Bluetooth Low Energy (BLE)

18 de agosto de 2019

Preparado a partir de distintas fuentes como recurso rápido para adquirir una visión básica sobre beacons BLE. La lectura de este pequeño texto debe ser complementada con el estudio de sus fuentes:

Getting Started with Bluetooth Low Energy, O'Reilly Media, 2014.
Introduction to Bluetooth® Low Energy, wiki/tutorial de Microchip.
Bluetooth® low energy Beacons, Texas Instruments, 2016.

Introducción

Desde la aparición de la especificación 4.0 en 2010, las especificaciones Bluetooth 4.x permiten a los dispositivos implementar, usando la misma antena y banda de frecuencias, una interfaz Bluetooth clásico o una interfaz Bluetooth LE (o BLE). BLE nace con la vocación de permitir la creación de productos IoT que operen durante meses o años con una pila de botón. Hay dispositivos *single-mode* (que sólo son compatibles BLE) y dispositivos *dual-mode* (compatibles Bluetooth clásico y LE).

La versión Bluetooth 5 añade sobre las 4.x un mayor alcance, mayor velocidad de transmisión (si bien limitada a 1 Mbps máximo teórico, siendo un valor realista 5-10 kByte/s) y mayor longitud de la trama de advertising (hecho al que podemos dar un buen uso en este proyecto, como veremos más adelante).

Topología de red y roles

Junto al modo broadcast, BLE define también un modo basado en conexión, cada uno con sus ventajas y desventajas.

Modo broadcast
En broadcasting la comunicación es unidireccional. Se define el rol de broadcaster y el de observer. El broadcaster envía periódicamente paquetes de advertising (no orientados a conexión y que pueden llevar una pequeña payload, ya sea un identificador o una medida) que puede recibir cualquier dispositivo que esté escuchando. Los observers escanean (proceso de scanning) las frecuencias (canales) reservados para broadcasting y reciben los paquetes de advertising.

Un paquete de advertising contiene una payload de 31 bytes con información sobre el broadcaster, dejando hueco para unos pocos bytes donde podemos añadir nuestra información (medida de temperatura, humedad y concentración de gases). BLE contempla

un método para enviar más datos, mediante un paquete *scan response*, pero no lo usaremos en nuestro proyecto.

Los beacons BLE hacen uso del broadcasting y uno de los métodos más extendidos es el iBeacon definido por Apple, uno de los gigantes tecnológicos que más ha apostado por BLE.

Modo basado en conexión

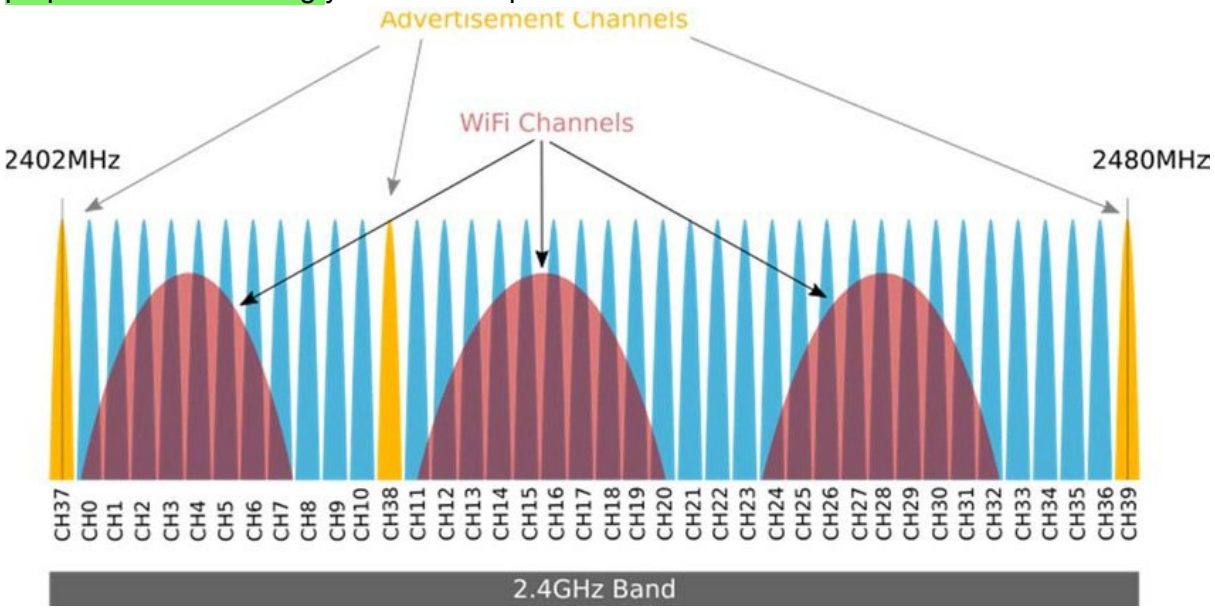
La transmisión bidireccional de datos requiere establecer una conexión, de modo que ya no podremos usar el modo broadcasting. Se define el rol de nodo central (central node) que escanea en busca de paquetes de advertising enviados por periféricos (rol peripheral) con peticiones de conexión y en su caso, mediante un protocolo, establece una comunicación privada con el periférico. La conexión supone el envío periódico de paquetes de datos en las dos direcciones, sin establecerse prioridad o diferencia por el mero hecho de ser el nodo central o el periférico. Los roles, por tanto, se limitan al proceso de establecimiento de la conexión.

Un periférico puede estar conectado a múltiples nodos centrales. Un nodo central a múltiples periféricos. Y un nodo puede ser a la vez central y periférico. Esto permite crear topologías de red complejas.

Bluethoot y WIFI tiene banda libre
Entonces es muy facil que se solapen.

Capa física en BLE

BLE define 40 canales en la banda ISM de 2.4 GHz, de los que 3 se usan para el envío de paquetes de advertising y 37 de datos para las comunicaciones basadas en conexión. Beacons



Los tres canales de advertising (37, 38 y 39) están cuidadosamente elegidos para no solaparse con canales WiFi. Las comunicaciones de datos usan frequency hopping, de modo que el canal usado por una conexión va cambiando con el tiempo. Se minimiza así el efecto de interferencias.

Donde más renta enviar los beacons

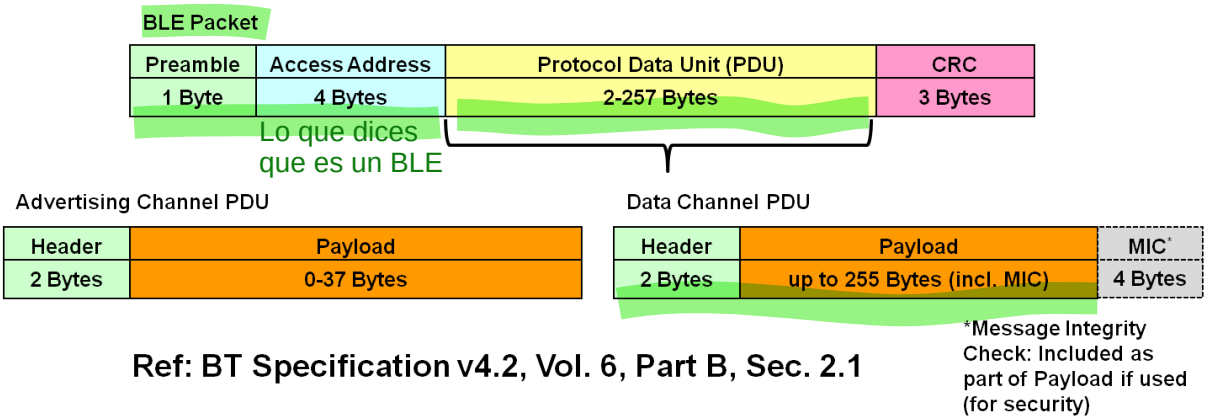
El movil no va a escuchar todos los beacons porque se pueden perder entonces se envian en tres canales para que sea mejor.

Capa de enlace en BLE

BLE= beacon

Las direcciones físicas Bluetooth constan de 48 bit (6 bytes), de forma similar a las direcciones MAC en Ethenet. Las direcciones pueden ser públicas (las concede el IEEE bajo petición, son fijas para cada dispositivo y nunca cambian) o aleatorias (random, para permitir mayor privacidad). Estas direcciones aleatorias pueden ser estáticas (sólo cambian tras un power-up) o privadas (cambian cada cierto tiempo).

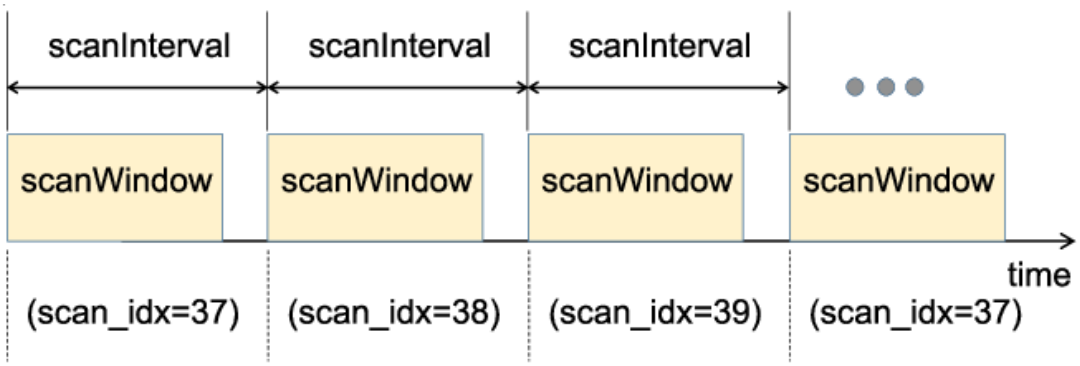
Los paquetes BLE a nivel de capa de enlace tienen un formato único y pueden ser de dos tipos, advertising (los únicos usados en modo broadcast, aunque también se usan en modo conexión para descubrir periféricos -esclavos- desde el nodo central -master-) y tipo data (usados en modo conexión para enviar datos bidireccionalmente).



Ref: BT Specification v4.2, Vol. 6, Part B, Sec. 2.1

https://microchip.wdfiles.com/local--files/wireless:ble-link-layer-packet-types/packet-format-top-level.png

Los paquetes de tipo advertising se envían periódicamente (seleccionable entre 20 ms y 10.24 s). Un advertiser (broadcaster o peripheral) envían paquetes cada cierto intervalo en hasta 3 canales secuencialmente. Un scanner (observer o central node) escanea estos tres canales conforme a dos parámetros (scan interval -cada cuánto pasa de un canal a otro- y scan window -cuánto tiempo escanea cada canal-).



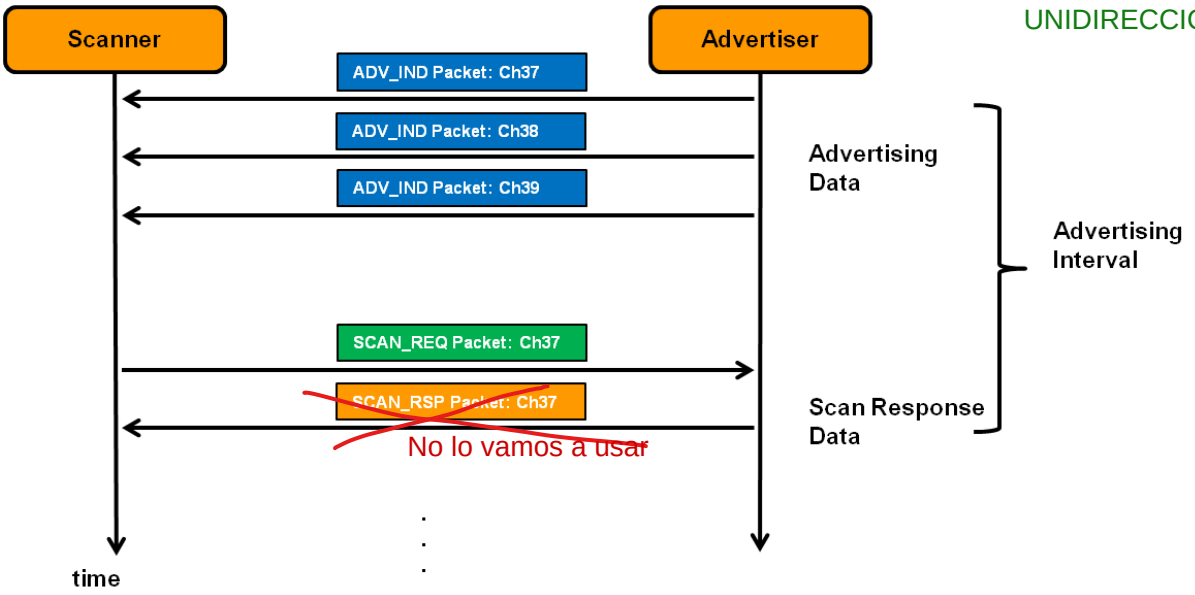
https://ai2-s2-public.s3.amazonaws.com/figures/2017-08-08/ca597996259349c64f89d023a2af1bb8481f975e/3-Figure2-1.png

En función de la frecuencia con que se envían las tramas de advertising y los parámetros de escaneo en el scanner, pueden perderse más o menos tramas antes de que una se reciba

correctamente. En el caso de escaneo pasivo todo termina aquí: no hay respuesta por parte del scanner.

En el caso de escaneo activo, el scanner responde con un SCAN_REQ (scan request packet), a lo que el advertiser correspondería con un SCAN_RSP (scan response packet).

Un ejemplo del proceso de escaneo activo queda reflejado en la siguiente figura.



https://microchip.wdfiles.com/local--files/wireless:ble-link-layer-discovery/advertising-event.png

Tipos de paquetes de advertising

En función del tipo de conexión que pretenda el advertiser, las tramas de advertising informan sobre tres atributos:

- Conectable (caso de modo de conexión) o no conectable (caso de modo de broadcasting)
- Escaneable (y el scanner puede enviar un paquete de scan request) o no escaneable (y no está permitido un paquete de scan request por parte del scanner)
- Directed (se indica dirección del advertiser y del scanner, se usa en modo de conexión) o indirected (no se indica dirección del destinatario, es el caso de broadcasting)

Las librerías para conexión BLE definen constantes para las distintas combinaciones posibles con las que definirás el tipo de paquete de advertising que quieres enviar. Por ejemplo, para broadcasting se usan los siguientes tipos de paquetes de advertising:

ADV_IND, ADV_NONCONN_IND, ADV_SCAN_IND

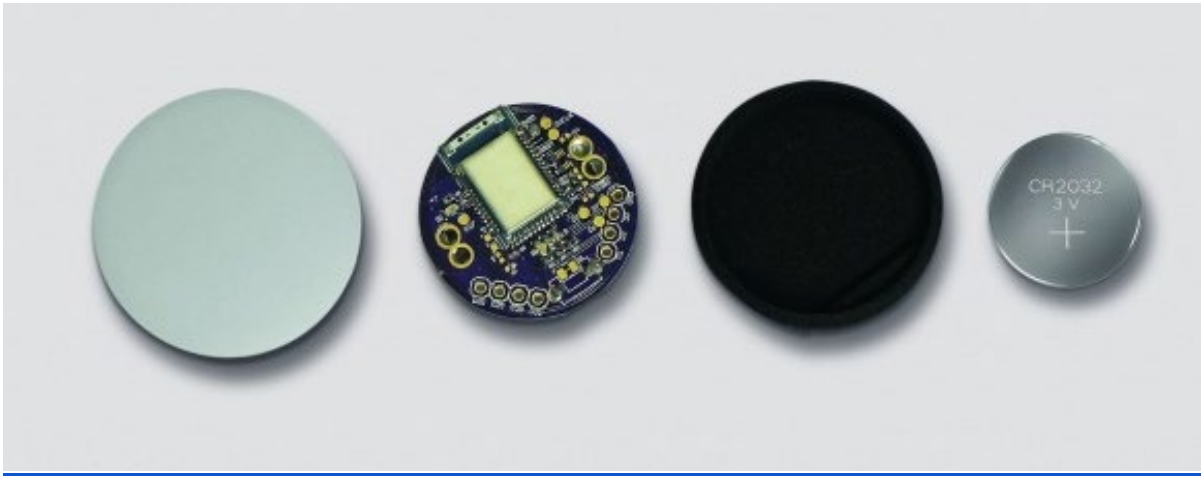
ADV_IND indica un paquete de broadcast (no indica destinatario) conectable y no escaneable.

ADV_SCAN_IND indica un paquete de broadcast (no indica destinatario) conectable y escaneable

ADV_NONCONN_IND indica un paquete de broadcast no conectable y no escaneable. Es el tipo que usaremos en nuestro proyecto, porque es el que usa la especificación iBEacon desarrollada por Apple.

Beacons BLE

Un beacon BLE actúa como un faro, transmitiendo a intervalos regulares una información. En el caso de un faro, esta información es estática y dice “estoy aquí”. Para los marineros dice algo más, pues su código (intervalo de repetición) es diferente de otros faros cercanos, de modo que lo identifica. Esta información es estática y un Beacon BLE (dispositivo que transmite paquetes de advertising en broadcast) puede hacer lo mismo… y algo más. Porque tiene a su disposición algunos bytes libres en el paquete de advertising con el que dar información sobre su estado (no queda cerveza en la nevera o el nivel de la batería es bajo) o sobre el mundo exterior (parámetros ambientales, velocidad de giro de un motor, etc).



https://circuitcellar.com/cc-blog/open-source-bluetooth-low-energy-beacon/

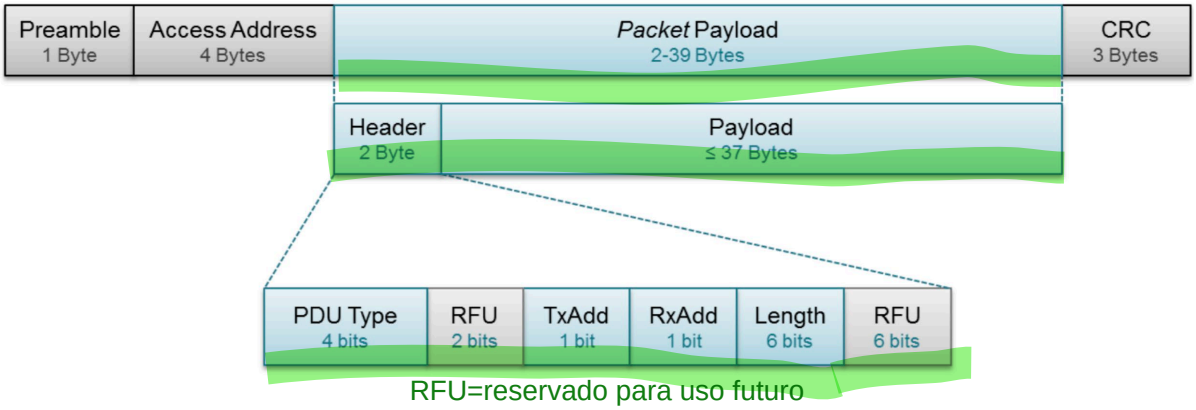
La imagen anterior muestra la estructura interna de un **beacon BLE**. A un precio inferior a $2 (incluyendo micro y antena impresa), compite con otras tecnologías como WiFi o Zigbee en aplicaciones IoT (tales como localización, estado y sensado).

Hay beacons BLE no conectables y por tanto sólo envían información. También hay beacons BLE que pueden entrar en modo orientado a conexión (por ejemplo, para poder cambiar su programa).

Si echamos un vistazo más detallado al formato trama BLE, encontramos los siguientes campos:

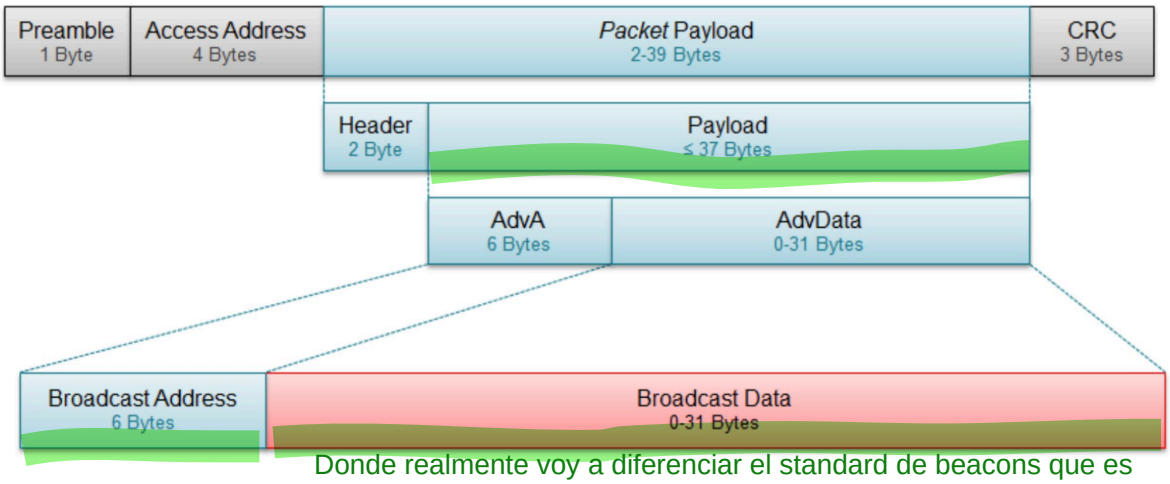
- Preámbulo: es un código fijo de 8 bit (0xAA para paquetes de tipo broadcast) A=1 EN BINARIO
- Access address, un código que es fijo para paquetes de tipo broadcast (0x8E89BED6)

- CRC, es un código de redundancia cíclica de 24 bit para detección de errores
- Payload, carga útil del paquete, con un tamaño entre 2 y 39 bytes. Consta de los siguientes subcampos:
    - Header, 2 bytes que identifican el tipo de trama. PDU Type es “0011” para paquetes ADV_NONCONN_IND. TxAdd indica si la dirección de advertiser es pública (0) o random (1). RxAdd no se usa en beacons. Length indica el tamaño del bloque “payload” en bytes



RFU=reservado para uso futuro

El bloque de payload comienza por la dirección del advertiser (6 bytes) y a continuación encontramos el bloque de **Advertising Data**. Diferentes estándares de Beacons BLE harán diferente uso de este bloque, si bien todos tienen una estructura similar.



Donde realmente voy a diferenciar el standard de beacons que es

Broadcast address
Dirección de 6 bytes del advertiser que puede ser fija (y venir fija en hardware), configurable o incluso aleatoria para permitir mayor privacidad. Depende del dispositivo.

Advertising Data
Este bloque consta de 0 a 31 bytes y es una secuencia de Advertising Data Elements (AD). Cada AD consta de:

- Primer byte: longitud del AD (excluyendo este byte)
- Segundo byte: tipo del AD. Hay una lista de códigos predefinidos aquí.
- AD data, uno o más bytes del tipo indicado en el byte anterior

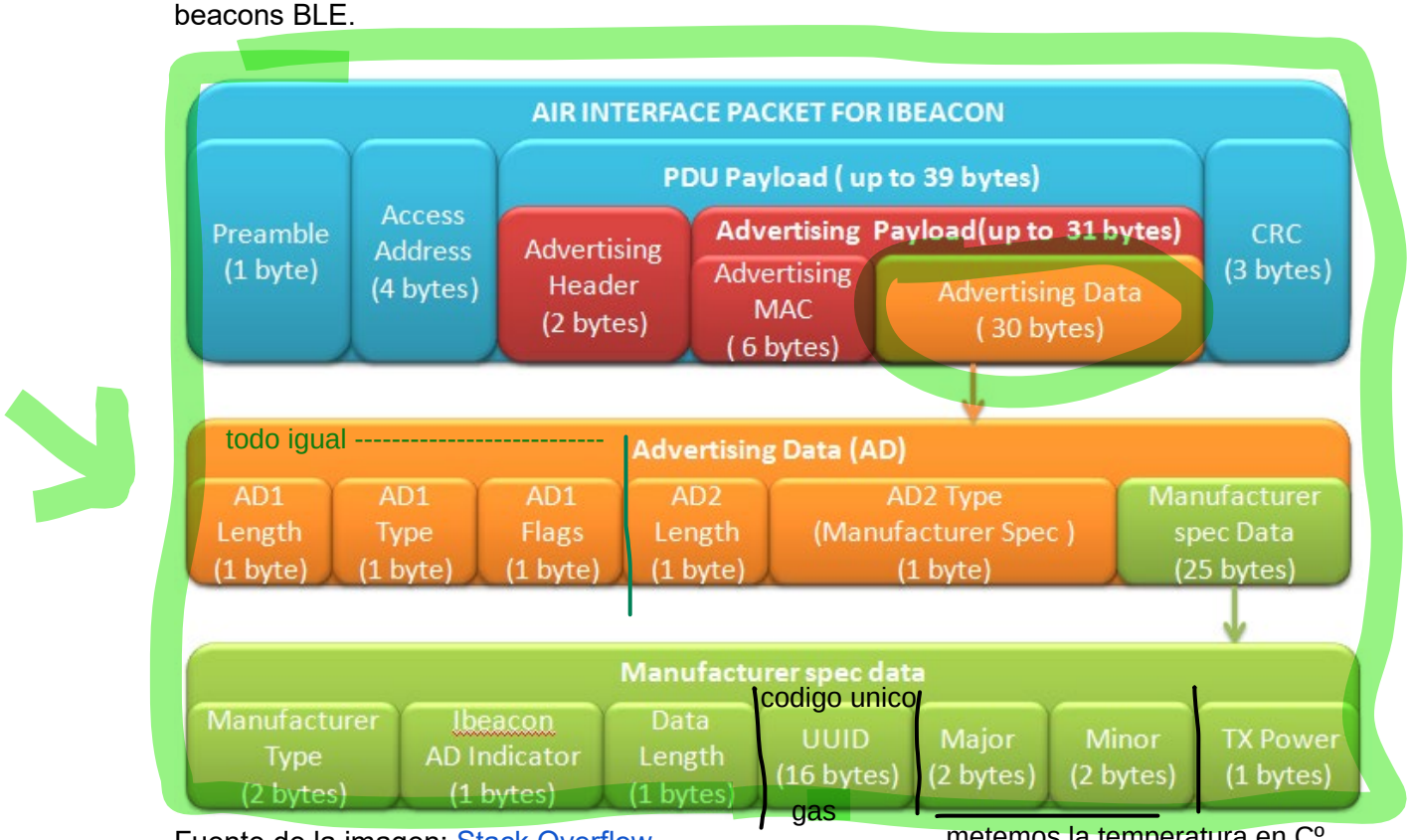
Ejemplo de AD: “0x020106” debe leerse como un AD de 2 bytes, de tipo “flags” cuyo valor es 0x06.

iBeacons

Los dos tipos de beacons más extendidos son iBeacon, de Apple y propietario, y AltBeacon, un estándar abierto. Nuestro módulo Sparkfun Pro nRF82540 Mini usa por defecto iBeacons, de modo que nos centramos en este formato.

El Advertising Data (30 bytes en iBeacon) está formada por dos **Advertising Data Elements** (AD1, AD2 en la figura siguiente).

El primer AD consta sólo de tres bytes: longitud de AD sin contar este byte (02h), tipo (que es “flags”, código 01h) y el valor de los flags que es 06h (indica que el dispositivo es BLE y no soporta BT estándar). Por tanto, la cadena “02:01:06” es común como primer AD a en los beacons BLE.



Fuente de la imagen: Stack Overflow

El segundo AD consta de 27 bytes. Comienza por la longitud (1Ah, es decir 26 bytes sin contar este byte), el tipo de AD (código FFh, “Manufacturer Data Specific”) y le siguen 25 bytes:

- 2 bytes de Company ID (por ejemplo, Apple es”00:4C”). Ten en cuenta que este campo se envía en formato little endian (es decir, se envía 4C:00 en lugar 00:4C). La tabla de códigos asignados a compañías está aquí.

- 2 bytes de beacon type (02 para iBeacon)

- 1 byte de longitud (15h, es decir, 21 bytes)

Hasta aquí, los iBeacons son todos iguales. Ahora comienza la parte que puedes tocar:

Lo que vamos a tocar

- 16 bytes de UUID, que podemos fijar a nuestro antojo. Hay que escoger un código que identifique a nuestro servicio. UUID es un acrónimo para “Universally Unique ID”. Identifica un atributo (servicio, característica o un descriptor).Hay UUIDs cortos, de 16 bit, que recoge servicios predefinidos y UUIDs largos de 128 bit (16 bytes). No hay una base de datos mundial que recoja todos los UUIDs, pero si generas un UUID aleatoriamente, la probabilidad de que lo use otro servicio es remota (de hecho es 3.10^{-39}). iBeacon usa UUIDs de 128 bit que puedes generar aleatoriamente o basándote en un mensaje. Por ejemplo:
`45:50:53:47:2d:47:54:49:2d:50:52:4f:59:2d:33:41`
corresponde a los códigos ASCII de la cadena “`EPSG-GTI-PROY-3A`“.

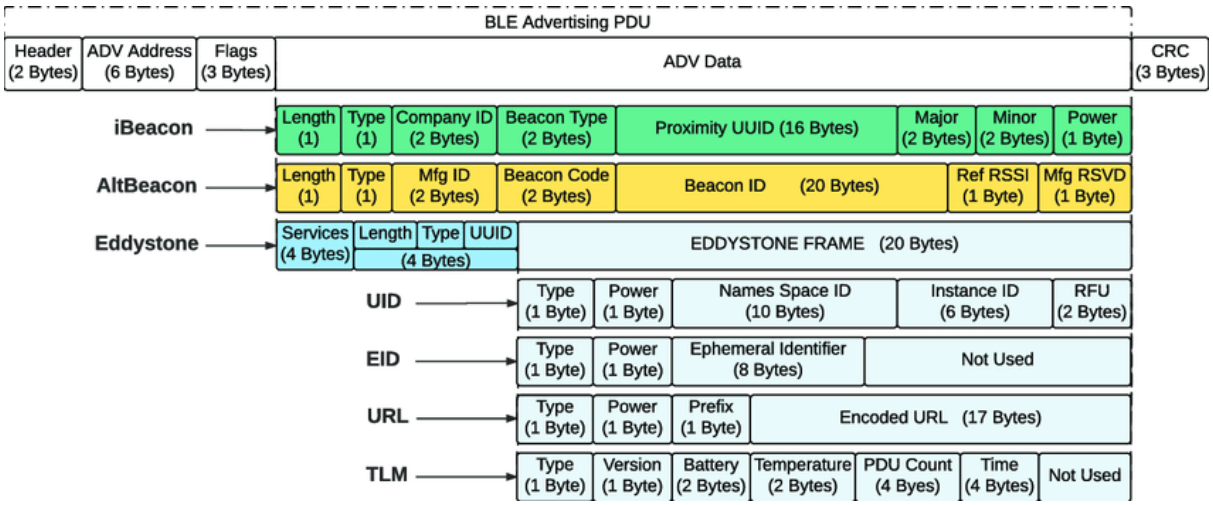
- 2 bytes de “major” y 2 bytes de “minor”, que podemos fijar a nuestro antojo y en donde os sugerimos poner la información del sensor

- 1 byte de potencia de la señal de radiofrecuencia (usado para estimar distancias)

IBeacon especifica que se deben usar los tres canales disponibles para el advertising, y que el periodo de los beacons debe ser de 100 ms.

Otros tipos de beacons BLE

Los otros estándar de Beacon más extendido junto a iBeacon son AltBeacon y Eddystone. En la siguiente figura puedes observar las diferencias.



Fuente de la imagen: Researchgate