web de los

desarrolladores, www.filezilla-project.org/. Se descargará la versión del cliente. La página se muestra en la figura 3.39

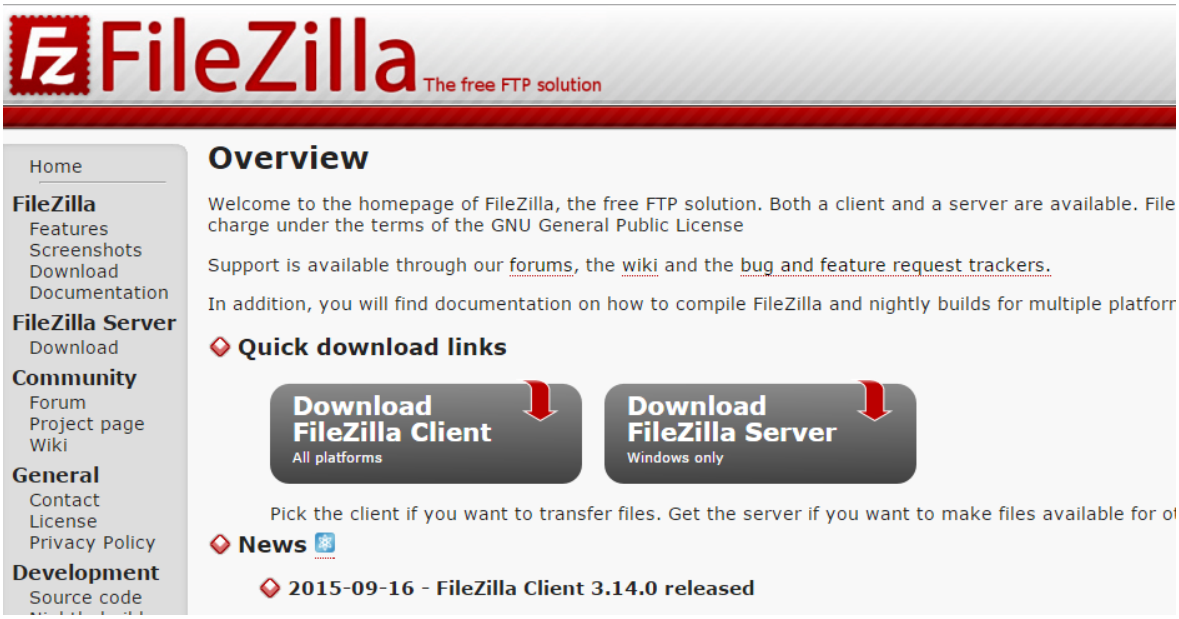


Figura 3.39 Pagina de descarga Filezilla Client

Esta página mandará a otra en la que se elige la plataforma en la que se va a instalar, en este caso Windows de 64 bits. Se ejecutará el instalador y en el asistente se pide que se acepten los términos de uso para continuar, como se muestra en la figura 3.40.

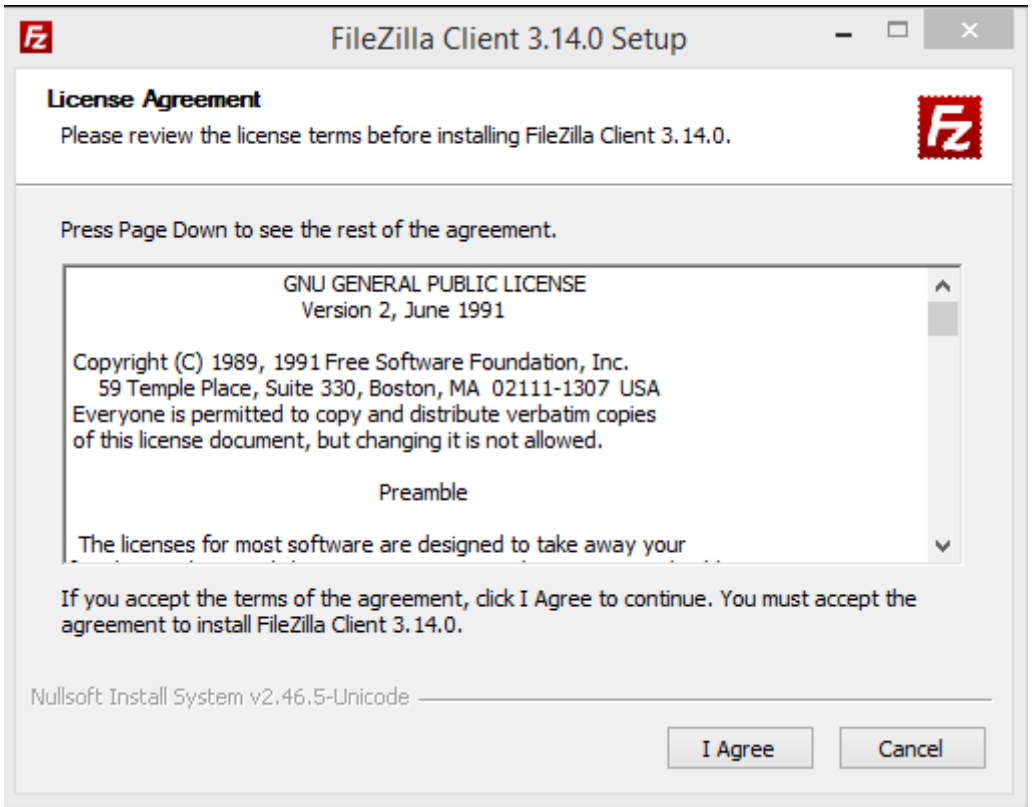


Figura 3.40 Asistente de instalación.

Seguidamente el asistente nos pedirá datos como, la ubicación de la carpeta de instalación, si solo queremos instalar el cliente o si el programa es para todos los usuarios del equipo. Una vez introducidos estos datos dará comienzo la instalación, como se ve en la figura 3.41.

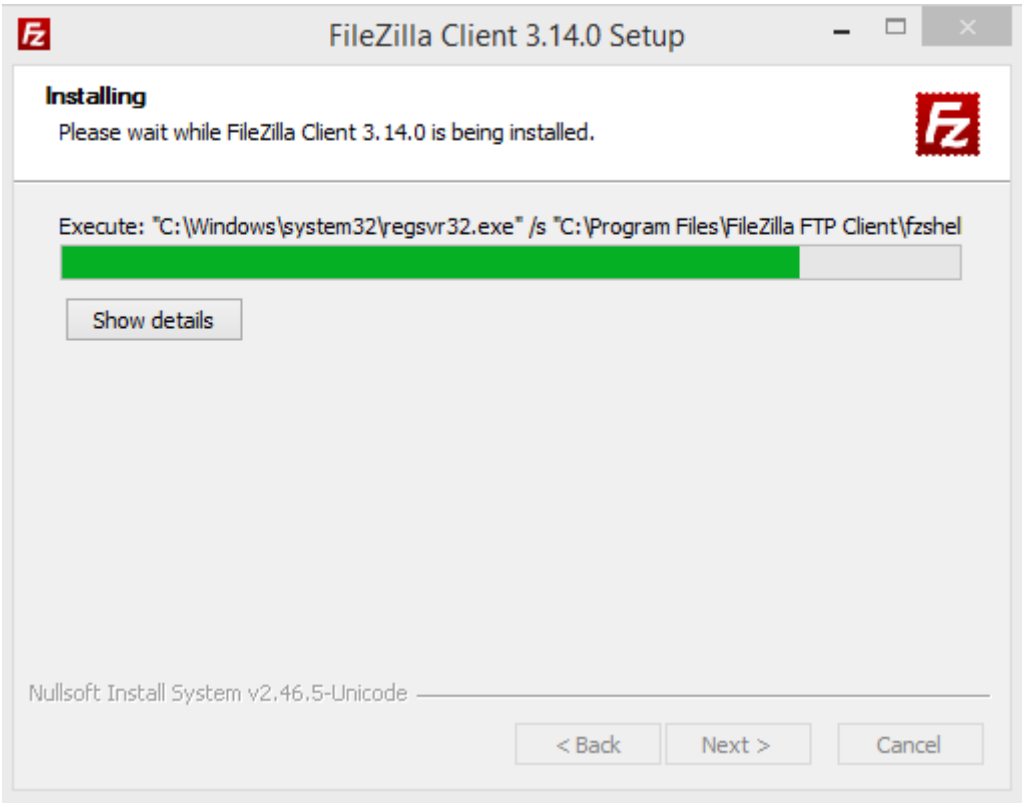


Figura 3.41 Instalación del cliente SFTP.

Una vez instalado se accederá al programa, mostrado en la figura 3.42, y se introducirán los siguientes datos de acceso:

- Servidor: 10.10.10.1
- Nombre de usuario: root
- Contraseña: Libelium
- Puerto: 22

Y se establecerá la conexión con el servidor SFTP.

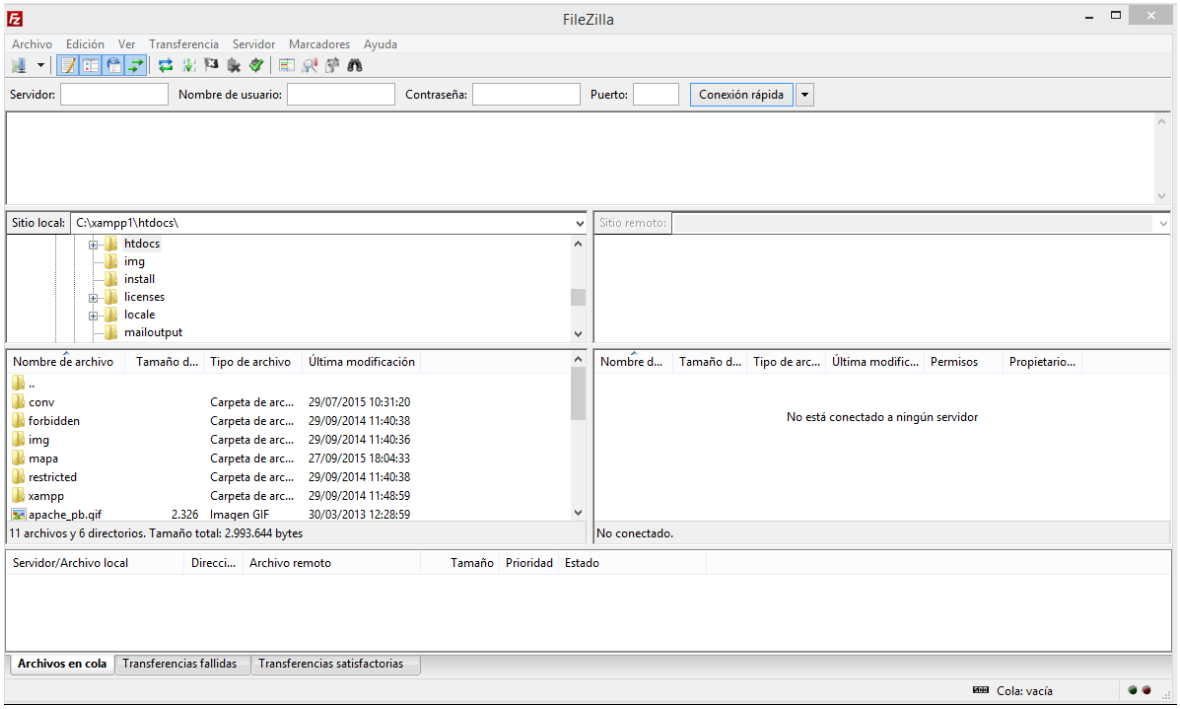


Figura 3.42 Cliente SFTP.

3.5.2 Manager System.

Este es el apartado en el cual se maneja toda la configuración del router Meshlium, desde aquí se puede configurar todas las configuraciones y permite acceder a la base de datos, ya que se encuentra el PHPmyadmin instalado.

Nada más acceder al Manager System estando conectado a través de la red wifi creada por el router, a través de la dirección “10.10.10.1/ManagerSystem“, aparecerá una pantalla de login en la que habrá que introducir el usuario “root” y contraseña “Libelium”, se puede ver en la figura 3.43, seguidamente se accederá a la página principal del Manager System, mostrada en la figura 3.44.

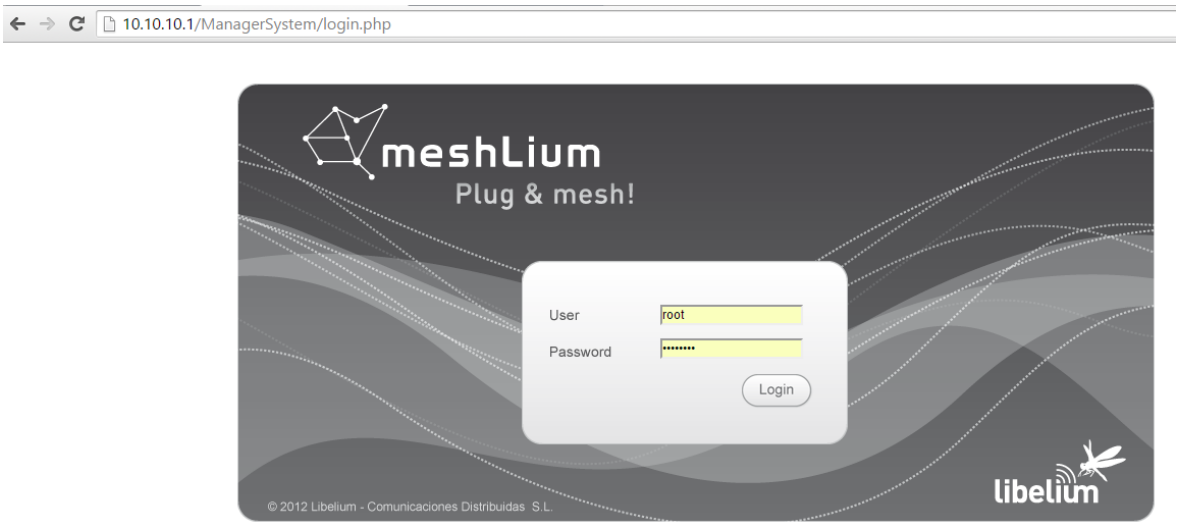


Figura 3.43 Pagina de login System Manager.

Figura 3.44 Página principal System Manager.

Dentro del Manager System encontramos los apartados de interfaces, sensor networks, cloud connector, tools, System, update manager y help. Para la configuración de 805.15.4 que es lo que se necesitará para este proyecto, se ha de entrar en el apartado Sensor network, se muestra en la figura 3.45.

Figura 3.45 Apartado Sensor Networks.

En la sección 802.15.4, que se muestra en la figura 3.46, se podrán configurar los parámetros correspondiente el módulo de comunicación indicado, en él se configurará el canal C y la id 3332.

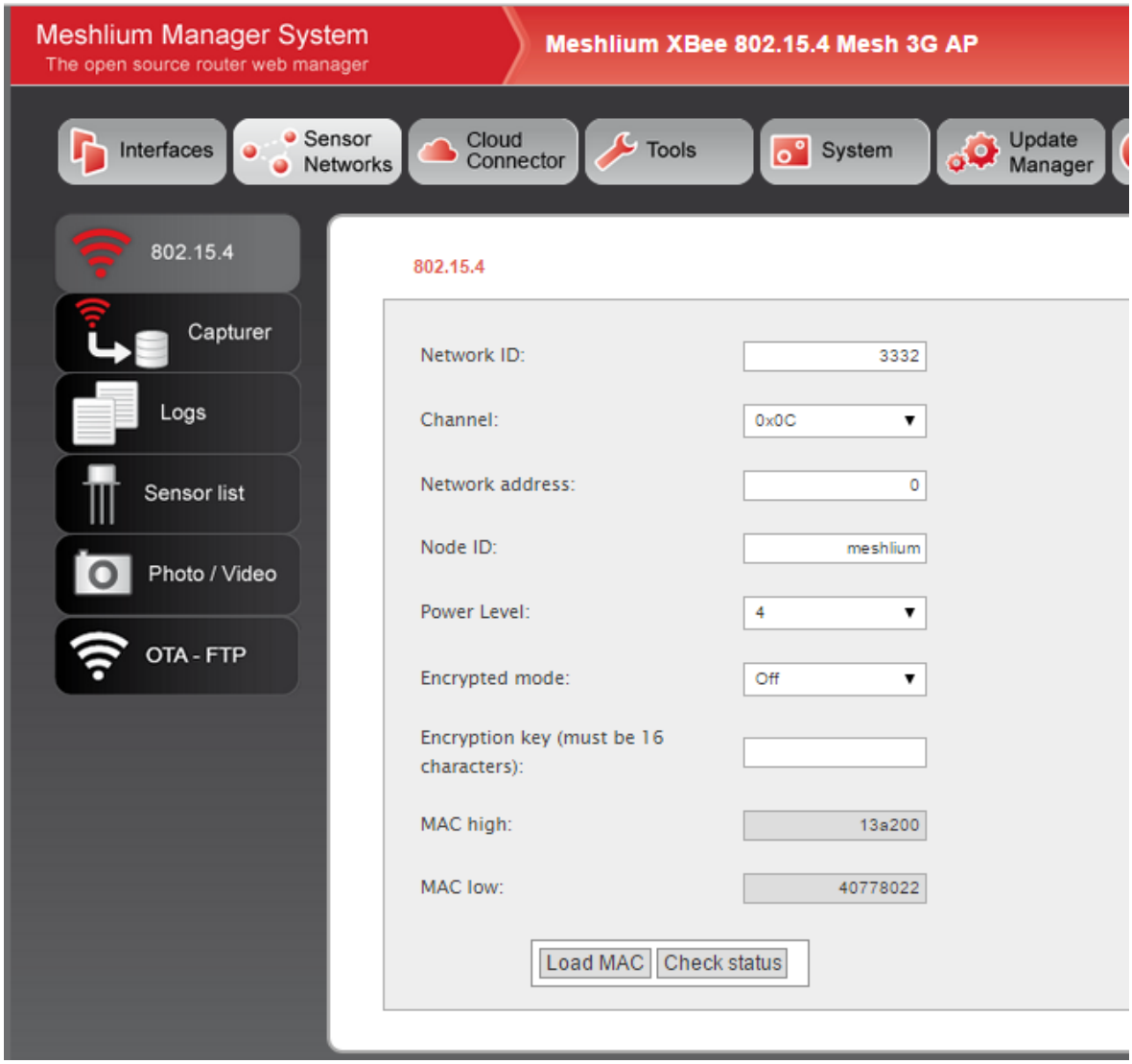


Figura 3.46 Parámetros de configuración 15.4.

En el apartado Logs se almacenan todos los datos que se reciben en la interfaz 802.15.4, se muestra en la figura 3.47. Y en la sección Capture, que se puede ver en la Figura 3.48, se muestran las tramas procesadas por la función Sensor Parser, que es la función que idéntica todas las tramas que llegan al router Meshlium con el formato de trama correspondiente a la librería Waspframe mencionada en el apartado 3.4.2.

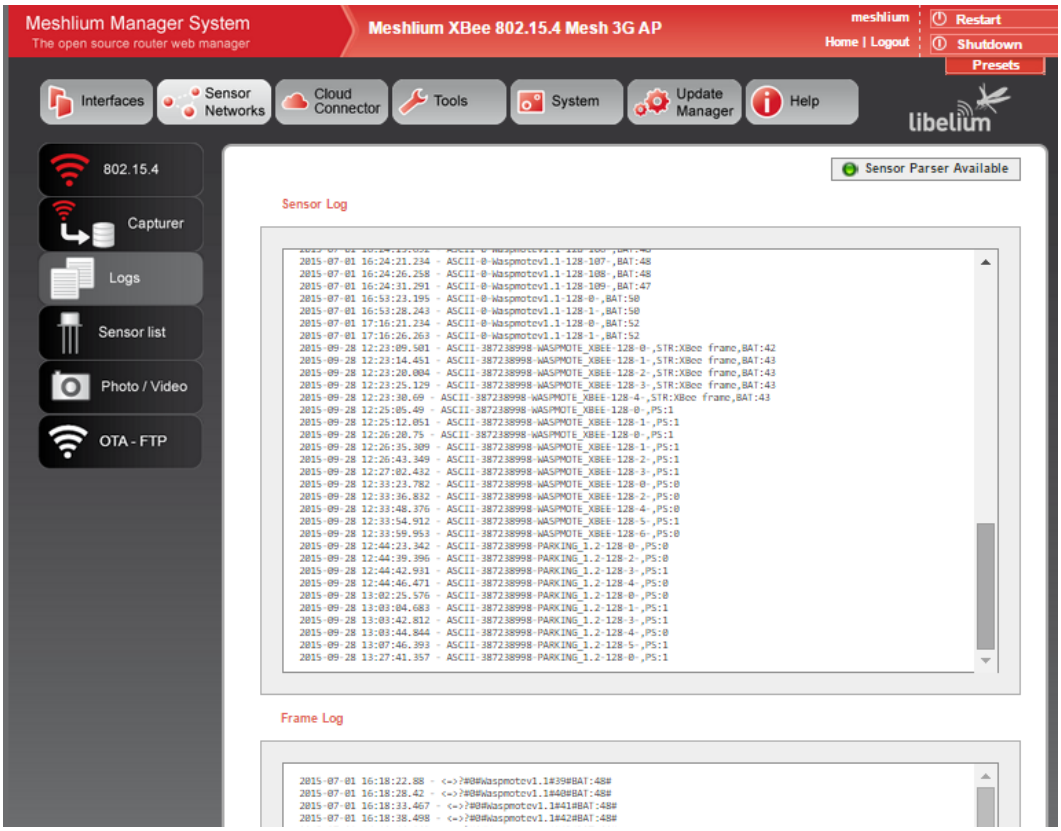


Figura 3.47 Apartado Logs

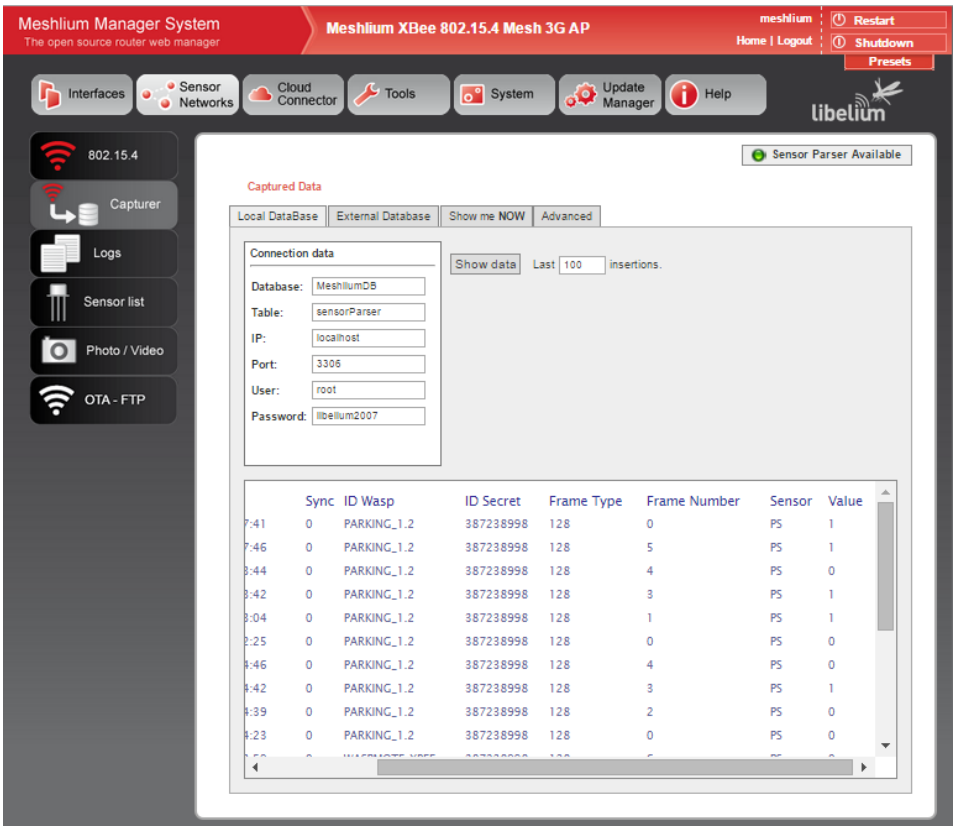


Figura 3.48 Apartado Capturer.

A la hora de acceder a la base de datos se hará desde el apartado tools, como se muestra en la figura 3.49, pulsando en el apartado phpmyadmin.

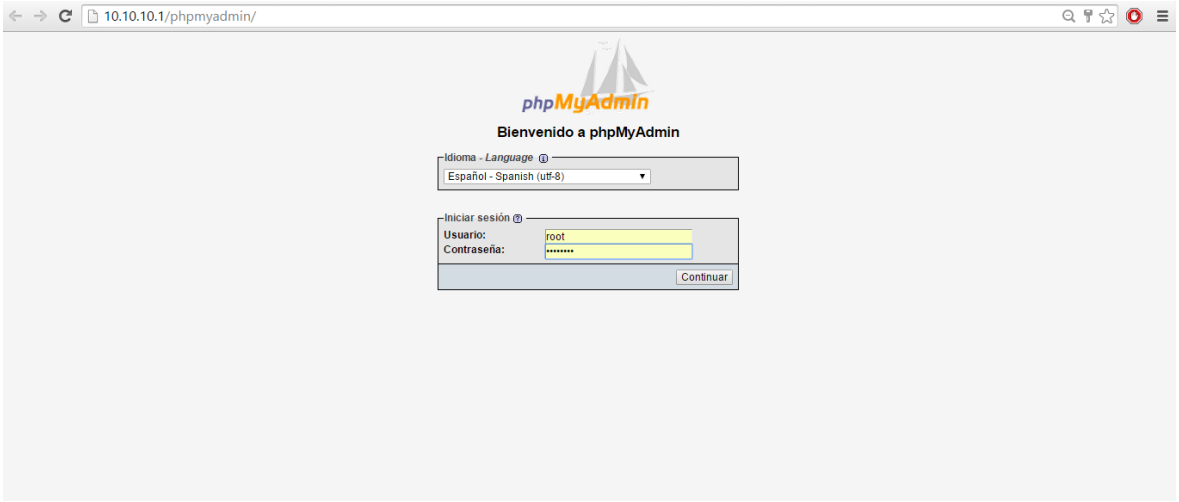


Figura 3.49 Login phpMyadmin.

Para acceder se han de introducir los valores usuario “root” y contraseña “libelium2007”.

3.5.3 Base de datos

En este apartado se indicará el diseño de la base de datos y su implementación para el control de los sensores de parking.

Para el control de los sensores de parquin se ha diseñado una tabla parking con campos de identificador, localización, coordenada X, coordenada Y y PS. Como se puede ver en la figura 3.50

	Campo	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra	Acción
☐	identificador	varchar(50)	latin1_swedish_ci		No			
☐	localizacion	varchar(100)	latin1_swedish_ci		No			
☐	x	varchar(100)	latin1_swedish_ci		No			
☐	y	varchar(100)	latin1_swedish_ci		No			
☐	ps	int(1)			No			

Figura 3.50 Campos tabla parking.

Cada uno de los sensores que se vayan a instalar en la red de sensores, para gestionar las plazas de aparcamiento, se debe dar de alta manualmente en la base de datos junto con su identificador, localización y coordenadas. El identificador será la clave principal, ya que debe ser único para no confundir los sensores. El estado de la tabla se puede consultar en la figura 3.51. Dentro de esta aparecen los dos sensores de parking con lo que se cuenta para este proyecto, uno el asociado al Waspmote 1.1 y el otro al 1.2.

Ambos se han dado de alta con coordenadas correspondientes con dos puntos situados en el nuevo campus de Linares, como podremos ver en el siguiente apartado. A estos se les ha asociado su identificador que será el que tienen que utilizar para las comunicaciones.

	identificador	localizacion	x	y	ps
	parking_1	Campus	-3.64642230000004	38.0866796	0
	parking_2	Campus	-3.6447050619628953	38.0858989676069	1

Figura 3.51 Contenido de la tabla parking.

El valor de PS será el estado en el que se encuentra la plaza de aparcamiento y a la vez, el único variable de esta tabla una vez se ha dado de alta el sensor. Se actualizará mediante el uso de la aplicación con la última entrada correspondiente a cada identificador en la tabla Sensor Parser.

3.5.4 Aplicación Web.

En esta sección se explicará el funcionamiento de la aplicación implementada para consultar el estado en el que se encuentran las plazas de aparcamiento.

La aplicación se subirá al router a través del cliente SFTP citado activando previamente el modo lectura y escritura mediante la conexión SSH establecida con el cliente citado.

El objetivo de la web era mostrar en un mapa de google el estado de las plazas mediante un código de colores (verde libre y rojo ocupado), para poder mostrar esto dentro del mapa, se tiene que contar con una cuenta de desarrollador de Google, para conseguir una contraseña de la API y que se pueda mostrar el mapa correctamente. Esto se puede hacer de forma sencilla visitando la página para desarrolladores de google y luego creando un proyecto desde la consola de desarrollo, mostrada en la figura 3.52.

Figura 3.52 Creación de un proyecto de Google para obtener una key de la API.

La aplicación hace una consulta a la tabla parking, emplazada en la base de datos del Meshlium, para poder mostrar el circulo en el mapa correspondiente a el sensor asociado utilizando las coordenadas x e y, además del estado de PS. Pero antes actualiza esta tabla con las entradas de Sensor Parser, por si ha ocurrido un cambio en el estado del sensor de parking en el tiempo transcurrido desde la última consulta. Para actualizar esta tabla se mira para cada sensor, la última entrada con su identificador y sensor PS, a continuación se modifica en valor de PS en la tabla parking, en la línea correspondiente a cada sensor.

El mapa resultante de la aplicación se puede observar en la figura 3.53. En el cual se muestran los dos sensores de parking disponibles uno con la plaza ocupada y otro con la plaza libre.

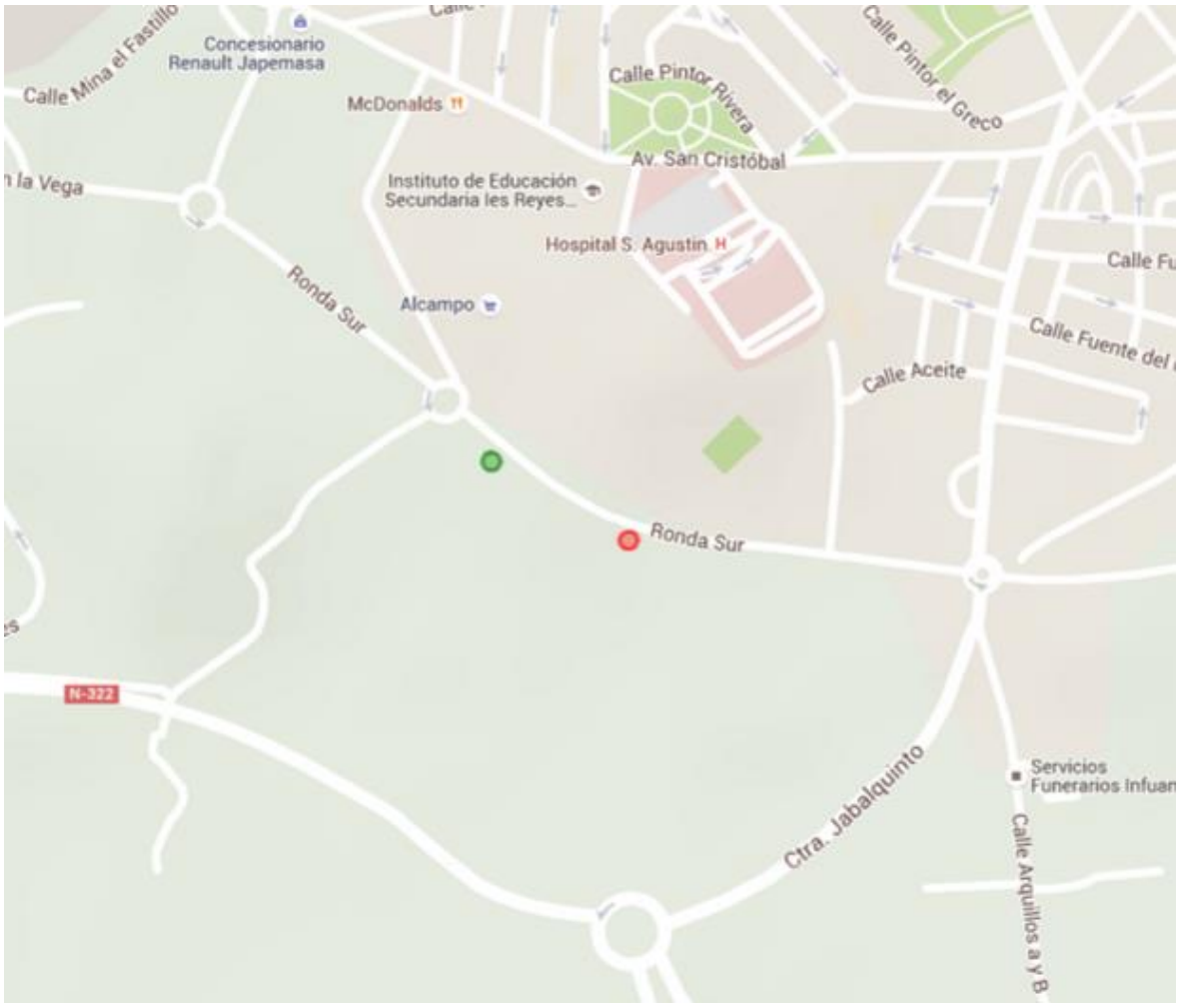


Figura 3.53 Mapa resultante.

Además también se ha implementado un buscador que realiza una búsqueda filtrando por el parámetro localización y muestra el número de plazas libres en una determinada zona. La búsqueda se hará a través de un selector en el que se muestran todas las zonas de la base de datos. Esta búsqueda se muestra en la figura3.54.

Figura 3.54 Búsqueda plazas libres en zona campus.

Capítulo 4
RESULTADOS Y DISCUSIÓN

Como resultado del proceso desarrollado en el Capítulo 3, ha sido posible poner en marcha las dos topologías de red citadas, con el uso tanto de comunicaciones 802.15.4 y DigiMesh. A continuación se enumeran las pruebas realizadas y los resultados obtenidos.

Pruebas realizadas:

-Se ha realizado una prueba de testeo del sensor de parking, situándolo dentro de su encapsulado y aparcando un coche encima del sensor, comprobando el correcto funcionamiento de la placa de aparcamiento que detectaba perfectamente la presencia del vehículo.

-Se ha realizado una prueba de distancia de transmisión con los módulos DigiMesh, desde un extremo a otro del nuevo campus en Linares (unos 500 metros), y las comunicaciones fueron en todo momento satisfactorias.

-Se ha puesto en marcha una red de sensores Waspmote versión 1.2 con módulos DigiMesh, con el objetivo de crear un modo síncrono cíclico de sueño con un coordinador y un coordinado, se observó que se respetaba perfectamente el tiempo de sueño pero no el tiempo de estar despierto, si algún nodo terminaba antes sus tareas programadas que el tiempo de estar despierto, se ponía a dormir y se despertaba con la señal del coordinador después de este si respetar el tiempo de sueño.

-Se ha realizado una prueba de envío multi-salto en comunicaciones DigiMesh, para comprobar que sin tener acceso directo desde el emisor al receptor se pude llegar a través de los otros módulos de la misma red, con resultados satisfactorios.

-Se ha realizado una prueba de consumo con la potencia de transmisión al mínimo y con un intervalo de envío de 5 minutos, entre envío y envío el sensor se encuentra durmiendo, la prueba se realizó durante 24 horas, y durante ese tiempo se consumió algo más del 1% de una batería de 1150 mAh. El fabricante indica que con una batería de 26 Ah y con mediciones y envío con el mismo intervalo que la prueba realizada la duración de la batería sería de 5.3 años. Con los datos obtenidos se puede concluir que estos datos son viables y que las baterías podrían tener alrededor de 5 años de vida útil. Se ha que tener en cuenta que al colocar la potencia de envío al máximo el consumo casi se duplica por lo que la autonomía con una batería de 26 Ah no superaría los 3 años.

Resultados obtenidos:

-Se ha conseguido configurar los Waspmote para que obtengan el estado del sensor de parking y el posterior envío de este a través de los diferentes modos de transmisión, tanto en la versión 1.1 como en la 1.2.

-Se ha realizado la configuración de todos los módulos 802.15.4 disponibles y la programación de los Waspmote de las dos versiones y se ha puesto en marcha la topología diseñada para esta comunicación.

-Se han configurado los módulos DigiMesh para montar una red mallada con ciclos de sueño y posible reenvío de datos si el emisor no tiene contacto directo con el objetivo además de la programación de los Waspmote para que la red funcione correctamente.

-Se ha creado con éxito una aplicación web que actualiza la base de datos parking, situada dentro del router Meshlium con los identificadores de los sensores y el estado de la plaza de aparcamiento, usando las tramas recibidas que se almacena en la tabla Sensor Parser. Además muestra en un mapa el estado de cada plaza de aparcamiento asociada a su correspondiente sensor de forma gráfica y se tiene un selector en el que se realiza una consulta para obtener el número de plazas libres en una determinada zona.

Capítulo 5
CONCLUSIONES

Este Trabajo Fin de Grado tenía como objetivo principal el diseño, programación e implementación de una plataforma de parking exterior inteligente mediante red de sensores inalámbricos. Y tras los métodos y configuraciones realizados en el capítulo 3 y las pruebas y resultados expuestos en el capítulo 4, se puede concluir que se han cumplido totalmente los objetivos planteados se ha llegado a las siguientes conclusiones:

- El modo de transmisión DigiMesh ha resultado mucho más interesante de lo esperado, debido a todas las funcionalidades que implementa respecto al modo de transmisión 802.15.4.

- El sistema realizado con los módulos DigiMesh ha resultado ser más funcional y útil que el realizado con modo de transmisión 802.15.4. Sobre todo la funcionalidad de reenvío de paquetes, descubrimiento de rutas y el mayor alcance que se obtiene con los módulos asociados a DigiMesh.

- Se puede decir que la implementación de las dos topologías expuestas es totalmente viable y no sería a un elevado coste, ya que una vez instalado la autonomía de funcionamiento de los sensores podría rondar los 5 años y pasado estos solo habría que sacar los sensores cambiar las baterías y volver a enterrarlos en el asfalto.

Capítulo 6
REFERENCIAS BIBLIOGRAFICAS

Bibliografía

[1] Alberto Bielsa; Smart City project in Santander to monitor Parking Free Slots, Libelium World, February 22nd, 2013.

[2] IEEE Standard 802.15.4-2003, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), IEEE, October 2003.

[3] National Instrument; Five Factors to Consider When Implementing a Wireless Sensor Network (WSN), June 27, 2012.

[4] Libelium Comunicaciones Distribuidas; DigiMesh Networking Guide, June, 2015.

[5] Libelium Comunicaciones Distribuidas; Smart Parking Board Technical Guide, November, 2014.

[6] Patricia Núñez; La búsqueda de aparcamiento provoca el 30% de los atascos, IBM, Septiembre, 2011.