IoT (Internet of Things)

Stefano Di Lena

2025

Indice

1 Introduzione

IEEE(2015): "L'Internet of Things è un sistema che consiste in una rete di sensori, attuatori e altri oggetti *smart* che hanno come obiettivo interconnettere *tutte* le cose in modo da renderle intelligenti, programmabili e capaci di interagire tra di loro e con gli umani".

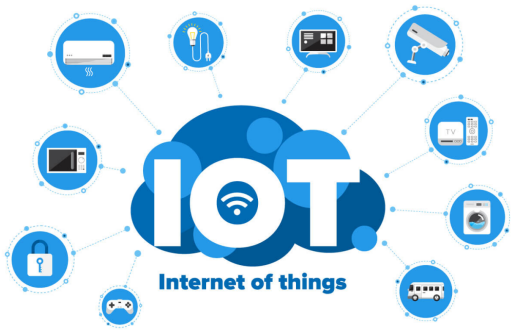


Figura 1.1: IoT (Internet of Things)

1.1 Requisiti IoT

- **Affidabilità**: può essere espressa come il massimo tempo percentuale in cui il sistema soddisfa i requisiti. Devono funzionare nel 99.999% dei casi critici. Per aumentare il livello di affidabilità la ridondanza può essere introdotta in tutti gli elementi di un sistema IoT (sensori, servers, etc.).
- **Latenza**: deve essere inferiore di $100ms$ per classificare un sistema IoT come "*Real-Time*". Per ridurre le latenze si possono aumentare le risorse computazionali e di trasmissione evitando di creare bottlenecks
- **Scala**: rappresenta la dimensione del sistema (in termini di numeri dei nodi, moduli, operazioni, server). Un sistema è ben scalabile quando è in grado di adattarsi a run-time, quindi riuscendo ad operare senza conoscere tutti gli elementi a priori.
- **Sicurezza**: uno dei temi fondamentali dell'IoT, abbiamo device molto fragili perché hanno una debole potenza computazionale.
- **Privacy**: IoT utilizza molti dati, questi vanno protetti perché contenenti informazioni personali.
- **Frammentazione**: ci sono molti blocchi non cooperanti tra loro, è necessario sviluppare dei middlewares in grado di rendere possibile la cooperazione tra vari sistemi.

2 Modulazione

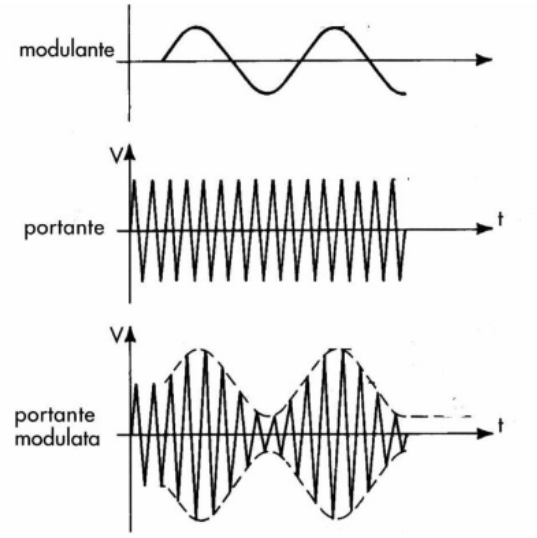


Figura 2.1: Segnale da Trasmettere, Portante e Segnale Modulato

La modulazione è una tecnica che consente di poter inviare un segnale informativo (un bit o un insieme di essi). Questo segnale informativo viene moltiplicato per un segnale modulante (seno o coseno), ottenendo il segnale modulato.
Ci sono diversi tipi di modulazioni:

- Fase
- Ampiezza
- Frequenza

$$s(t) = a(t) \cdot cos(2\pi ft + \phi(t))$$

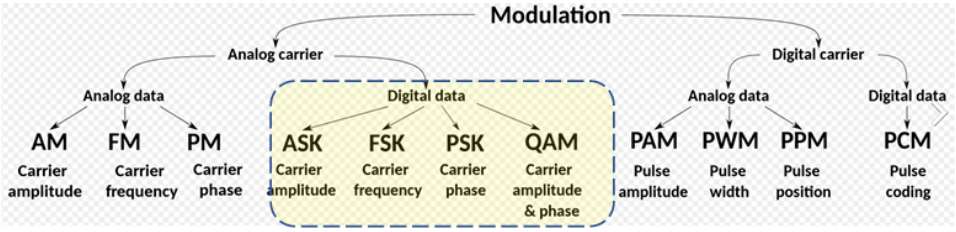


Figura 2.2: Tipi di Modulazione

Quando si effettua una modulazione bisogna prestare attenzione all'ordine di modulazione 'M' (un numero che precede il tipo di modulazione e fa capire quanti simboli è possibile trasmettere con essa).
Ci sono tabelle in cui sono indicate le tipologie di modulazione, il coding rate ed il data rate.

MCS Index	Modulation	Coding rate	Data rate (kbps)	
			1 MHz	2 MHz
0	BPSK	1/2	300	650
1	QPSK	1/2	600	1300
2	QPSK	3/4	900	1,950
3	16-QAM	1/2	1,200	2,600
4	16-QAM	3/4	1,800	3,900
5	64-QAM	2/3	2,400	5,200
6	64-QAM	3/4	2,700	5,850
7	64-QAM	5/6	3000	6,500
8	256-QAM	3/4	3,600	7,800
9	256-QAM	5/6	4000	Not valid
10	BPSK	1/2 with 2x repetition	150	Not valid

Figura 2.3: IEEE 802.11ah MSC Setting vs Data Rate with 1.2MHz Bandwidth, $NSS = 1$ and $GI = 8\mu m$

Il coding rate serve a proteggere i bit informativi quando si trasmettono dei dati attraverso dei bit di controllo (ad esempio un coding rate 1/2 vuol dire che trasmettiamo 1 bit informativo ogni 2; se usiamo un coding rate 1/4 vuol dire che trasmettiamo 4 simboli e di questi quattro solo uno è quello di informazione, gli altri sono di controllo). All'aumentare del Coding Rate aumenta anche il Data Rate.

- Symbol Rate (R_s): numero di simboli che è possibile trasmettere al secondo
- Bit Rate (R_b): numero di bit che è possibile trasmettere al secondo

R_b ed R_s sono legati tra loro da M $\rightarrow R_b = log_2(M) \cdot R_s$

Per M=2 $\rightarrow log_2(2) = 1 \rightarrow R_b = R_s$

- *Tempo*: $T_s = log_2(M) \cdot T_b$
- *Energia*: $E_s = log_2(M) \cdot E_b$
- *Probabilità di Errore*: $P_s = log_2(M) \cdot P_b$

Quando si confrontano le modulazioni in termini di Energia, si deve tenere contro di E_b e NON di E_s (perché E_s dipende dal numero dei bit per simbolo).

Per il rapporto segnale-rumore:

$$SNS = \gamma_b = \frac{E_b}{N_o}$$

Dove N_o è la *densità spettrale di potenza* del rumore $\rightarrow N_o = \frac{N}{BW}$. N è il rumore e BW è la banda.

Invece il numero di simboli che è possibile trasmettere e il numero di bit trasferiti per ogni simbolo sono legati dalla seguente formula:

$$2^N = M \rightarrow N = log_2(M)$$

Un altro parametro importante è *l'efficienza spettrale* (η), che rappresenta il numero di bit che è possibile trasmettere al secondo per Hz:

$$\eta = \frac{R_b}{BW} = \frac{log_2(M) \cdot R_s}{BW}$$

La banda BW è solitamente uguale a $R_s = \frac{1}{T_s}$ se in T_s (tempo di trasmissione del simbolo) viene trasmesso un impulso di Nyquist.

Se invece in T_s viene trasmesso un impulso rettangolare, la banda BW sarà uguale a $2R_s = \frac{2}{T_s}$

Quindi: $R_s \leq BW \leq 2R_s$

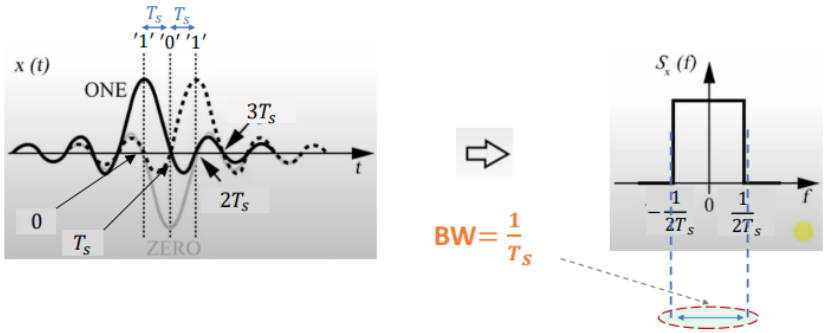


Figura 2.4: L'impulso di Nyquist è il minimo valore di bandwidth per il segnale modulato

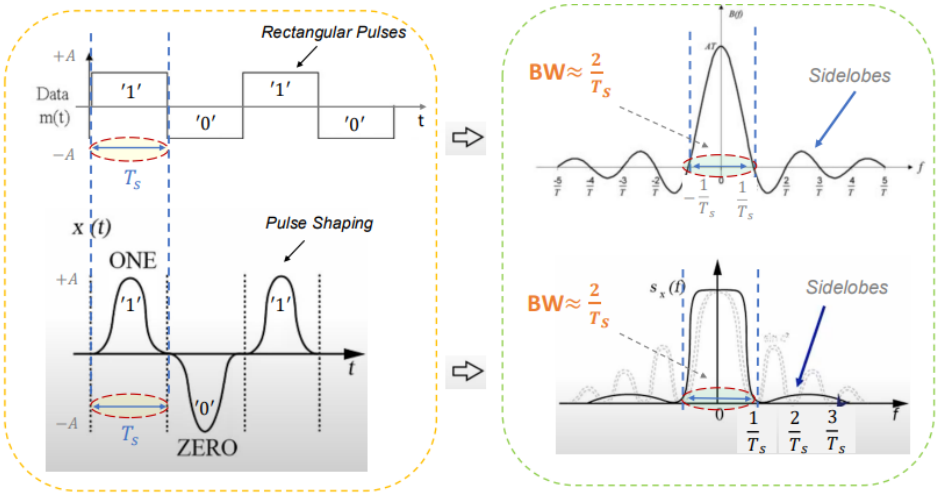


Figura 2.5: Bandwidth e pulse

2.1 Modulazione di Ampiezza

In questa modulazione l'ampiezza varia mentre fase e frequenza sono costanti.

$$s(t) = a(t) \cdot cos(2\pi ft + \phi(t))$$

2.1.1 ASK

In questa modulazione l'ordine è M=2 (è possibile trasmettere quindi 0 oppure $1 \rightarrow R_b = R_s$ e $T_b = T_s$).

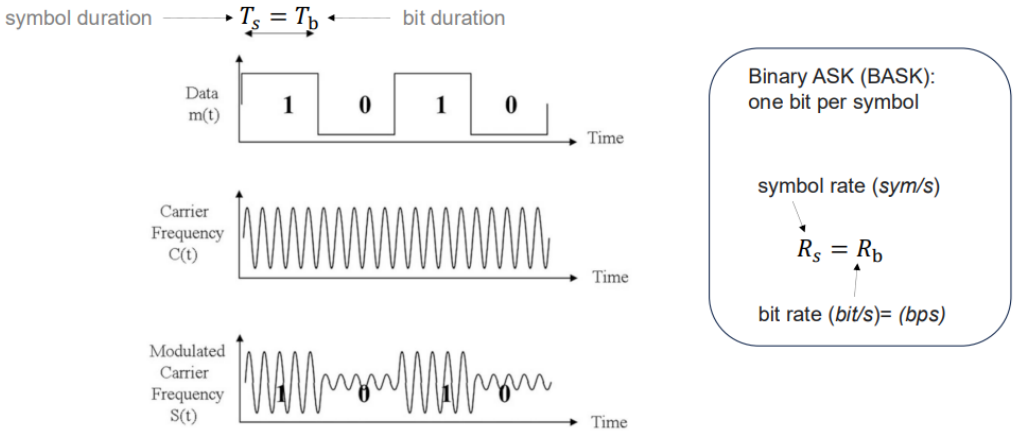


Figura 2.6: Binary ASK (BASK)

Problematica: è sensibile al rumore. Se trasmetto un simbolo con una certa ampiezza, a causa del fading del canale, in ricezione si può ricevere un segnale con un valore di ampiezza diverso da quello inizialmente trasmesso (quindi potrebbe essere scambiato un simbolo con un altro, ad esempio un 1 ed uno 0).

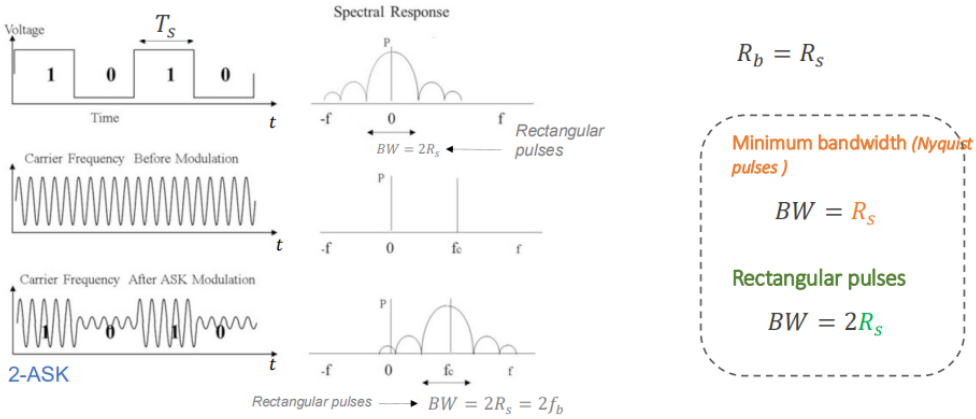


Figura 2.7: 2-ASK

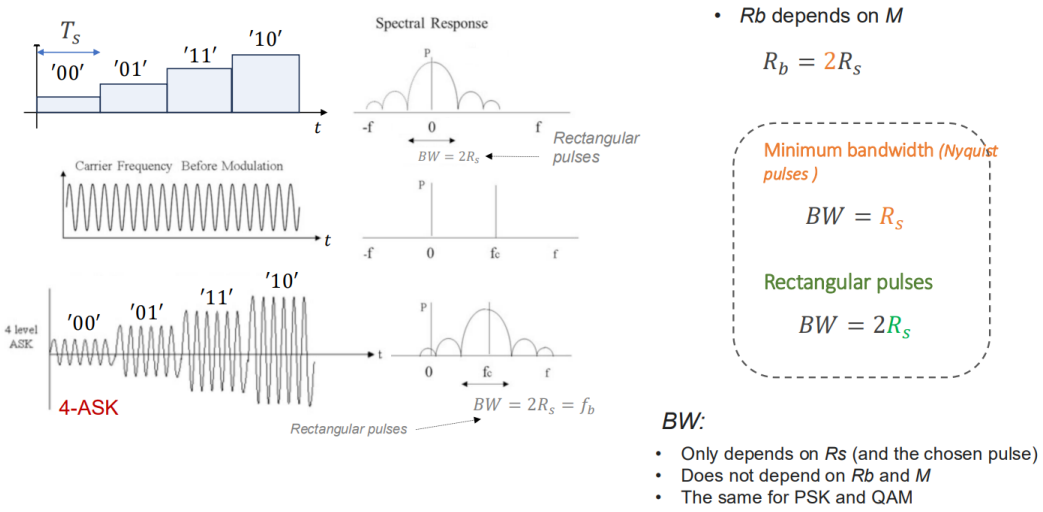
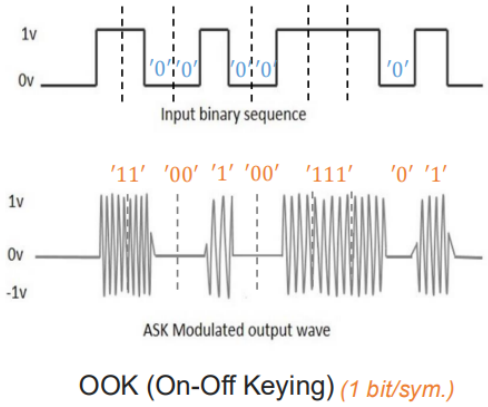


Figura 2.8: 4-ASK

2.1.2 OOK (On-Off-Keying)



Nella modulazione OOK si trasmette un segnale quando si deve trasmettere 1 e non si trasmette nulla quando si deve trasmettere 0. É buono quindi dal punto di vista del consumo energetico, perché quando trasmettiamo 0 risparmiamo energia (non trasmettendo nulla).

2.2 Modulazione FSK

É una modulazione di frequenza robusta al rumore ma lenta, quindi viene spesso usata per separare i canali e non per trasmettere un segnale.

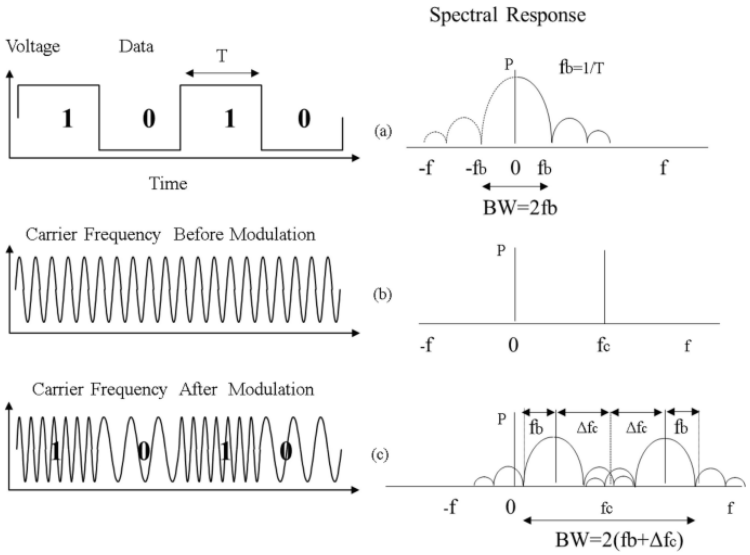


Figura 2.9: BFSK

$$f_b = R_b = R_s = \frac{1}{T_s} \quad , \quad \Delta = f_2 - f_1 = 2\Delta f_c$$

La banda nella FSK può essere $BW = 2R_s + \Delta$ (Rectangular pulses) oppure $BW = R_s + \Delta$ (Nyquist pulses).

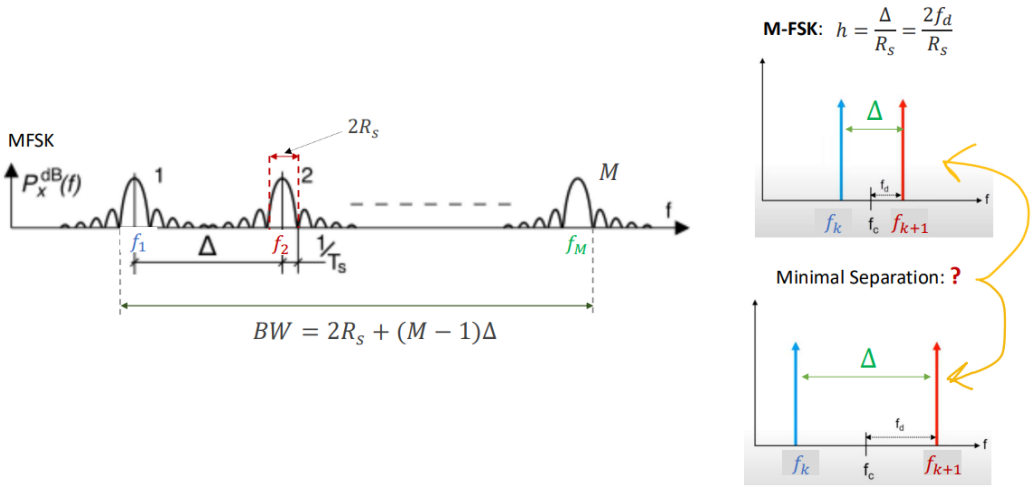


Figura 2.10: MFSK

Tutte le M frequenze possibili devono essere a coppie ortogonali per evitare l'interferenza intersimbolica. $BW = 2R_s + (M-1)\Delta$

- Modulazione Non-Coerente: $\int_0^{T_s} cos(2\pi f_1 t + \theta) cos(2\pi f_2 t)\, dt = 0$
- Modulazione Coerente: $\int_0^{T_s} cos(2\pi f_1 t) cos(2\pi f_2 t)\, dt = 0$

2.3 Modulazione QAM

In questa modulazione l'informazione è contenuta sia nella fase che nell'ampiezza (ed esse sono tra loro in quadratura).

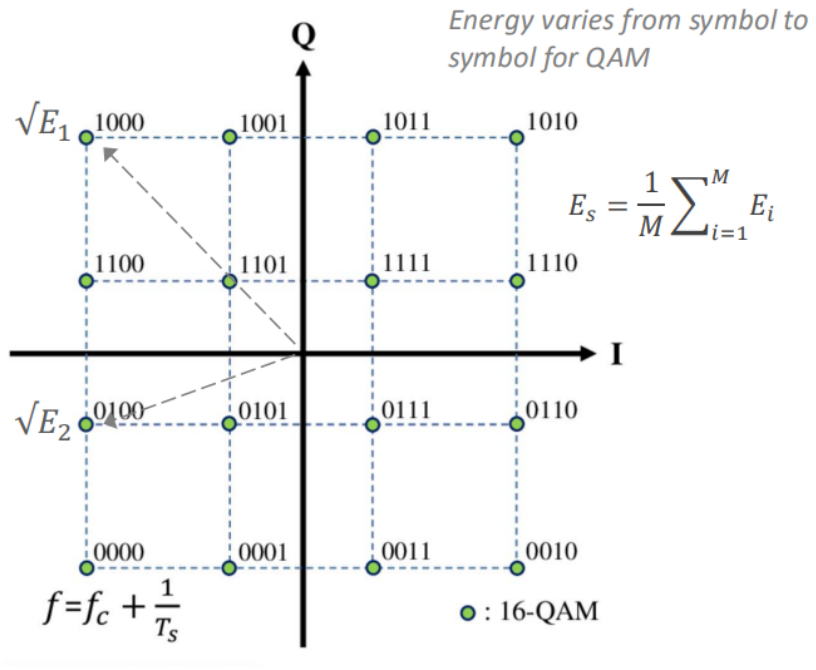


Figura 2.11: 16-QAM Constellation

É possibile esprimere la forma d'onda come somma di una *cosinusoide* e di una *sinusoide* entrambe modulate in ampiezza:

$$s(t) = a(t)cos(2\pi ft + \phi(t)) \xrightarrow{prostaferesi} s(t) = s_I(t)cos(2\pi ft) - s_Q(t)sin(2\pi ft)$$

L'ampiezza è il vettore che va dall'origine ad un singolo pallino. L'energia di ogni singolo simbolo E_i sarà differente per ogni simbolo, perché ampiezza ed energia sono legate dall'equazione: $A = \sqrt{\frac{2E_s}{T_s}}$

Se consideriamo un diverso ordine di modulazione (ad esempio passando da 16-QAM a 64-QAM) avrò un numero di simboli più alto ma la distanza tra i simboli sarà minore. Aumenterebbe M, avremo un maggiore Data Rate ed una maggiore banda ma il rischio di sbagliare aumenta.

2.4 Modulazione PSK

É una modulazione di fase.

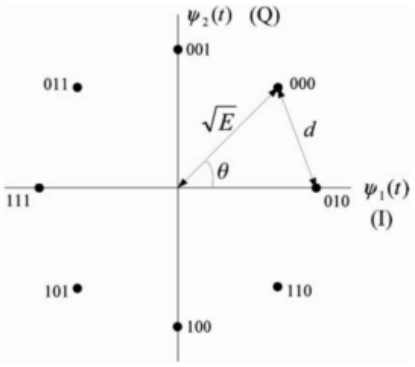


Figura 2.12: 8-PSK Constellation

In questa modulazione tutti i simboli avranno stessa ampiezza, se si inseriscono tutti i simboli su una circonferenza l'ampiezza sarà pari al raggio della stessa. L'energia sarà la stessa per tutti i simboli dato che l'ampiezza è costante.

2.4.1 BPSK

Nella BPSK è possibile trasmettere solo due simboli (0 ed 1) perché M=2. Se trasmetto 1 avrò una cosinusoide con fase 90°, se trasmetto 0 avrò una cosinusoide con fase 0°. La costellazione sarà la seguente:

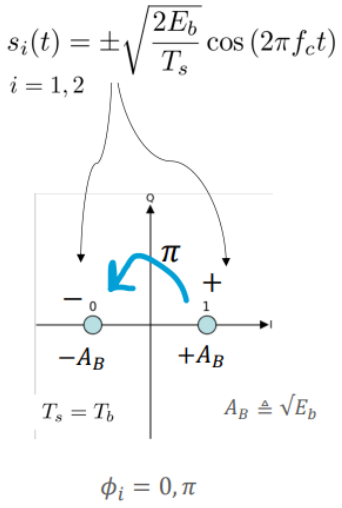
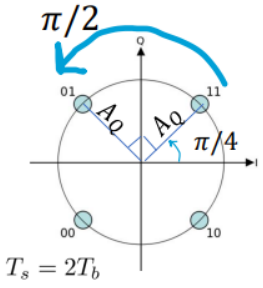


Figura 2.13: BPSK

2.4.2 QPSK

Nella QPSK è possibile trasmettere al massimo quattro simboli (ogni simbolo porterà 2 bit) perché M=4. Essa è costituita da 2 BPSK in quadratura. Con la QPSK si presentano dei problemi: aumento dello spettro e del rumore.

$$s_i(t) = \sqrt{\frac{2E_s}{T_s}} \cos\left(2\pi f_c t + (2i-1)\frac{\pi}{4}\right)$$
$$i = 1, \ldots, 4$$



$T_s = 2T_b$ $\quad A_Q \triangleq \sqrt{E_s} \equiv \sqrt{2E_b}$

$$\phi_i = \frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}$$

Figura 2.14: QPSK

2.5 Power Efficency

É il più basso SNR (Signal-to-Noise Ratio) per bit tale da garantire un preciso BER (Bit Error Rate).

2.6 Spectral Efficiency

É il numero massimo di bit che possono essere trasmessi al secondo mantenendo la QoS desiderata.

2.7 Link Adaptation

É l'adattamento del collegamento quando si cambia l'ordine della modulazione (M).

3 IoT Wireless Technology

Si possono distinguere due tipologie di frequenze:

- *Licensed Spectrum*: non si può usare liberamente se non pagando (perché posseduto da compagnie);
- *Unlicensed Spectrum*: si può usare liberamente.

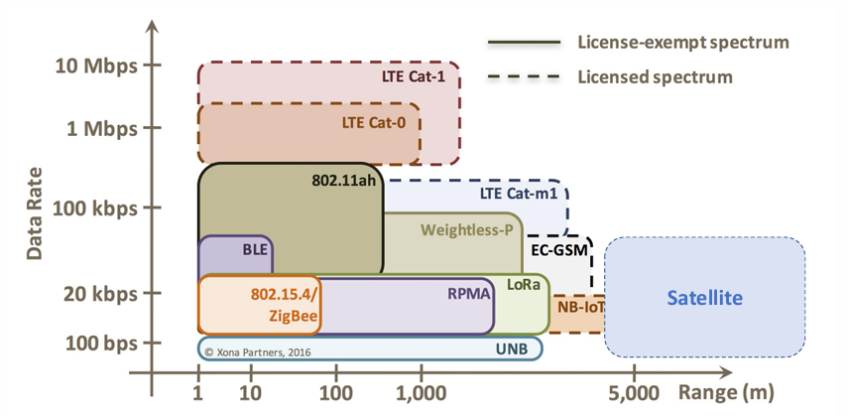


Figura 3.1: Alcune Tecnologie Wireless attualmente disponibili

3.1 Low-Power Wide-Area Network (LPWAN)

L'obiettivo principale delle LPWANs è avere un ampio range e basso consumo di energia, ma ciò comporta un basso data-rate ed un elevata latenza che rende le LPWANs non adatte a tutte le applicazioni.

Essa si concentra su device che inviano pochi Kb alla volta in modo non frequente, quindi, come banda viene utilizzata la *sub-1GHz* perché è meno affollata della 2.4 (utilizzata dal wi-fi tradizionale).

3.2 LPWAN Standards

3.2.1 3GPP (3rd Generation Partnership Project)

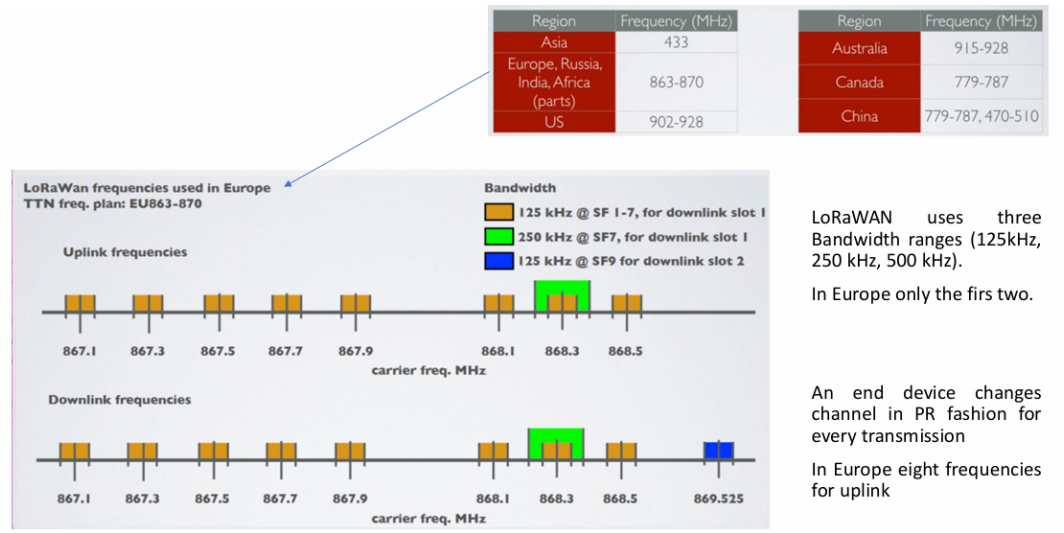
v·t·e [12][13]	LTE Cat 1	LTE Cat 1 bis	LTE-M				NB-IoT		EC-GSM-IoT
			LC-LTE/MTCe	eMTC					
			LTE Cat 0	LTE Cat M1	LTE Cat M2	non-BL	LTE Cat NB1	LTE Cat NB2	
3GPP Release	Release 8	Release 13	Release 12	Release 13	Release 14	Release 14	Release 13	Release 14	Release 13
Downlink Peak Rate	10 Mbit/s	10 Mbit/s	1 Mbit/s	1 Mbit/s	~4 Mbit/s	~4 Mbit/s	26 kbit/s	127 kbit/s	474 kbit/s (EDGE) 2 Mbit/s (EGPRS2B)
Uplink Peak Rate	5 Mbit/s	5 Mbit/s	1 Mbit/s	1 Mbit/s	~7 Mbit/s	~7 Mbit/s	66 kbit/s (multi-tone) 16.9 kbit/s (single-tone)	159 kbit/s	474 kbit/s (EDGE) 2 Mbit/s (EGPRS2B)
Latency	50–100 ms		not deployed	10–15 ms			1.6–10 s		700 ms – 2 s
Number of Antennas	2	1	1	1	1	1	1	1	1–2
Duplex Mode	Full Duplex		Full or Half Duplex	Full or Half Duplex	Full or Half Duplex	Full or Half Duplex	Half Duplex	Half Duplex	Half Duplex
Device Receive Bandwidth	1.4–20 MHz		1.4–20 MHz	1.4 MHz	5 MHz	5 MHz	180 kHz	180 kHz	200 kHz
Receiver Chains	2 (MIMO)		1 (SISO)	1 (SISO)	1 (SISO)	1 (SISO)	1 (SISO)	1 (SISO)	1–2
Device Transmit Power	23 dBm	23 dBm	23 dBm	20 / 23 dBm	20 / 23 dBm	20 / 23 dBm	20 / 23 dBm	14 / 20 / 23 dBm	23 / 33 dBm

4 LoRa

É una tecnologia wireless a bassa potenza progettata per la comunicazione a lunga distanza (10Km-15Km) tra i dispositivi, con consumo energetico ridotto.

4.1 Frequenze LoRaWAN

La scelta delle frequenze dipende dall'ente di regolamentazione e dalla regione considerata.



Le parti dello spettro che sono libere (chiamate *white*) rendono più efficiente il passaggio dalla comunicazione analogica a quella digitale, nel broadcast delle comunicazioni televisive.

4.2 LoRa Modulation

LoRa utilizza una modulazione CSS (Chirp Spread Spectrum), in cui la frequenza non viene cambiata in modo discreto ma viene cambiata continuamente.

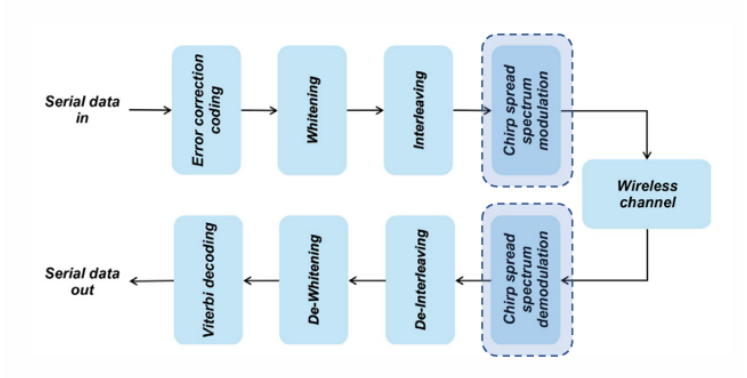


Figura 4.1: Architettura LoRa physical layer (PHY)

Esistono due tipi di *chirp*:

- *chirp-up*: la frequenza parte bassa ed aumenta linearmente;
- *chirp-down*: la frequenza parte elevata e poi decresce.

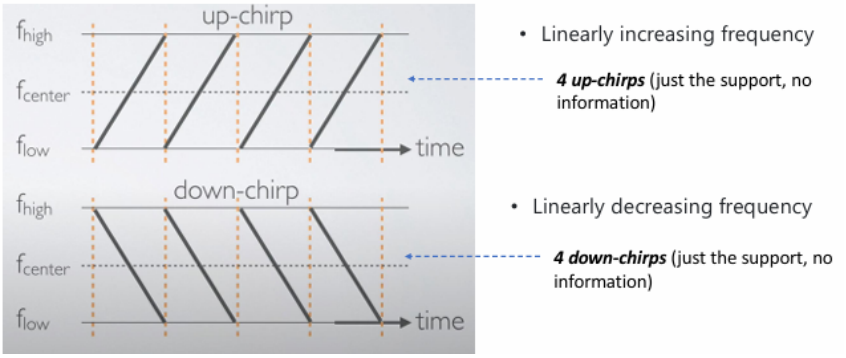


Figura 4.2: Esempio tipi di chirps per segnali non modulati

Un Chirp modulato parte da un punto a caso ed aumenta la frequenza fino a quando non raggiunge il massimo e poi scende bruscamente fino alla frequenza minima, per poi risalire con la stessa velocità. Impostando diverse frequenze di partenza è possibile assegnare a ciascuna un simbolo (0 o 1).

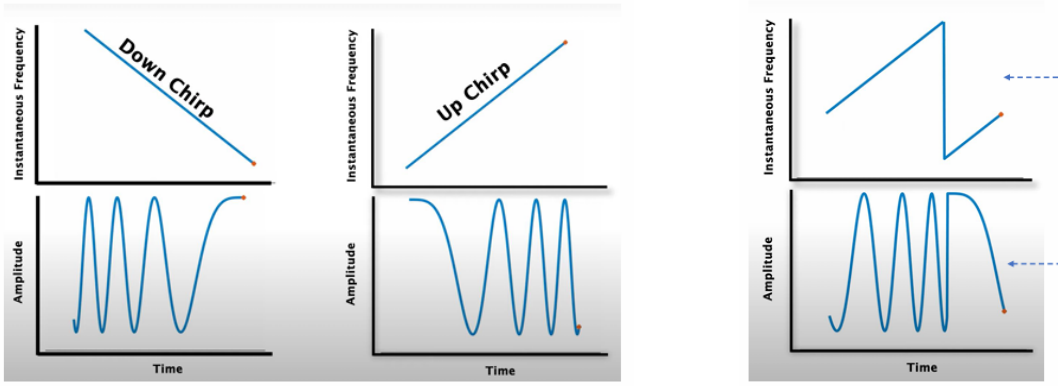
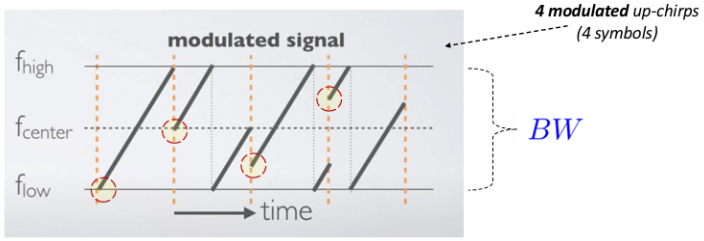


Figura 4.3: Segnale non modulato (sopra), segnale modulato (sotto)

Impostando la frequenza minima e massima oltre il quale il chirp non può andare è possibile controllare la larghezza di banda del segnale.

$$BW = f_{high} - f_{low}$$



In LoRa viene detto chirp ogni generico segnale di trasmissione. Il numero di simboli **M** che possiamo avere per ogni chirp in LoRa è:

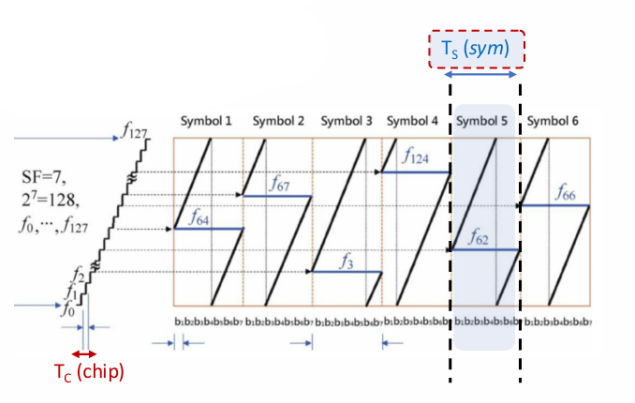
$$M = 2^{SF}$$

Il più piccolo intervallo che compone il set di frequenze completo è detto **chip**. M, quindi, rappresenta anche il numero di chips che compone ogni simbolo.

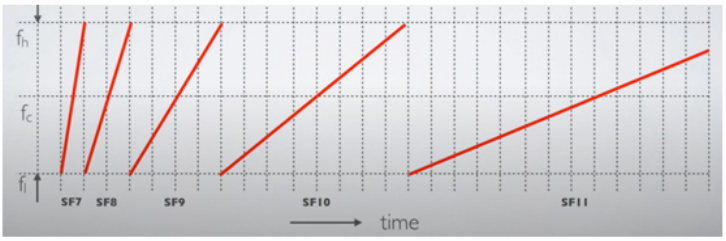
SF (Spreading Factor) rappresenta il numero di bit che possono essere codificati tramite ogni simbolo (spesso indicato nelle varie modulazioni con N).

Il chip-rate in LoRa è: $RC = \frac{1}{T_c} = BW$

La *durata del simbolo* sarà quindi: $T_{sym} = M \cdot T_c = \frac{2^{SF}}{BW}$



Aumentando il numero di bit usati (SF) aumenta anche la durata di trasmissione dei simboli. La durata dei chirp o dei simboli è detto *ToA* (Time on Air) ovvero il tempo che impiega il simbolo quando viene trasmesso per aria.



Le equazioni più importanti in LoRa:

- `bit rate`: $R_b = SF \cdot \frac{BW}{2^{SF}}$
- `symbol rate`: $R_s = \frac{BW}{2^{SF}}$
- `chip rate`: $Rc = Rs \cdot 2^{SF}$

Il **Coding Rate (CR)** è la percentuale di bit propriamente utilizzata dai modem LoRa per proteggere la trasmissione delle informazioni dai bursts e dalle interferenze. Può essere configurato in: 4/5, 4/6, 4/7, 4/8. Dove il 4/5 è il meno protetto, mentre il 4/8 protegge meglio le informazioni ma fa consumare metà della velocità solo per la protezione del canale.

Considerando anche il coding rate otteniamo:

$$R_b = SF\frac{BW}{2^{SF}}CR$$

SF	ToA (ms)
7	112.90
8	195.07
9	349.18
10	616.45
11	1150.98
12	2138.11

Figura 4.4: ToA per un messaggio standard (51 bytes) in millisecondi

Notiamo come quando l'SF aumenta linearmente, il ToA cresce in maniera esponenziale.

4.3 LoRa De-modulation

La demodulazione in LoRa avviene esattamente come per i segnali SS.

In ricezione si moltiplicherà il chirp ricevuto per il suo inverso.

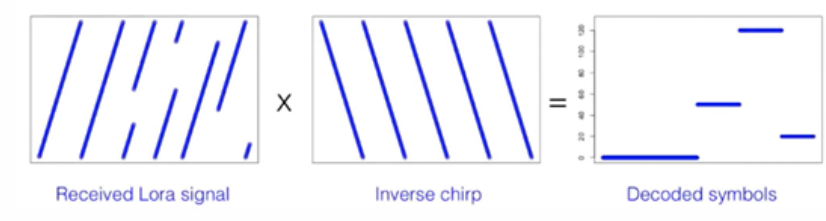


Figura 4.5: Esempio demodulazione (CSS) in LoRa

In caso di presenza di rumore si riuscirà comunque a decodificare il segnale di partenza, perché il rumore interferisce solo per un piccolissimo istante di tempo ad una particolare frequenza.

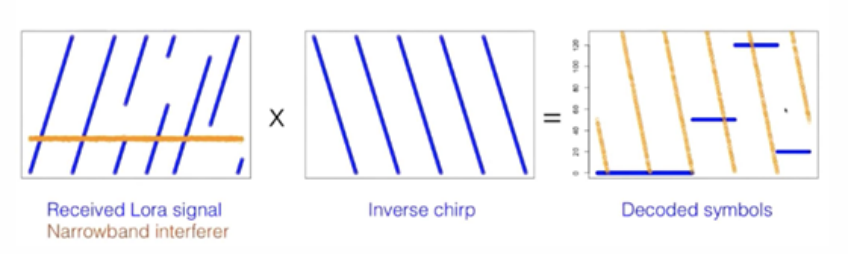


Figura 4.6: Esempio demodulazione in LoRa con presenza di rumore NB

La fase di demodulazione è preceduta da quella di *preamble*, in cui non viene trasmesso nulla, questo permette al ricevitore di sincronizzarsi con il trasmettitore.

4.3.1 Benefici del LoRa (CSS)

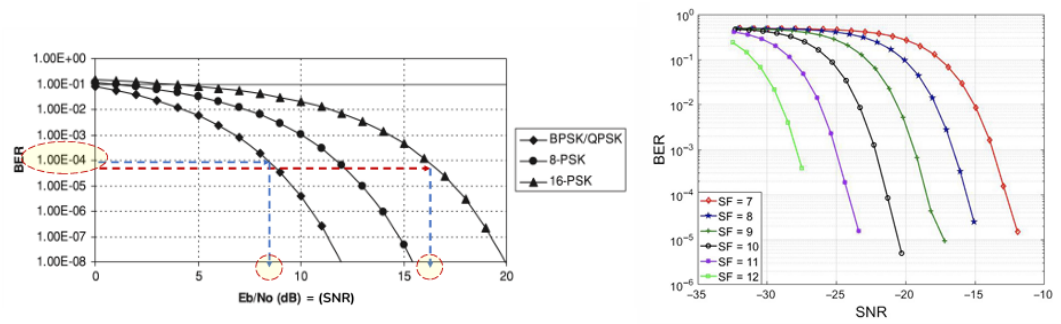


Figura 4.7: BER per la modulazione NB (sinistra), BER per la modulazione LoRa (destra)

Si può notare come nel caso della modulazione LoRa riusciamo ad ottenere valori di SNR negativi, dunque è possibile trasmettere un segnale che ha potenza cento o mille volte più bassa rispetto al rumore, ottenendo un BER ottimale.

4.4 LoRaWAN

LoRaWAN è costituita da end-nodes, gateways, network server ed application server.

4.4.1 Architettura LoRaWAN

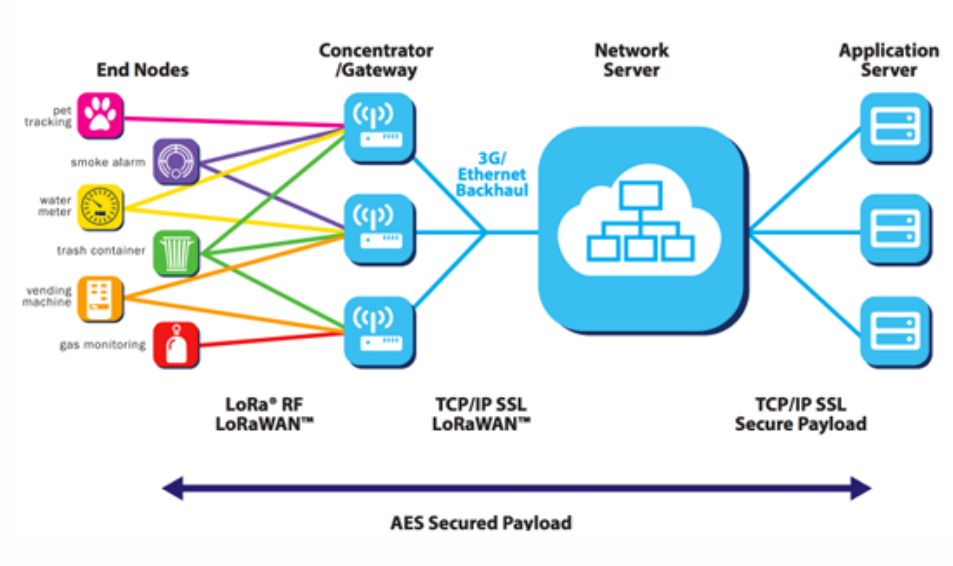


Figura 4.8: Architettura LoRaWAN

Questa architettura è chiamata *Star-of-Stars* perché un singolo end-node è connesso a più gateways (questo viene fatto per migliorare l'affidabilità).

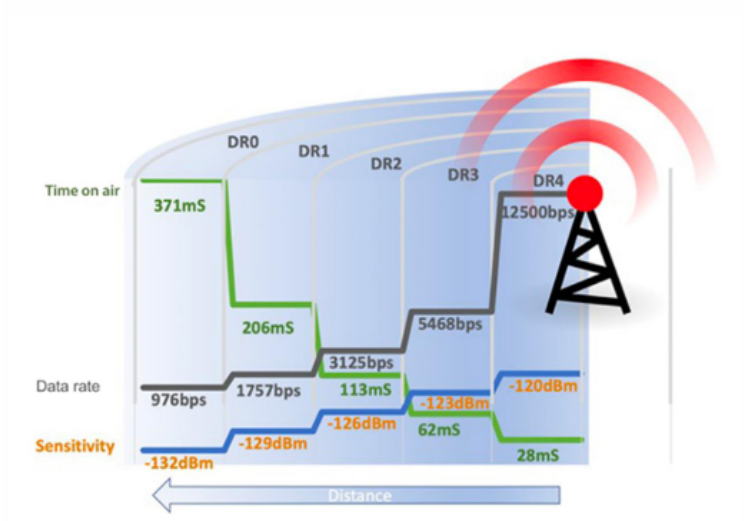


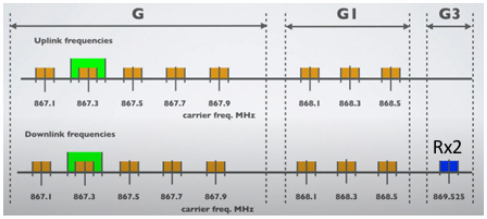
Figura 4.9: Velocità di trasmissione in funzione del range e della durata di trasmissione

Quando il Network Server riceve un dato da più gateways ed ha necessità a rispondere ai devices, trasmetterà la risposta solo ad uno di essi. Questa tecnica è chiamata **Adaptive Data Rate (ADR)**.

Il Network Server può cambiare lo Spreading Factor, la Banda e la TxPower (potenza di trasmissione).

4.4.2 LoRaWAN Channels

Channel	Uplink freq (MHz)	SPREADING FACTOR & BANDWIDTH RANGE
0	868.1	SF7BW125 to SF12BW125
1	868.3	SF7BW125 to SF12BW125 and SF7BW250
2	868.5	SF7BW125 to SF12BW125
3	867.1	SF7BW125 to SF12BW125
4	867.3	SF7BW125 to SF12BW125
5	867.5	SF7BW125 to SF12BW125
6	867.7	SF7BW125 to SF12BW125
7	867.9	SF7BW125 to SF12BW125
8	868.8	FSK
Download freq (MHz)		SPREADING FACTOR & BANDWIDTH RANGE
		Uplink channels 0-8 (RX1)
869.525		SF9BW125 (RX2 downlink only)



8 canali logici possono essere usati sia per il donwlink che per l'uplink.

Name	Band (MHz)	Limitations
G	863 - 870	EIRP<25 mW - duty cycle < 0.1%
G1	868 - 868.6	EIRP<25 mW - duty cycle < 1%
G2	868.7 - 869.2	EIRP<25 mW - duty cycle < 0.1%
G3	869.4 - 869.65	EIRP<500 mW - duty cycle < 10%
G4	869.7 - 870	EIRP<25 mW - duty cycle < 1%

Le reti LoRaWAN devono utilizzare bande di frequenza **ISM**. Queste sono soggette a regolamentazioni relative alla potenza massima di trasmissione e al duty-cycle.

Nell'utilizzo di queste bande ISM non è richiesto alcun costo per la licenza, quindi possono essere usate da chiunque (vantaggio), ma queste hanno un data-rate basso ed è possibile riscontrare interferenze (svantaggio).

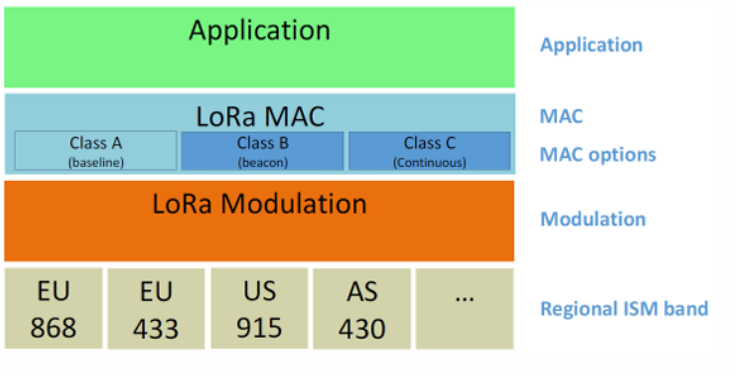


Figura 4.10: Stack protocollare LoRa

4.4.3 Incapsulamento/Decapuslamento tradizionale ISO-OSI

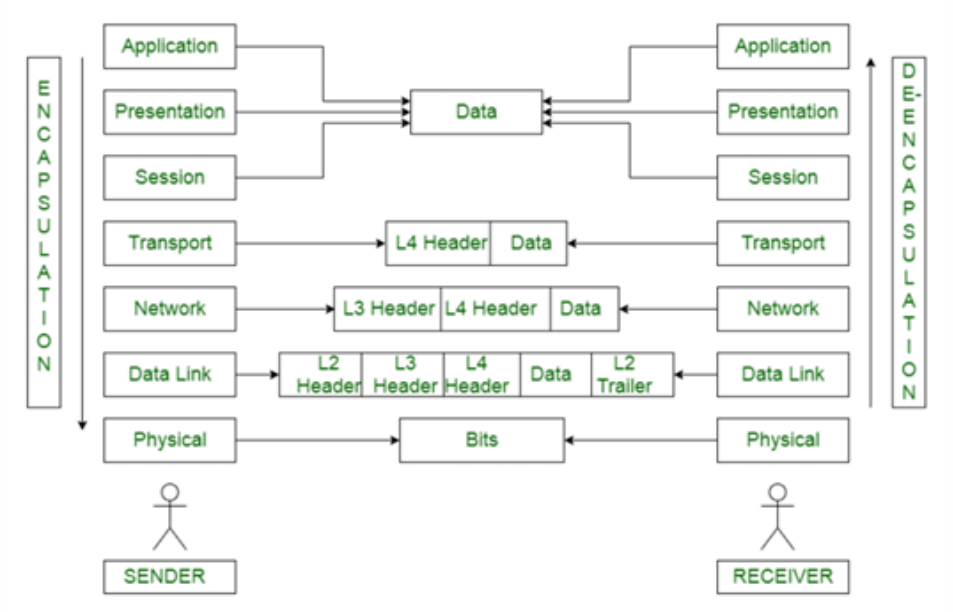


Figura 4.11: Encapsulation e De-encapsulation

Dividiamo la parte degli algoritmi e la parte di design in livelli. Nella figura in basso possiamo notare come, ad esempio, al livello 2 (*Data Link*) il pacchetto prende il nome di *Frames*.

Application	Data
Presentation	Data
Session	Data
Transport	Segments/ Datagrams
Network	Packets
Data Link	Frames
Physical	Bits

Figura 4.12: Protocol Data Unit (PDU)

4.5 Struttura del Frame LoRa

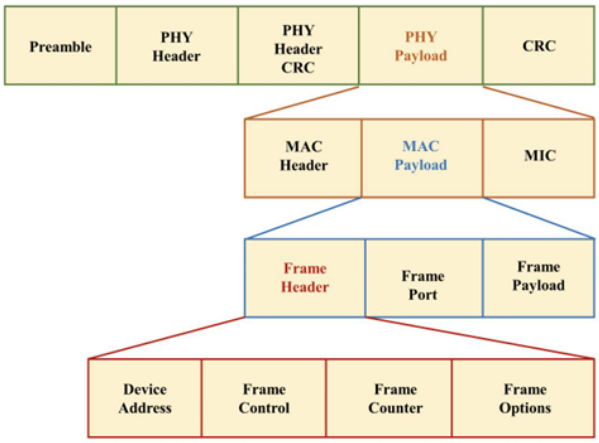


Figura 4.13: LoRa frame structure

Dal punto di vista fisico il Payload è il vero e proprio contenuto informativo, ma dal punto di vista MAC non tutti i bit del livello PHY rappresentano il messaggio utile, bensì contengono sia il MAC Header che il MIC (Message Integrity Control).

Nel CRC sono contenuti i bit di parità necessari a proteggere il pacchetto.

LoRa supporta formati di frame *espliciti* ed *impliciti*.

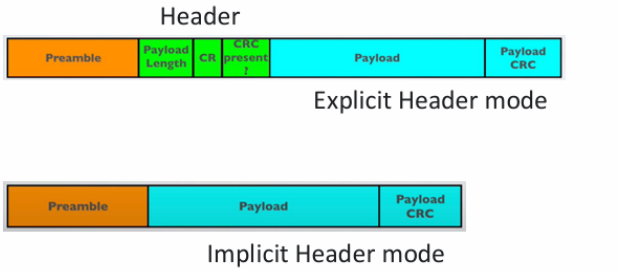


Figura 4.14: LoRa frame formats

Nei frame espliciti è utilizzata la struttura (Preamble, Header, Payload, CRC), mentre nei frame impliciti Header e CRC non sono inclusi (ma possono essere configurati manualmente).

Il formato implicito è più efficiente in termini di *overhead* ma generalmnte LoRaWAN utilizza il formato esplicito perché non è sempre possibile tralasciare l'Header (che contiene informazioni utili sul pacchetto).

Con overhead si intendono tutti i bit di controllo che vengono trasmessi pur non essendo dati reali.

4.6 Classi LoRa

Il livello 2 dello stack protocollare LoRa permette ai dispositivi di comunicare tra di loro. In questo livello troviamo le classi che indicano i diversi modi in cui i dispositivi possono comunicare.

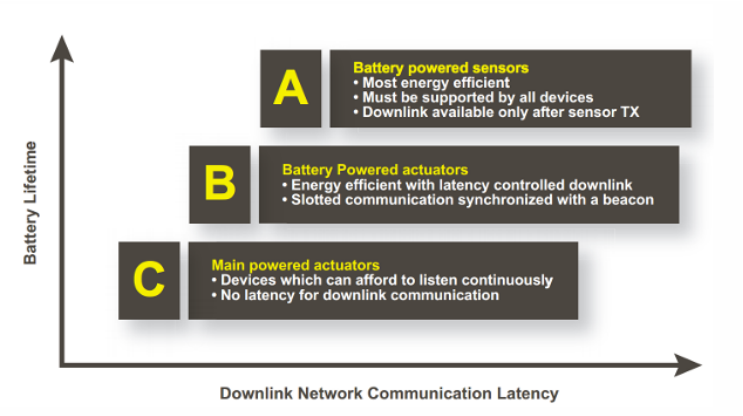


Figura 4.15: LoRa classes

4.6.1 Classe A

I dispositivi quando hanno bisogno di comunicare con il gateway o con il Network Server scelgono un canale (quindi uno Spreading Factor ed una frequenza) ed iniziano a trasmettere. Inizia quindi una fase di uplink dopo la quale il dispositivo aspetta la risposta (fase di downlink) che può avvenire in due finestre differenti.

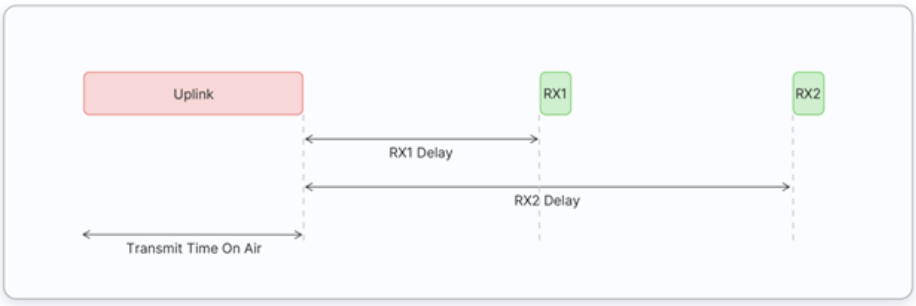


Figura 4.16: Class A

Per i dispositivi di classe A non è prevista una doppia ricezione, dopo una quantità di tempo in cui il dispositivo aspetta una risposta e questa non arriva sullo stesso canale usato in uplink, la prima finestra viene chiusa ed il dispositivo aspetta per un nuovo periodo di tempo (tipicamente il doppio) una risposta sulla seconda finestra (ossia in un altro canale con differente SF e BW). La risposta quindi non si potrà avere sulle due finestre contemporaneamente ma solo su una o sull'altra.

4.6.2 Classe B

Un dispositivo di classe B ha a disposizione nuovi slot (chiamati *Ping Slot*), rispetto a quelli già visti nella classe A.

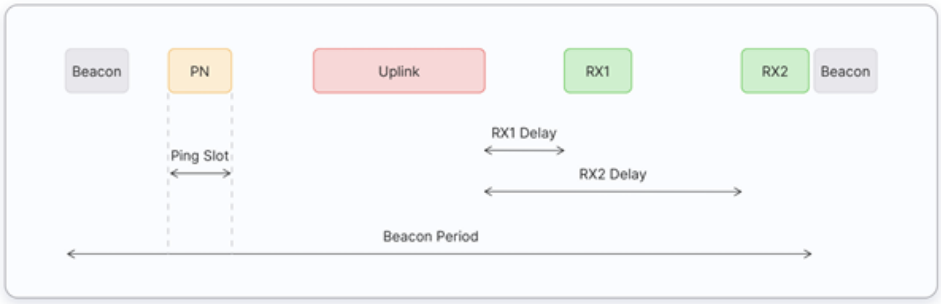


Figura 4.17: Class B

Il dispositivo di classe B ha quindi a disposizione più finestre per ricevere dati (nel caso in cui le due finestre disponibili per i dispositivi di classe A non siano sufficienti).

La programmazione delle risorse viene effettuata dal gateway tramite pacchetti di tipo **beacon** (pacchetti di downlink) che servono a comunicare ai dispositivi di classe B quando attivarsi per poter ricevere pacchetti.

Rispetto ai dispositivi di classe A, quindi, il consumo di potenza aumenta perché il dispositivo deve attivarsi più volte per la ricezione.

4.6.3 Classe C

I dispositivi di classe C sono attivi continuamente per questo motivo non necessitano di beacon per l'organizzazione degli slot.

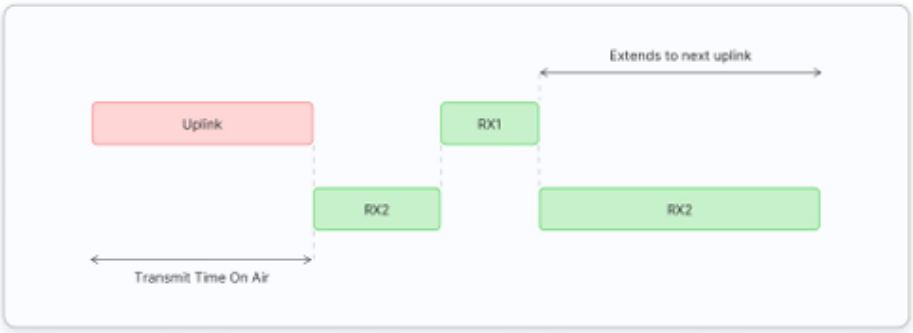


Figura 4.18: Class C

Questo tipo di procedura minimizza le latenze ma ha un consumo elevato di potenza.

4.7 Gestione delle Collisioni in LoRaWAN

In LoRaWAN la fase di accesso multiplo al mezzo non è molto sofisticata e generalmente si creano molte collisioni quando più dispositivi devono trasmettere dati.

LoRaWAN è molto simile ad ALOHA perché i dispositivi ed i gateway possono trasmettere in ogni momento, non esistono meccanismi LBT (Listen-Before-Talk) o CSMA, ma contrariamente ad ALOHA si hanno pacchetti con lunghezza variabile.

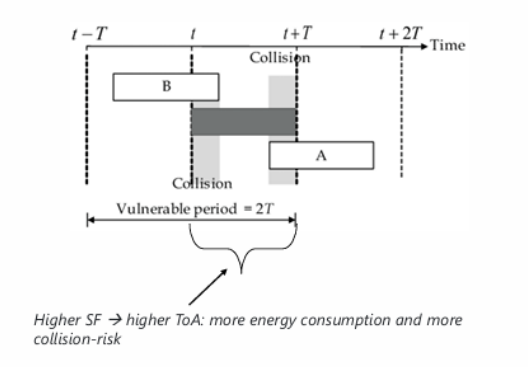


Figura 4.19: LoRaWAN collision

Il periodo di tempo in cui un pacchetto è vulnerabile alle collisioni è chiamato *vulnerable period*. In LoRaWAN questo è pari a 2T, dove T è il ToA del pacchetto.

LoRa riesce a demodulare facilmente pacchetti trasmessi sulla stessa frequenza ma con SF molto diversi. Se due pacchetti hanno uno SF molto simile, invece, possono collidere. Potrebbe accadere però che, avendo due pacchetti con lo stesso SF e la stessa frequenza, uno dei due viene demodulato correttamente, questo sarà quello trasmesso a maggiore potenza (*capture-effect*).

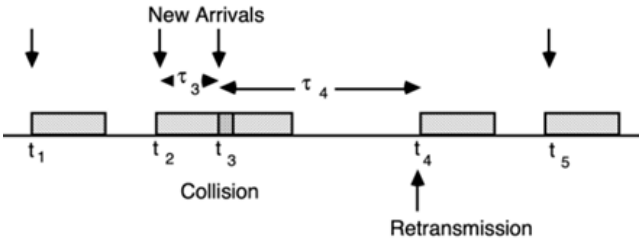


Figura 4.20: Pure ALOHA

Il protocollo con la gestione delle collisioni più simile a quella di LoRaWAN è il Pure ALOHA (*pure* sta ad indicare che non necessita la sincronizzazione, a differenza del protocollo ALOHA *slotted*). Non vengono utilizzati meccanismi di sincronizzazione perché con LoRaWAN vogliamo coprire una vasta area e

la sincronizzazione è preferibile usarla quando l'aria da coprire non è vasta e si ha un tempo di propagazione molto piccolo.

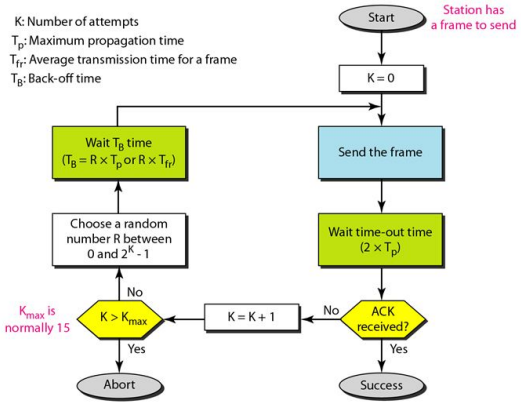
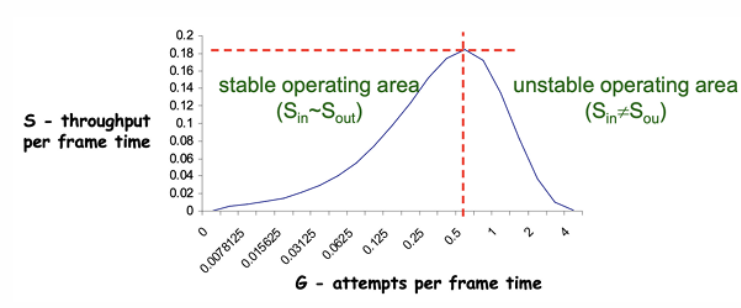


Figura 4.21: Pure ALOHA

Se dopo $2T_p$ (con T_p ritardo di propagazione massimo) non si è ricevuto nessun ACK si è verificata una collisione, altrimenti la trasmissione ha avuto successo. Quando si verifica una collisione si incrementa il contatore k e lo si confronta con una soglia. Si fissa una soglia k_{max} per evitare troppe ritrasmissioni dello stesso pacchetto. Una volta che il contatore raggiunge la soglia massima il pacchetto sarà perso e mai più trasmesso, se la soglia invece non è stata superata ($k < k_{max}$) si riorganizza la trasmissione in un istante casuale (back-off time), scegliendo un numero R (compreso tra 0 e $2^k - 1$).



Il throughput sarà:

$$S = G \cdot P_o = G(1-p)^{n-1} = G(1 - \frac{2G}{n})^{n-1} \approx Ge^{-2G}$$

Dove:

- G è il numero medio di tentativi di trasmissione durante il T_{air};
- P_o è la probabilità di successo (non c'è alcuna trasmissione durante il tempo di vulnerabilità);
- $(1-p)$ indica la probabilità che una stazione non trasmetta nulla,

- $(1-p)^{n-1}$ considera tutte le restanti $n-1$ stazioni;
- n è il numero di dispositivi che tentano di trasmettere pacchetti;
- $2G = np$ questo è dovuto al tempo di vulnerabilità (pari a 2T).

Dal grafico di S notiamo la poca efficienza di ALOHA e LoRa nella gestione degli accessi multipli. Quando G oltrepassa il punto massimo (0.5) il throughput peggiora sensibilmente.

$$G = N \times p_i \times \lambda_i \times T_{air}$$

Dove $\lambda_i \times T_{air}$ è il numero di frame generati nel ToA, mentre N è il numero di dispositivi che tanta di trasmettere dati in uno specifico SF.

Se sostituiamo G nella formula di ALOHA, si ottengono le prestazioni di LoRa:

$$S \approx Ge^{-2G} = (N \times p_i \times \lambda_i \times T_{air})e^{-2(N \times p_i \times \lambda_i \times T_{air})}$$

Considerando adesso il capture-effect, quando il Signal-to-Interference Noise Ratio (SINR) supera una soglia di threshold (Th) possiamo ricevere almeno un pacchetto.

$$SINR_X = \frac{P_X}{\sum P_I + \sigma^2} > Th$$

Dove:

- P_X è la potenza del segnale
- $\sum P_I$ rappresenta l'interferenza generata dalle collisioni
- σ^2 rappresenta il rumore
- Th è la soglia minima di SINR richiesto per decodificare il segnale X

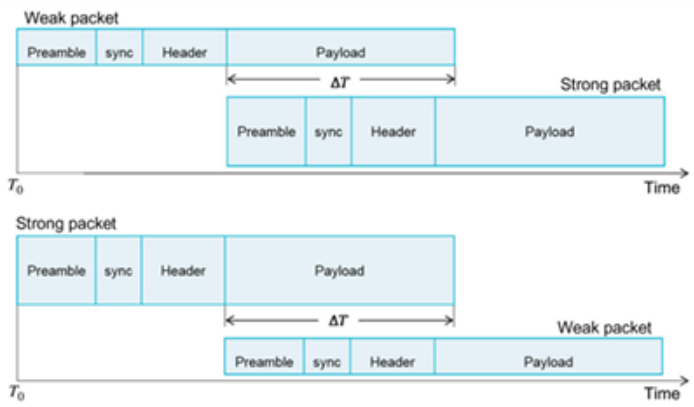


Figura 4.22: Capture scenario → (top): Stronger First; (bottom): Stronger Last

Il capture-effect può essere visualizzato in due modi.

1. Nel primo caso (in alto) un pacchetto debole in potenza (weak packet) viene trasmesso per primo e dopo un tempo ΔT di Payload viene caricato un strong packet.

2. Nel secondo caso (in basso) avviene esattamente il contrario.

4.8 Sicurezza LoRaWan

In LoRa il Payload viene criptato con algoritmi standard (AES128).

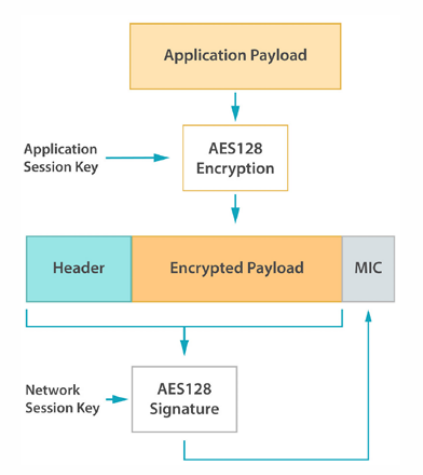


Figura 4.23: LoRaWAN security

5 Bluetooth Low Energy (BLE)

Il BLE è una tecnologia a basso Data Rate (max 3 Mbps), ma superiore rispetto a LoRa, e copre distanze molto brevi (max 10m). Il BLE necessita di una potenza ridotta per funzionare, quindi i dispositivi che integrano questa tecnologia hanno una lunga durata della batteria.

Sono state pubblicate varie versioni di bluetooth con l'obiettivo di aumentare il datarate, la velocità ed il radio range. Il BLE nasce dalla versione 4.0.

É possibile avere tipo di dispositivi supportanti o il BLE o la BR/EDR o entrambi:

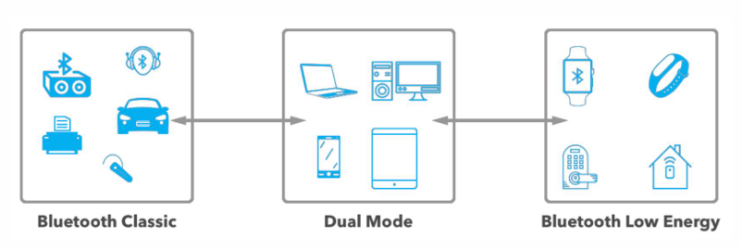


Figura 5.1: Tipi di dispositivi Bluetooth

- *Classic Bluetooth*: ad esempio tra telefono e cuffie, tra pc e stampante o tra pc e pc.
- *Low Energy Bluetooth*: ad esempio i sensori che monitorano l'attivita sportiva ottenendo i dati su di un orologio.
- *Dual Mode*: ad esempio un sensore che monitora il battito cardiaco ed invia i dati ai computer dell'ospedale utilizzando il cellulare come gateway.

5.1 Caratteristiche del BLE

5.1.1 Bande di Frequenza

La frequenza a cui lavora il bluetooth è la stessa del wi-fi (2.4 GHz). Il BLE a differenza del bluetooth classico utilizza solamente 40 canali, di questi solo 3 sono impiegati per il paring (advertising channel). Il BR/EDR utilizza invece 79 canali di hopping per garantire la connessione tra due dispositivi,

5.1.2 Tecnologia *Mostly-Off*

Tramite questa tecnologia il dispositivo resta spento per la maggior parte del tempo e si accende solo per trasmettere i dati.

5.1.3 Connessioni più veloci

Il bluetooth classico impiega 32 canali dedicati ed esegue la procedura di paring in 20ms, il BLE la riesce ad eseguire in 3ms.

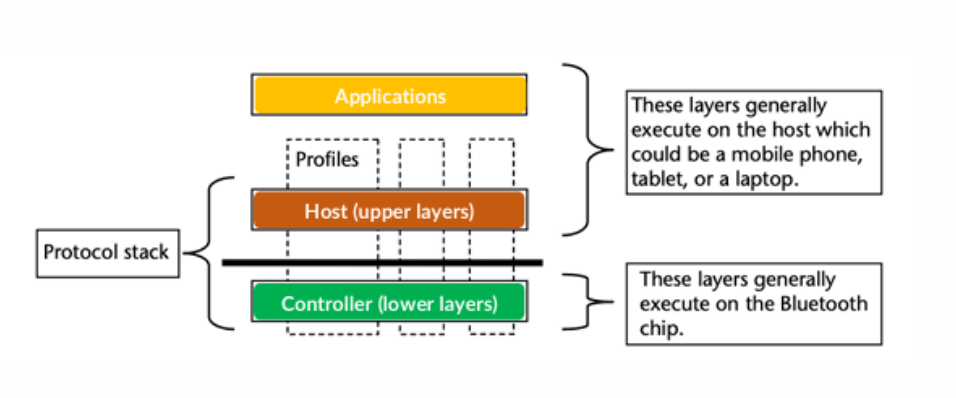
5.1.4 Pacchetti di dati più corti

Questo si traduce in header più corti e payload minori da trasmettere.
Vengono trasmessi/ricevuti meno bit.

5.1.5 Funzionalità ridotte

- **BLE**: non permette la trasmissione di segnali vocali; può avere qualsiasi numero di dispositivi in una piconet (Scatternet); non supporta il cambio di ruolo MASTER-SLAVE

5.2 Physical Layer



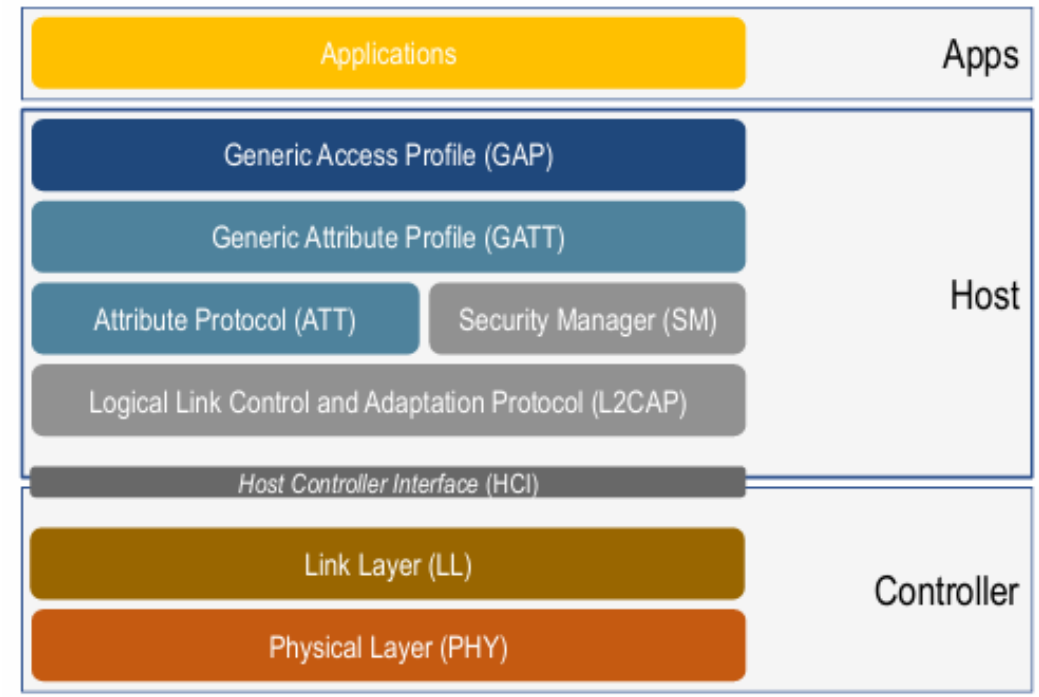


Figura 5.2: Stack architecture

Il BLE utilizza due schemi ad accesso multiplo:

- **FDMA** (Frequency Division Multiple Access).

  Questo utilizza 40 physical channels separati a 2MHz. Di questi canali, 3 sono usati come *primary advertising channels* e i restanti 37 sono usati come secondary advertising channels e *data channels*.

- **TDMA** (Time Division Multiple Access)

Viene usato anche un **FHSS** (Frequency Hopping Spread Spectrum) durante la connessione. Due dispositivi che stanno comunicando cambiano casualmente frequenza per scambiarsi i dati. L'FHSS inoltre permette ai dispositivi di evitare canali congestionati e utilizzati da altri dispositivi/tecnologie nell'ambiente circostante, migliorando quindi l'affidabilità.

5.3 Spectrum Usage

Si hanno interferenze dato che la banda utilizzata è la 2.4GHz (la quale essendo libera può essere sfruttata da chiunque).

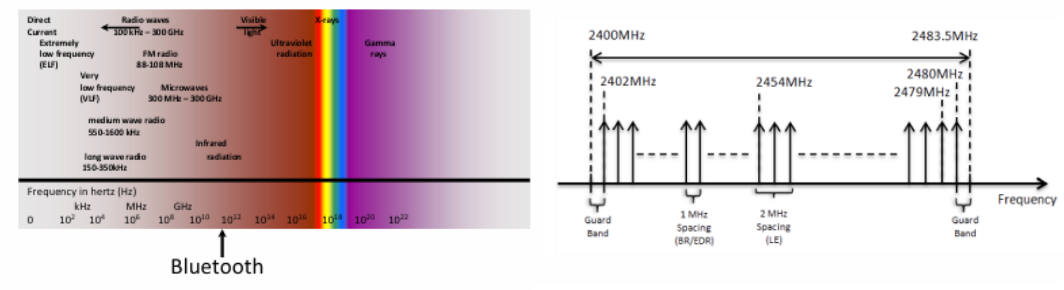


Figura 5.3: Spectrum Usage

La Guard Band viene utilizzata per evitare le interferenze tra i diversi canali.

5.4 Data e Advertising Channels

Come detto i canali radio sono divisi in:

- *Advertising Physical Channel*: utilizzante i canali 0, 12 e 39 per scoprire dispositivi, creare una connessione e fare il broadcast e la ricezione dei dati.

- *Data Physical Channel*: utilizzante i 37 canali restanti per la trasmissione dei dati.

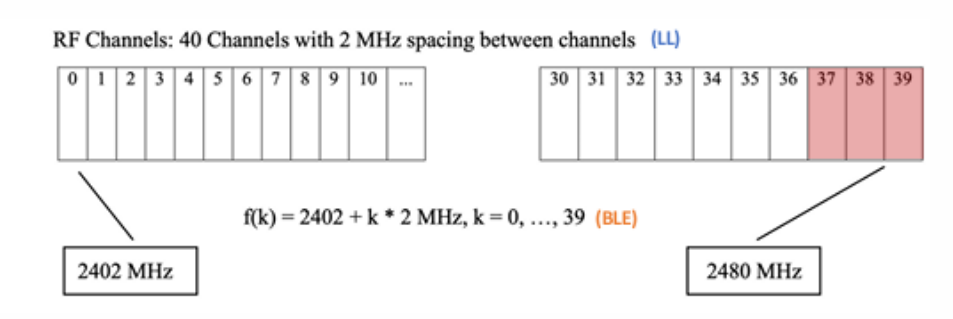


Figura 5.4: Canali logici e formula per calcolare la frequenza della portante

5.5 Bluetooth Modulation

Il BLE utilizza una modulazione GFSK (Gaussian Frequency Shift Keying), in questa modulazione viene trasmesso un bit per simbolo. Per trasmettere il bit 0 viene utilizzata una determinata frequenza mentre per trasmettere l'1 se ne utilizza un'altra.

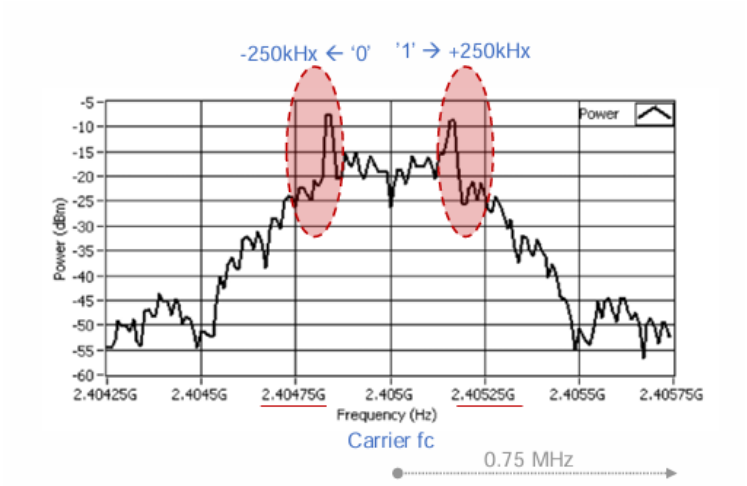


Figura 5.5: Rappresentazione di un segnale modulato GFSK

L'ampiezza totale del canale è 2MHz ma se consideriamo soltanto la banda impiegata per trasmettere il segnale utile, senza considerare le alterazioni del segnale successive a 0.25MHz, la banda diventa: 0.25MHz (bit 1) + 0.25MHz (bit 0) = 0.5MHz.

Il modulation index si calcola come:

$$h = \frac{\Delta f}{f_m} = \frac{\Delta f}{\frac{1}{2T_s}} = 2\Delta f T_s$$

dove Δf: peak frequecy-deviation della portante; f_m: highest frequency presente nel segnale modulante.

h serve a capire se la modulazione utilizzata è una narrow band o una wide band. Se il modulation index è minore di 1 si tratta di una narrow band.

5.6 Bluetooth Link Layer

Questo livello è responsabile del controllo, della negoziazione, della creazione dei collegamenti e della selezione delle frequenze per la trasmissione dei dati.

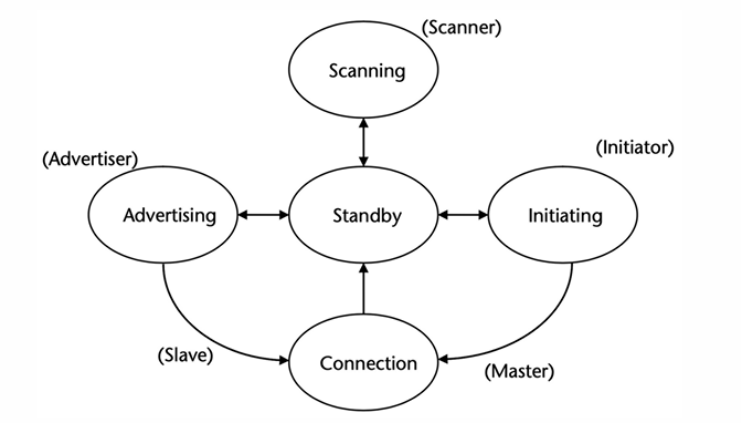


Figura 5.6: Diagramma degli stati e ruoli del LL (Link Layer)

Chiamiamo i dispositivi che trasmettono pacchetti di advertising sul canale PHY → **advertisers**, quelli che ricevono i pacchetti di advertising senza voler connettersi al dispositivo sono denominati **scanners**, quelli che devono connettersi ad un altro dispositivo ascoltando i pacchetti di advertising collegabili vengono detti **initiators**.

Una volta stabilita la connessione l'initiator diventa il dispositivo master ed il dispositivo di advertising diventa il dispositivo slave, in quella che viene definita **piconet**.

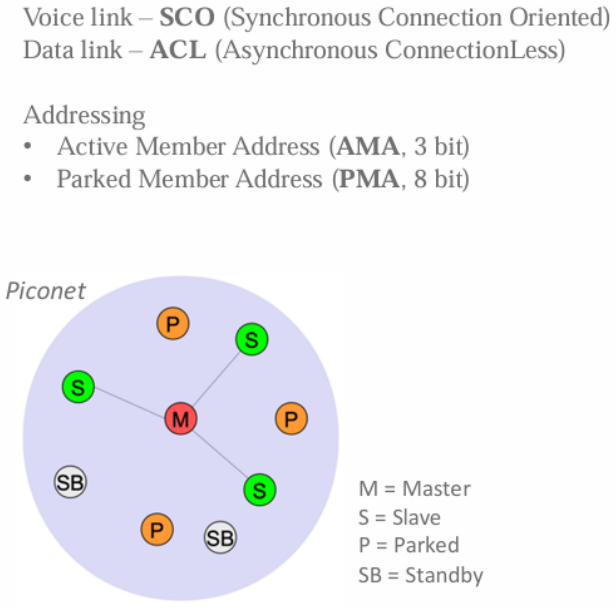


Figura 5.7: Piconet

Nella piconet del bluetooth classico è possibile avere al massimo 7 dispositivi connessi come Slaves, mentre nel BLE non abbiamo questo limite.

É possibile collegare due o più piconet insieme per formare una **scatternet**.

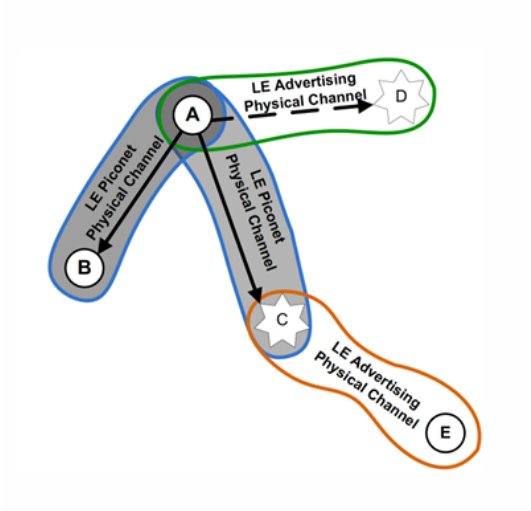


Figura 5.8: Scatternet

5.7 BLE Events

Il canale fisico del BLE è diviso in unità di tempo chiamate *eventi*, che non hanno una durata fissa (a differenza della fixed slot structure del bluetooth classico). I device Low Energy trasmettono pacchetti che sono "posizionati" in questi eventi e per ogni evento si può avere la trasmissione di più pacchetti.

5.7.1 Advertising

Il dispositivoha un pacchetto di advertising che trasmette sul canale. Se c'è un secondo dispositivo che fa da scanner sul canale in ascolto esso ascolterà il pacchetto e risponderà con una richiesta per avere più informazioni. La chiusura dell'evento di advertisement può avvenire in due modi:

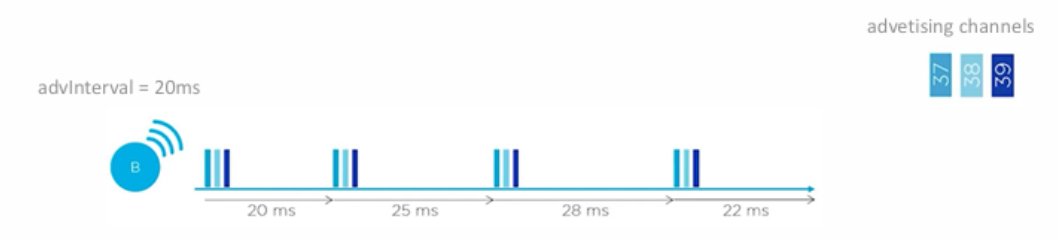
1. non riceviamo alcuna risposta (perché non ci sono dispositivi nell'area di advertisement o perché nessun dispositivo in quel momento è in grado di rispondere)

2. esiste almeno un dispositivo che risponde con una request, l'advertiser risponderà con un altro pacchetto contenente informazioni più dettagliate e l'evento si chiuderà

I pacchetti di advertisement vengono trasmessi sui canali logici: 37, 38, 39. Si supponga che lo scanner sia in ascolto esclusivamente sul canale 38 e che un advertiser (in particolare un sensore) comunichi di essere un termometro. Il sensore trasmette prima il pacchetto di advertising sul canale 37, al quale non riceve risposta. Dopo un certo intervallo di tempo trasmette il pacchetto al canal 38, lo scanner decidere di richiedere informazioni aggiuntive ed il sensore

risponde con la temperatura misurata. Passato un altro intervallo di tempo il pacchetto viene trasmesso sul canale 39, sul quale non riceve risposta.

Al termine di questa rapida successione, il sensore si ferma nuovamente per un advertising interval pari a circa 200ms. Solitamente l'intervallo di tempo che intercorre tra la trasmissione del pacchetto di advertising su due canali successivi è compreso tra i 300ms ed i 400ms.



Nella figura la B è il broadcaster, le tre linee verticali di diversa gradazione di blu sono gli advertising events che rappresentan diversi cicli di advertising.

Il tempo tra due cicli di advertising consecutivi è dato dalla formula:

$$T_{advEvent} = advInterval + advDelay$$

Esistono sette tipologie di advertising events:

1. **Connectable and Scannable Undirected Event**. Questo tipo consente agli altri dispositivi di ricevere i pacchetti di advertisement, inviare una richiesta di scansione all'advertiser e stabilire una connessione con esso.

2. **Connectable Undirected Event**. Questo tipo consente agli altri dispositivi di ricevere i pacchetti di advertisement e stabilire una connessione con l'advertiser.

3. **Connectable Directed Event**. Questo tipo consente ad un dispositivo specifico di ricevere i pacchetti di advertisement e stabilire una connessione con l'advertiser.

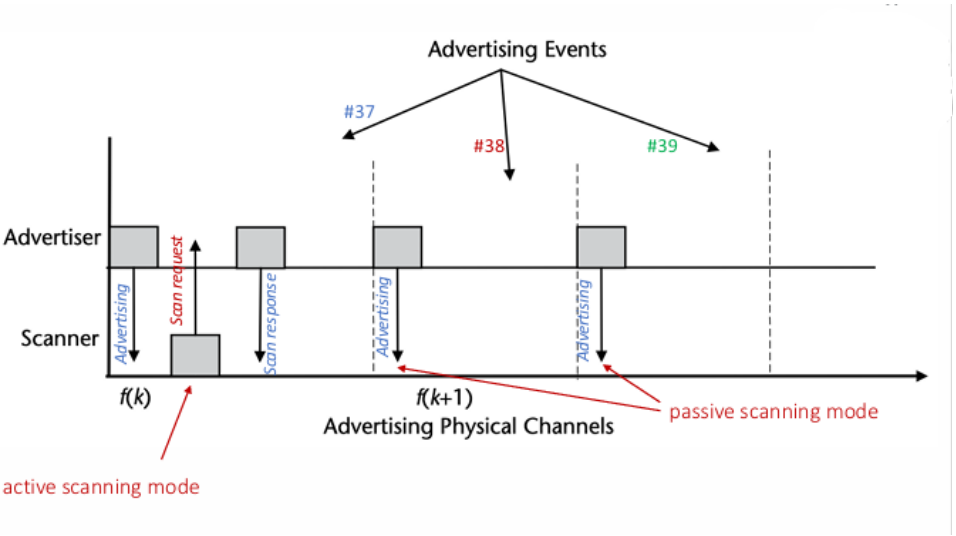
4. **Non-Connectable and Non-Scannable Undirected Event**. Questo dispositivo consente ad altri dispositivi di ricevere i pacchetti di advertisment. Tuttavia, non consente di effettuare richieste di scansione o di stabilire una connessione con l'advertiser. [Ha metà dell'elettronica rispetto ad altri devices perché prevede solo le operazioni di invio].

5. **Non-Connectable and Non-Scannable Directed Event**. Questo dispositivo consente ad un dispositivo specifico di ricevere i pacchetti di advertisment. Tuttavia, non consente di effettuare richieste di scansione o di stabilire una connessione con l'advertiser. [Ha metà dell'elettronica rispetto ad altri devices perché prevede solo le operazioni di invio].

6. **Scannable Undirected Event**. Questo dispositivo consente ad altri dispositivi di inviare richieste di scansione all'advertiser per ricevere pacchetti di advertisement aggiuntivi. Sono i devices che accettano di essere scansionati, cioè accettano richieste per fornire pù informazioni ma non accettano connection request. Più richieste un advertiser accetta, più potente è in termini di capacità di elaborazione.

7. **Scannable Directed Event**. Questo dispositivo consente ad un dispositivo specifico di inviare richieste di scansione all'advertiser per ricevere pacchetti di advertisement aggiuntivi. Sono i devices che accettano di essere scansionati, cioè accettano richieste per fornire più informazioni ma non accettano connection request. Più richieste un advertiser accetta, più potente è in termini di capacità di elaborazione.

5.7.2 Scanning



Nella prima parte del grafico viene inviato un advertising packet al quale uno scanner risponde con una request. In seguito, l'advertiser risponde con un pacchetto contenente più informazioni e poi l'evento si chiude.

La frequenza $f(k)$ è la stessa durante tutto l'evento per il BLE, solo quando inizia un secondo evento la frequenza cambia, in questo caso spostandosi dal canale 37 al 38. Il secondo evento però si chiude prima perché non riceve risposta. Le ragioni per cui lo scanner non risponde nei canali 38 e 39 sono:

- lo scanner non vuole rispondere perché non ha nulla da dire;
- lo scanner non può rispondere perché il pacchetto di advertising non è scansionabile;
- lo scanner non può rispondere perché si trova in modalità passiva;
- lo scanner non può rispondere perché essendo un dispositivo IoT economico ed a basso impatto energetico non è provvisto di un trasmettitore.

[Si parla di active scanner se il dispositivo è provvisto sia di trasmettitore che di ricevitore, e di passive scanner se è provvisto esclusivamente dal ricevitore].

I principali parametri che determinano l'efficienza, in termini di energia, di uno scanner sono:

- **Scan type**: attivo se lo scanner risponde con una richiesta, passivo altrimenti.
- **Scan window**: indica per quanto tempo lo scanner resta in ascolto sul canale.
- **Scan interval**: indica la frequenza con la quale lo scanner ascolta il canale.

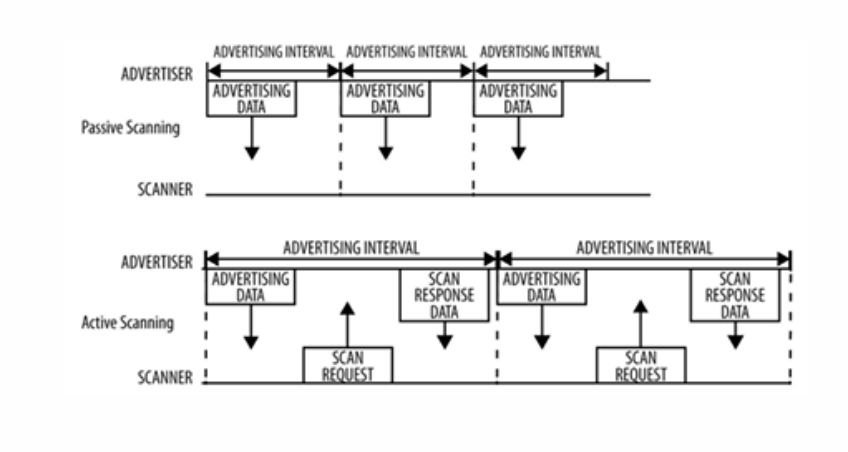


Figura 5.9: Differenza tra Passive scanning ed Active scanning

[Gli intervalli di advertising sono più ampi nell'active scanning (dato che lo scanner può rispondere al pacchetto di advertising effettuando una richiesta di scan)]. Notiamo dalla Figura 5.10 la correlazione tra duty cycle e la scanning windows. Minore è il duty cycle e maggiore energia viene risparmiata dal dispositivo.

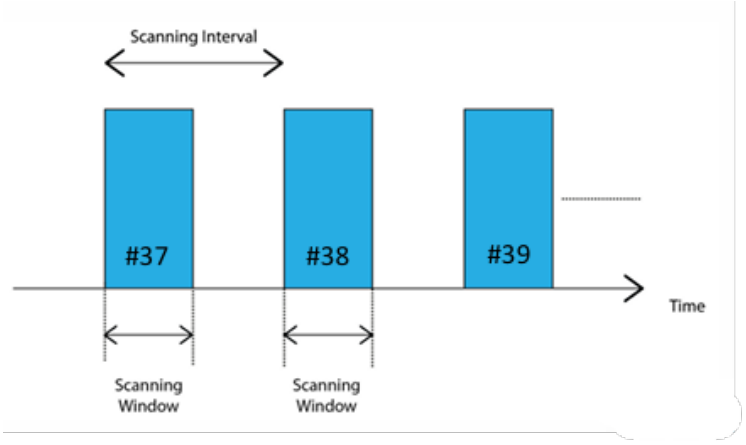


Figura 5.10: Interpretazione grafica della scanning window

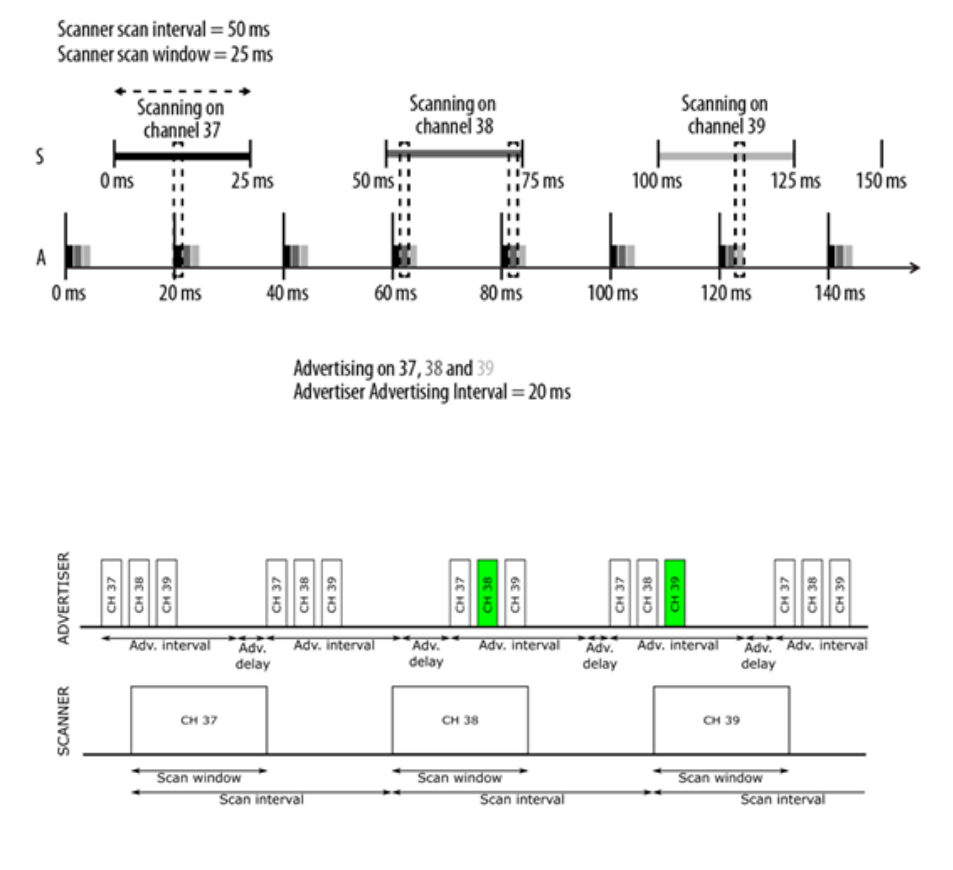


Figura 5.11: Esempi di relazione tra advertising e scanning

Il duty cycle dello scanner è dato da:

$$DC_s = \frac{scanning \quad window}{scaning \quad interval} = \frac{25}{26} = 0.5$$

Il duty cycle dell'advertiser invece si calcola come:

$$DC_a = \frac{tempo \quad advertiser \quad attivo}{tempo \quad complessivo} = \frac{1.2}{20} = 0.06$$

5.7.3 Initiating

É uno stato transazionale che si verifica nel momento in cui un dispositivo di scanning, dopo aver ricevuto un pacchetto di advertisement, può decidere di instaurare una connessione inviando un apposito pacchetto a cui risponderà a sua volta l'advertiser con un pacchetto di conferma per la connessione.

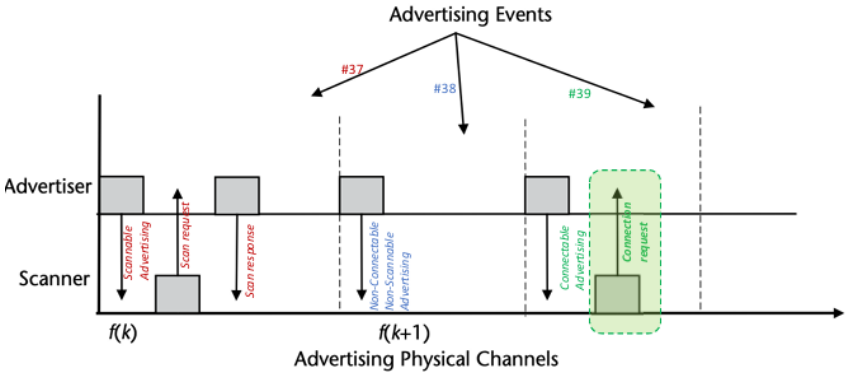


Figura 5.12: Evento di connessione tra advertiser e scanner

L'advertiser trasmette il pacchetto di advertisement sul canale 37 ed attende, ma non riceve alcuna risposta. Passa pertanto a trasmettere il pacchetto sul canale 38 e riceve una connection request packet seguito da un intervallo di tempo obbligatorio di 1.25ms in cui il master e lo slave (scanner ed advertiser) attendono prima di entrare in connessione, in maniera tale da definire i parametri per comunicare in maniera efficace. Successivamente inizia la trasmit window size nella quale il master comunica allo slave quando deve mettersi in ascolto. Una volta che il master trasmette il pacchetto si può considerare stabilita la connessione, lo slave risponde immediatamente ed il connection event si chiude. Se non viene trasmesso nulla lo slave torna in modalità sleep ed attende il successivo intervallo di connessione.

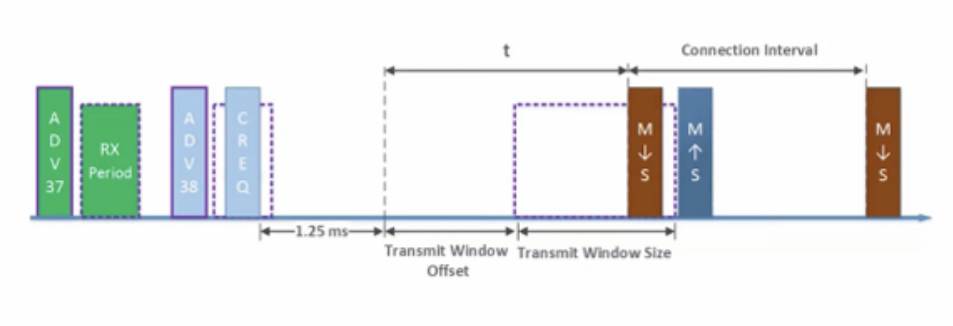


Figura 5.13: Dettaglio sull'evento di connessione tra advertiser e scanner

Il primo connection event è impostato a seguito della connection request *CONNECT_REQ PDU (Protocol Data Unit)*.

Alcuni dei parametri che il master trasmette nella connection request sono:

- *Transmit Window Offset* $\longrightarrow$ il tempo che il dispositivo deve attendere dopo aver ricevuto la CONNECT_REQ;
- *Transmit Window Size* $\longrightarrow$ la dimensione della finestra di trasmissione;
- *Connection Interval* $\longrightarrow$ intervallo tra due connessioni;
- *FH Parameters* $\longrightarrow$ canale da usare durante la trasmissione (channel map e hop increment).

É possibile osservare in Figura 5.14 come la connection request è trasmessa al canale 38, al seguito della quale il master trasmette sul canale #0 e lo slave risponde sullo stesso canale. Dopo ciò il connection event si chiude (se lo slave ha risposto non c'è altro da trasmettere evidentemente). Il successivo connecgtion event si verifica sul canale #5 (hop ad una nuova frequenza) ed ha una durata maggiore perché avvengono tre trasmissioni del master con le relative risposte dello slave. Notiamo come non cambia il connectio interval (massimo intervallo di tempo entro il quale master e slave scambiano pacchetti), si avranno solo cambiamenti nella durata del connection event.

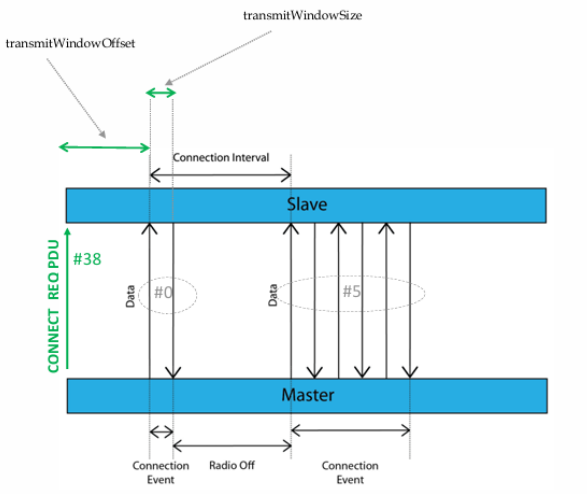


Figura 5.14: Eventi di connessione tra master e slave

5.7.4 Connection (BR/EDR)

La strategia di accesso è una combinazione tra *Frequency Hopping (FH)* e *Time Division Duplex (TDD)*.

- **FH**: la sequenza di hopping spazia su 79 canali, secondo un ordine pseudo casuale dettato dal master e con un intervallo di tempo tra salti pari a $625\mu s$ (1600 hop per secondo). É realizzato slot-by-slot (nel passare da uno slot al successivo i dispositivi cambiano frequenza).
- **TDD**: l'informazione è trasmessa una direzione per volta con la trasmissione che alterna fra due direzioni. Gli slot sono sempre di $625\mu s$ ed il master interroga gli slave.

In Figura 5.15 la comunicazione tra un dispositivo master e tre slave (tutti con lo stesso clock e con la stessa divisione degli slot temporali del master). Questa struttura rigida comporta un dispendio maggiore di energia perché essendo gli slave sincronizzati sanno quando inizia un nuovo slot (anche quelli che non sono di loro interesse) non entrano quindi in modalità sleep.

Il master invia un pacchetto al primo slave in uno slot di durata $625\mu s$ con frequenza $f(2k)$, e questo risponde con un pacchetto grande cinque slot e frequenza $f(2k+1)$. Per il secondo slave si ha, invece, che è il master ad inviare un pacchetto grande 3 slot e lo slave a rispondere in uno slot.

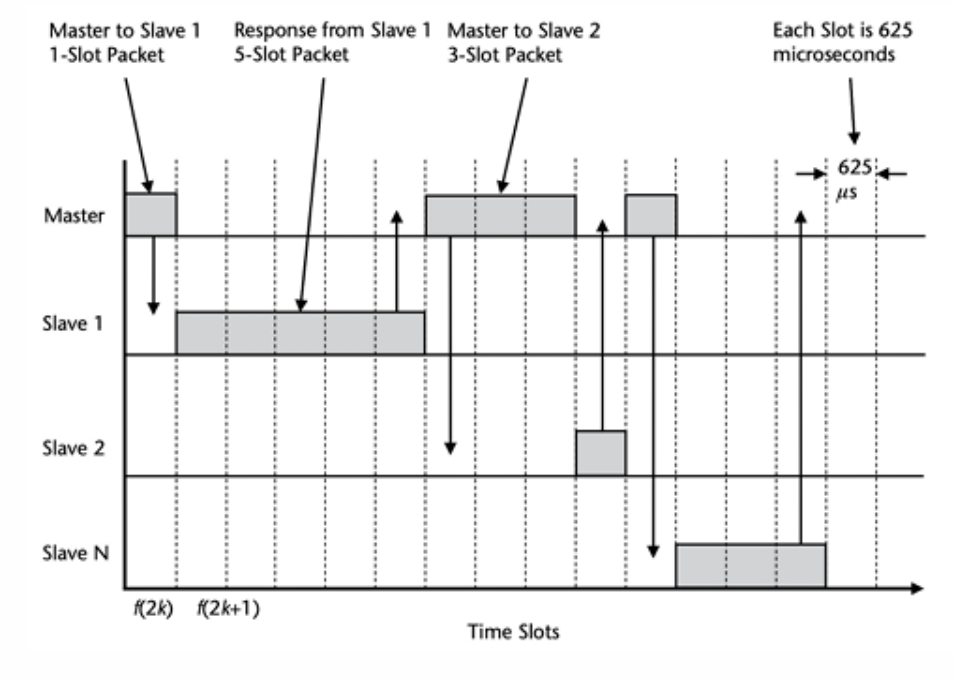


Figura 5.15: Comunicazione in slot tra master e slaves

Si distinguono due possibili comunicazioni nel Bluetooth Classic:

- **SCO (Synchronous Connection Oriented)** $\longrightarrow$ usato per le comunicazioni vocali, prevede pacchetti di lunghezza fissa pari ad uno slot ed il *Forward Error Connection (FEC)* come sistema per la gestione degli errori sul canale (la voce è meno sensibile agli errori rispetto ad altre informazioni).

- **ACL (Asynchronous ConnectionLess)** utilizzato per trasmettere dati, permette la trasmissione di pacchetti più grandi (3 o 5 volte lo slot) e prevede l'uso di *acknowledgment* per la gestione degli errori (il ricevitore, all'insorgere di un errore, chiede la ritrasmissione attraverso un acknowledgment).

La dimensione di un pacchetto è multiplo dispari dello slot perché il protocollo impone che le trasmissioni del master siano concesse negli slot pari, mentre le trasmissioni degli slave in quelli dispari.

Notiamo in Figura 5.16 come la prima riga prevede un trasmissione vocale, pertanto i pacchetti durano soltanto uno slot ciascuno con una sua specifica frequenza. Nella seconda riga il master trasmette un pacchetto di tre slot a frequenza f_k e lo slave risponde in uno slot a frequenza f_{k+3} (il salto di frequenza è coerente con il numero di slot trascorsi). Per la stessa riga possiamo dire lo stesso, a causa della trasmissione di un pacchetto lungo cinque slot a frequenza f_{k+1} si ha un pacchetto successivo a frequenza f_{k+6}.

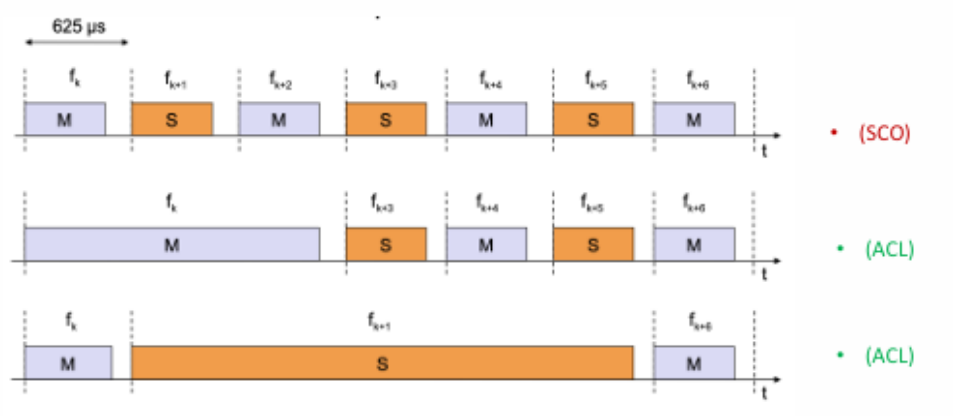


Figura 5.16: Differenza tra comunicazione SCO e ACL

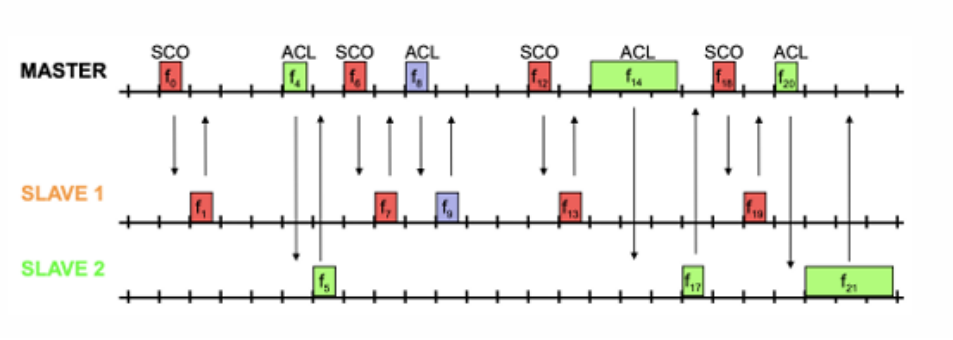


Figura 5.17: Comunicazione sincronizzata tra master e slave

Come mostrato in Figura 5.18 la trasmissione di un pacchetto termina poco prima del corrispettivo slot. Questo periodo di idle è impiegato per la sincronizzazione, in modo che i dispositivi sappiano quando termina una trasmissione e quando sta per partire la successiva.

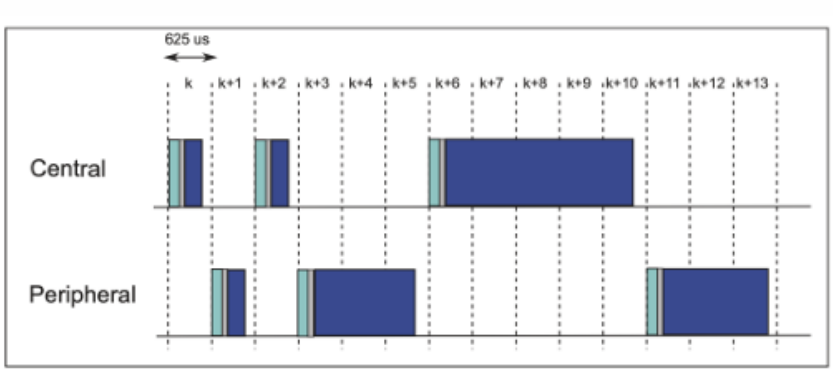


Figura 5.18: Periodo di sincronizzazione antecedente la chiusura di uno slot

5.7.5 Connection (BLE)

Il concetto di evento in BLE permette di suddividere il tempo in una maniera logicamente più flessibile rispetto a quella rappresentata dagli slot di durata

statica. [In questo modo è il master a comunicare per ogni connessione la frequenza, quindi non vi sono clock e pattern di frequency hopping generali].

Il motivo per cui il Bluetooth Classic coinvolge al massimo sette slave risiede nel fatto che ciascuno di esse deve rispettare la struttura a slot e quindi consumare energia anche se il suo duty cycle è irrisolto.

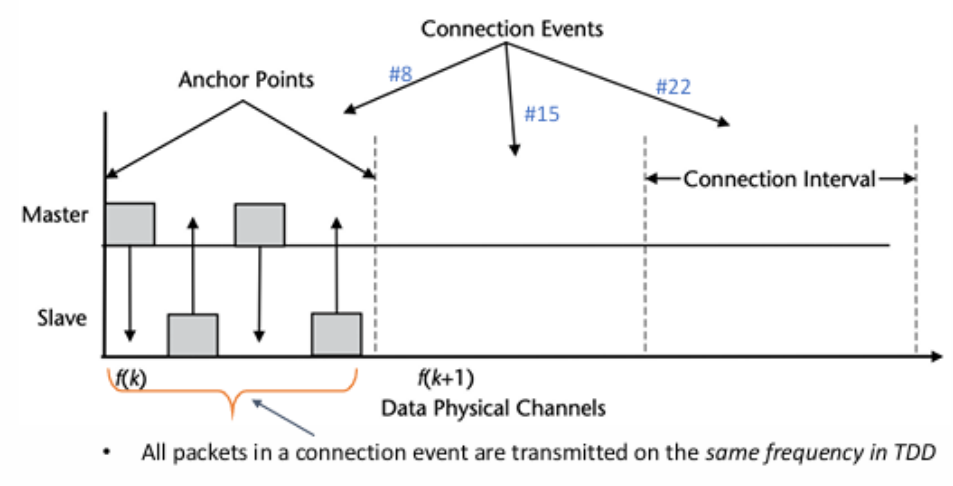


Figura 5.19: Comunicazione master/slave BLE

Dalla Figura 5.19 notiamo che viene usato il canale 8 per il primo evento, il canale 15 per il secondo evento ed il 22 per il terzo evento. L'inizio di un connection event è chiamato anchor point.

Per connection interval si intende il tempo che intercorre tra un evento e l'altro. Cambia da slave a slave ed è controllato dal master. Ha un range che varia da 7.5 ms a 4.0 s per incrementi di 1.25 ms. Quando un master ed uno slave si connettono per la prima volta, lo slave può anche mandare in prima risposta il PPCP (Peripheral Preferred Connection Parameters), che è un'informazione di controllo con la quale specifica le sue capacità ed il suo duty cycle (quindi il connection interval desiderato).

Tramite lo Slave Latency (un parametro dell'advertisin packet), lo slave può comunicare al master il numero massimo di connection event che può saltare senza dover ascoltare né replicare. Il master può anche decidere di usare un connection interval piccolo ma se non può essere rispettato dallo slave può impostare la latenza di quest'ultimo in modo tale da non rispondere al master per un certo numero di eventi consecutivi. [In questo modo lo slave non comanda il connection interval e può risparmiare energia].

Il supervision timeout è il massimo tempo permesso tra due pacchetti ricevuti, serve comprendere se è stata persa la connessione. Ad esempio se uno slave ha impostato la sua slave latency a dieci ma non è in grado di rispondere anche l'undicesima, la dodicesima e la tredicesima volta, il master comprende che lo slave ha avuto un problema e dichiara persa la connessione. Per comunicare nuovamente con il master lo slave deve prima tornare indietro alla fase di advertiser.

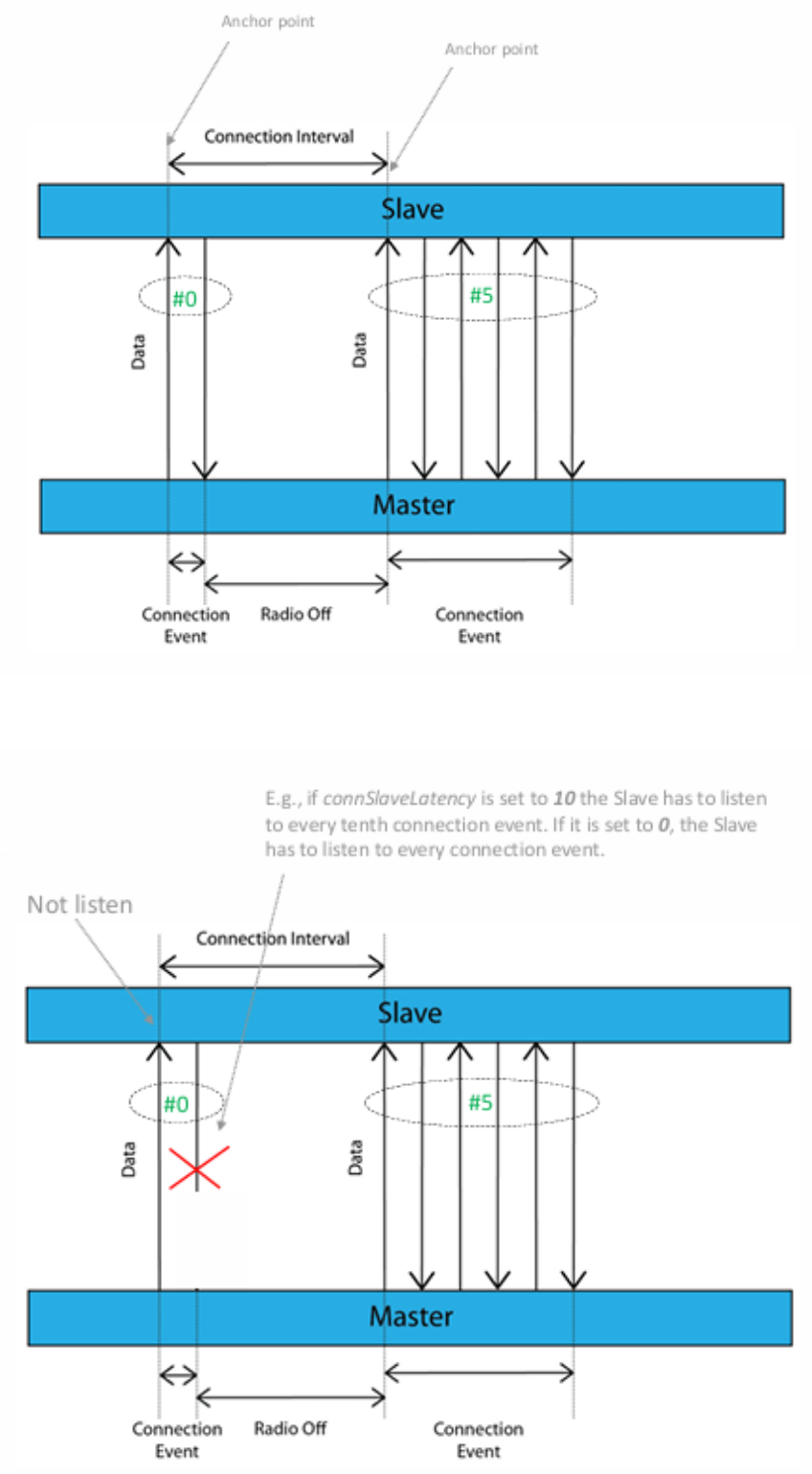


Figura 5.20: Connessione tra Master e Slave in BLE

Adaptive Frequency Hopping (AFH)

Il BLE utilizza questa strategia per saltare tra i 37 canali dati. Il master decide se ciascuno di essi può essere usato o meno.

Quando il master sospetta un'interferenza sul canale lo esclude dal pattern frequency hopping. L'informazione riguardante i canali buoni e cattivi è riportata in un bitmap, comunicata dal master verso lo slave insieme al connection request packet.

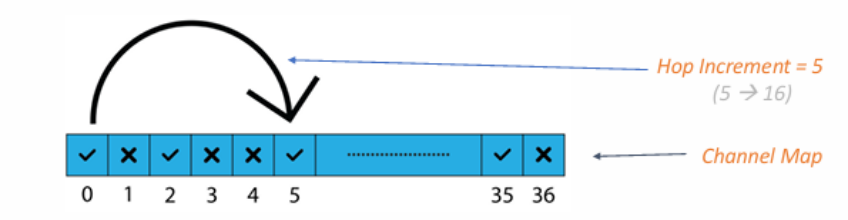


Figura 5.21: Interpretazione grafica del Channel Map

Un'ulteriore parametro che il master comunica al suo slave è l'hop increment (un numero scelto casualmente tra 5 e 16 che determina la strategia del salto).

La combinazione di queste due informazioni permette di stabilire quale canale viene usato in ogni connection interval.

L'hop increment è scelto casualmente per ridurre la probabilità di collisione (che si ha se due master scelgono la stessa frequenza di partenza). L'algoritmo utilizzato è il seguente:

$$f_{n+1} = (f_n + hopIncrement) \mod 37$$

Esso è detto adattivo perché il master ascolta periodicamente il canale è ha la possibilità di rimappare il canale (nella channel map). Ad esempio il canale 4 non è usato perché ci sono delle interferenze, però il master ascolta periodicamente questo canale e se si accorge di cambiamenti (non ci sono più interferenze) può inviare all'interno di un connection event un pacchetto allo slave aggiornando il channel map. Il canale ottenuto con un hop increment di N potrebbe essere soggetto ad interferenze. In tal caso sia il master che lo slave sanno che il canale non può essere usato (perché il master aveva inviato precedentemente il channel map) pertanto saltano entrambi altri N canali fino a quando ne trovano uno che può essere usato.

In definitiva, l'hop increment è il minimo salto che il master è lo slave possono fare da un connection event al successivo, tutti i salti dovuti ad eventuali interferenze sono multipli.

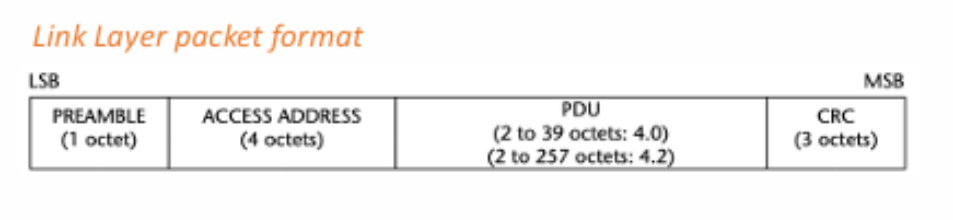
5.8 Pacchetti e LLCP

Il link layer di BLE prevede un solo formato di pacchetto, impiegato si per gli advertising che per i physical channel. Al contrario, il BR/EDR prevede numerosi formati di pacchetti:

- ID - usato prima della connessione tra master e slave per comunicare l'identificativo;
- NULL - non contiene alcuna informazione ma viene utilizzato dagli slave per comunicare il successo della precedente trasmissione o lo stato del buffer di ricezione (quindi sono pacchetti di controllo per verificare che la connessione è attiva e tutto procede correttamente);
- POLL - privo di informazione, è trasmesso dal master agli slave per caire se quest'ultimi sono attivi;
- FHS (Frequency Hop Synchronization) - trasmesso dal master per comunicare la nuova strategia di hopping, questo prima che la piconet venga stabilita o quando una piconet esstente cambia in un'altra;
- DM1 (Data Medium Rate 1-slot) - usato per trasportare pacchetti di controllo e di dati.

5.8.1 Formato del Pacchetto

Nella struttura unica dei pacchetti BLE è possibile distinguere un *Preamble*, un *Acces Code*, la *PDU (Protocol Data Unit)* ed il codice *CRC (Cyclic Redundancy Check)*.



In figura sono riportati anche LSB (Least Significant Bit) ed MSB (Most Significant Bit) che permettono al trasmettitore ed al ricevitore di conoscere l'ordine con il quale vengono comunicati i dati.

- Preamble: sequenza di bit a livello fisico nota sia per il trasmettitore che per il ricevitore, impiegata per la sincronizzazione, per il timing e per il AGC (Automatic Gain Control). Tramite essa il ricevitore può comprendere quando arriva un nuovo pacchetto e la sua potenza in ricezione.
- CRC: utilizzato per proteggere il canale da errore, rendendolo più robusto. Consiste in un checksum di 24 bit calcolato sulla PDU.
- Access Address: è specifico del protocollo Bluetooth. Ciascun nodo ha bisogno di un indirizzo per essere identificato durante la trasmissione e la ricezione, in aggiunta a questo il bluetooth richiede anche un nuovo indirizzo specifico (l'Access Address) per la connessione. Attraverso questo è possibile evitare anche interferenze, perché un master riesce a capire

se uno slave non è autorizzato a trasmettere (quando i campi non corrispondono) e procede alla chiusura dell'evento, mandando i dispositivi in sleep per risparmiare energia.

- PDU: usata per trasmettere l'informazione vera e propria, ha lunghezza variabile (da 2 a 39 per il BLE 4.0 e da 2 a 257 per le versioni più recenti). Solitamente è preferibile tenere basso il numero massimo di byte dedicati alla PDU.

5.8.2 Bit Stream Processing

É un'attività del physical layer che trasmette i bit logici (l'informazione) in forme d'onda che vengono radiate sul canale.

Figura 5.22: Fasi del Bit Stream Processing

I blocchi che permettono ciò sono:

- CRC Generation - fese dove viene generato il codice per proteggere il canale dagli errori;
- Whitening/Dewhiteding - fase in cui i bit di ciascun pacchetto sono codificati in sequenze casuali prima della trasmissione secondo una codifica (nota da protocollo) sia dal master che dallo slave;
- Encryption/Decryption - fase opzioale eseguita solo se il master vuole una connessione di tipo criptato.

5.8.3 Address

I dispositivi bluetooth sono identificati da un indirizzo a 48bit (simile al MAC address). Si distinguono in:

1. Public Addresses - ciascun dispositivo ha un numero statico, intrinseco al dispositivo, che lo identifica;
2. Random Addresses -un dispositivo è in grado di connettersi ed essere identificato senza impiegare il suo indirizzo hardware in modo da proteggere la sua privacy, attraverso la generazione casuale a runtime. Questo tipo di indirizzi si dividono ancora in

(a) Static Address - l'indirizzo cambia sporadicamente (è generato in fase di accensione o al cambiamento di determinate impostazioni del dispositivo, ma non cambi durante un ciclo di carica);

(b) Private Address - può essere *non-resolvable* quando l'indirizzo logico casuale è rovvisorio per un certo intervallo di tempo (ma comunemente non è utilizzato), oppure *resolvable* quando l'indirizzo logico può essere cambiato a discrezione del dispositivo a runtime in maniera semplice. Il dispositivo utilizza una chiave, chiamata IRK (Identify Resolving Key), per produrre un indirizzo privato a partire da quello logico.

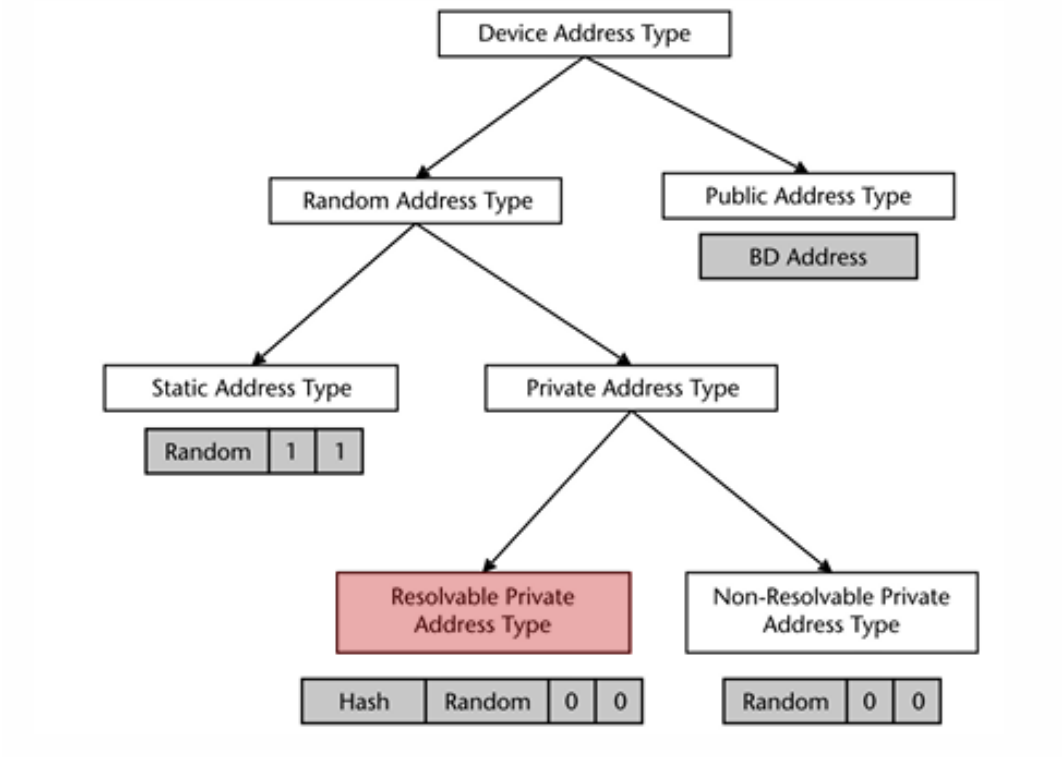


Figura 5.23: Formati di Device Address

Quello identificato in rosso in Figura 5.23 è il formato più flessibile ed efficiente. Dispositivi vecchi potrebbero non avere la possibilità di utilizzare questa funzionalità e sono costretti ad utilizzare un Public Address. Dispositivi recenti hanno un bit dedicato nell'indirizzo che identifica se questo è pubblico (se posto a 0) o privato (se posto ad 1).

5.8.4 Device Filtering and White list

Il link layer potrebbe essere ristretto per rispondere solo a determinati dispositivi. Si definisce white list la lista degli indirizzi dei dispositivi che permettono di comunicare con loro. [Se vi è un dispositivo avente indirizzo casuale ed un altro dispositivo vuole inserirlo nella sua white list deve conservare la IRK associata a quel dispositivo per poter risolvere l'indirizzo].

Ogni volta che un dispositivo riceve un pacchetto, controlla inizialmente il suo indirizzo e se questo corrisponde ad un dispositivo che permette la comunicazione; se non fa parte della white lst non risponde per risparmiare energia.

5.8.5 Advertising Channel PDU

L'Adverising Channel PDU presenta un Header ed un Payload. In base al tipo di PDU presente nell'header se ne distinguono tre tipi: *Advertising, Scanning ed Initiating*, corrispondenti alle tre attività precedentemente analizzate. Il campo RFU (Reserved for Future Use) è solitamente imposto ad uno specifico pattern in quanto inutilizzato. I campi TxAdd indicano se trasmettitore e ricevitore utilizzano un indirizzo pubblico o casuale (quindi se è da usare o meno la chiave per risolvere l'indirizzo).

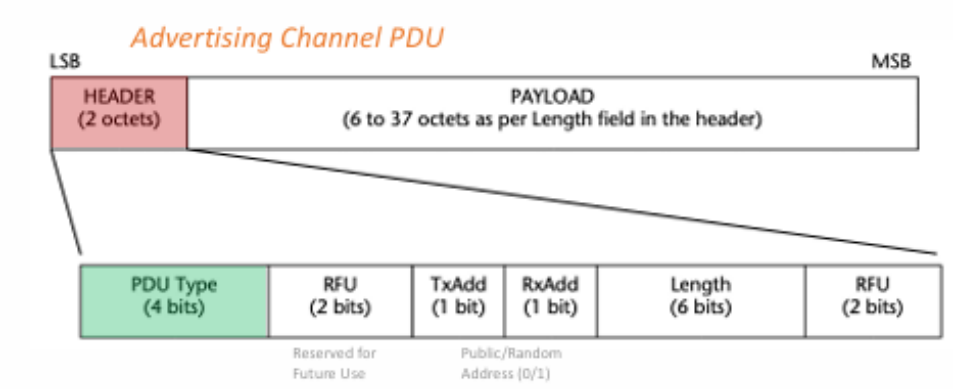


Figura 5.24: Struttura di un'Advertising Channel PDU

5.8.6 Data Channel PDU

I Data Channel PDU vengono scambiati tra master e slave uno volta che la connessione viene stabilita. Sono pacchetti trasmessi sul physical channel e possono essere anche pacchetti di controllo. Durante la trasmissione può capitare che vi sia la possibilità di ridefinire alcune proprietà della connessione come l'AFH.

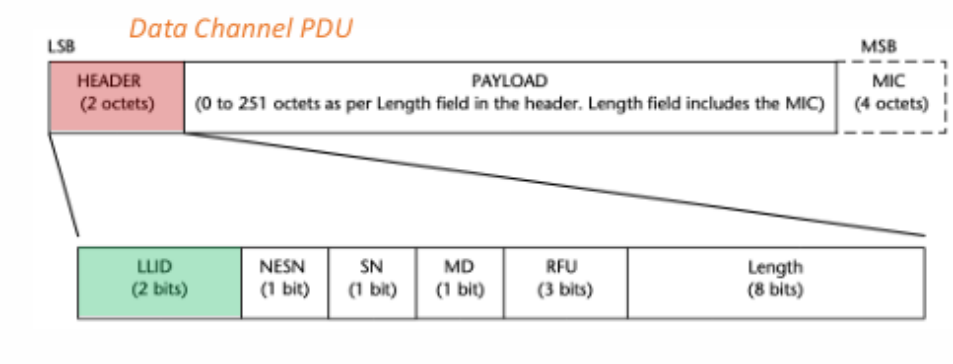
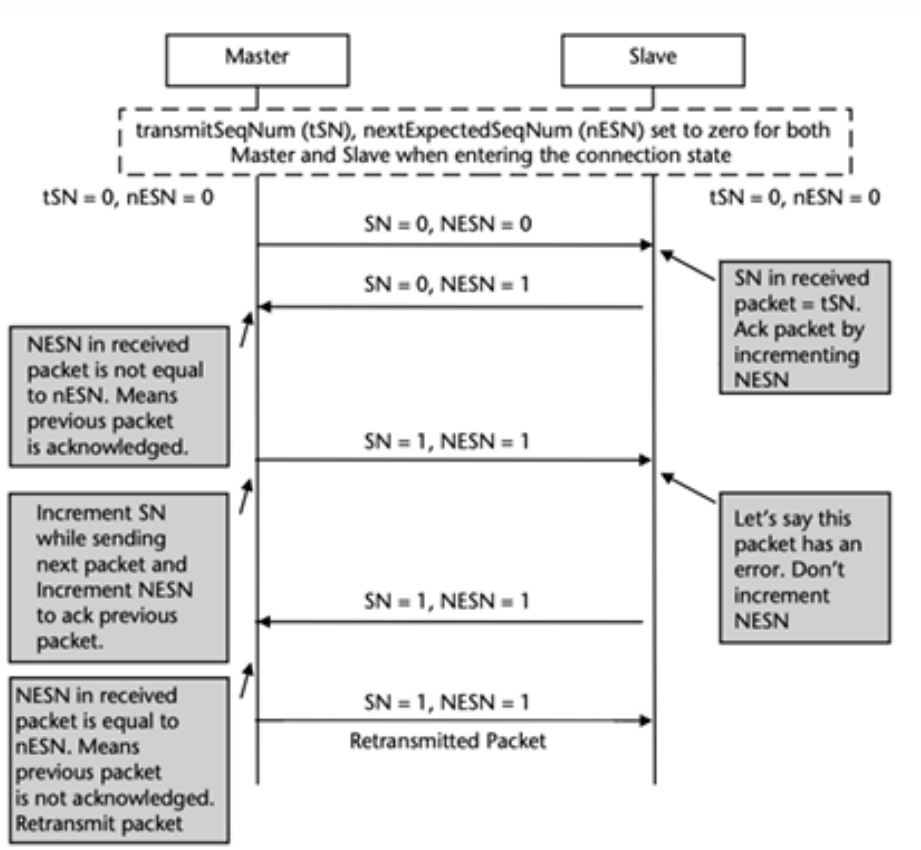


Figura 5.25: Struttura di un Data Channel PDU

Il Data Channel PDU presenta un Header, un Payload ed eventualmente un MIC (Message Integrity Check) nel caso in cui il pacchetto sia criptato. Nell'header il campo LLID (Link Layer ID) determina il tipo di data channel PDU. Il campo RFU (Reserved for Future Use) è settato arbitrariamente ad una sequenza. I campi SN (Sequence Number) e NESN (Next Expected Sequence Number) sono due bit utilizzati per implementare la strategia di *acknowledgment*.

5.8.7 ACK e Flow Control



Il bit SN identifica il pacchetto corrente mentre il bit NESN identifica il pacchetto che si aspetta di ricevere.

Il master trasmette il primo pacchetto con SN e NESN entrambi impostati a zero. Il fatto che SN sia 0 implica che il master sta inviando il suo primo pacchetto, NESN impostato a 0 invece indica che il master si aspetta una risposta dallo slave. Quest'ultimo risponde con il suo primo pacchetto avente SN pari a 0 e NESN pari ad 1 perché è in attesa di un secondo pacchetto da parte del master. Esso risponde allora con SN e NESN entrambi impostati ad 1. In questo ponto lo slave si accorge attraverso il CRC che non può decodificare correttamente il pacchetto ricevuto (si verifica un errore); risponde pertanto con SN e NESN entrambi impostati ad 1 (il NESN non incrementato fa capire al master che deve ritrasmettere il pacchetto precedente).

5.8.8 LLCP (Link Layer Control Protocol)

É il protocollo responsabile dell'incapsulare i dati (L2CAP) prevenienti dai livelli superiori ed al passarli al link layer.

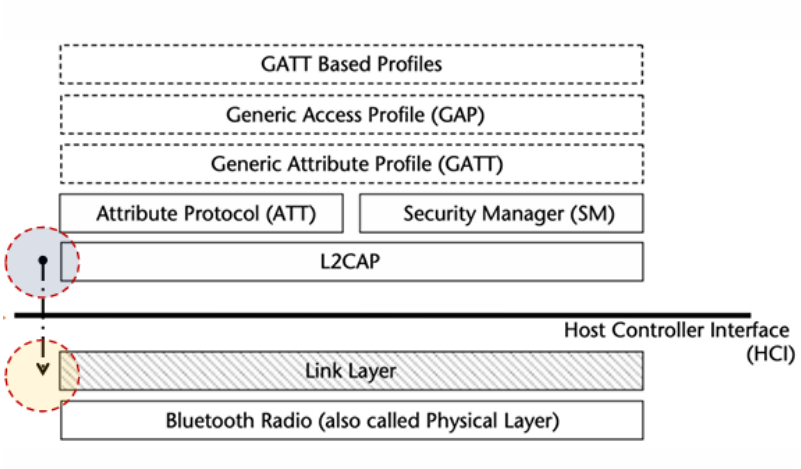


Figura 5.26: Gestione della comunicazione tramite LLCP

L'obiettivo è quello di raccogliere i diversi tipi di pacchetti e messaggi studiati (control message e data messages) in procedure. Queste sono set di messaggi, quindi un handshake tra il master e lo slave utilizzata per uno scopo specifico.

La Channel Map Update Procedure è usata quando il master ha intenzione di aggiornare il pattern di Frequency Hopping. Come detto più volte, all'inizio della connessione il master invia allo slave il pattern ma sulla base delle condizioni dell'ambiente monitorante può occasionalmente decidere di cambiarlo (questa procedura è messa in atto con un singolo pacchetto).

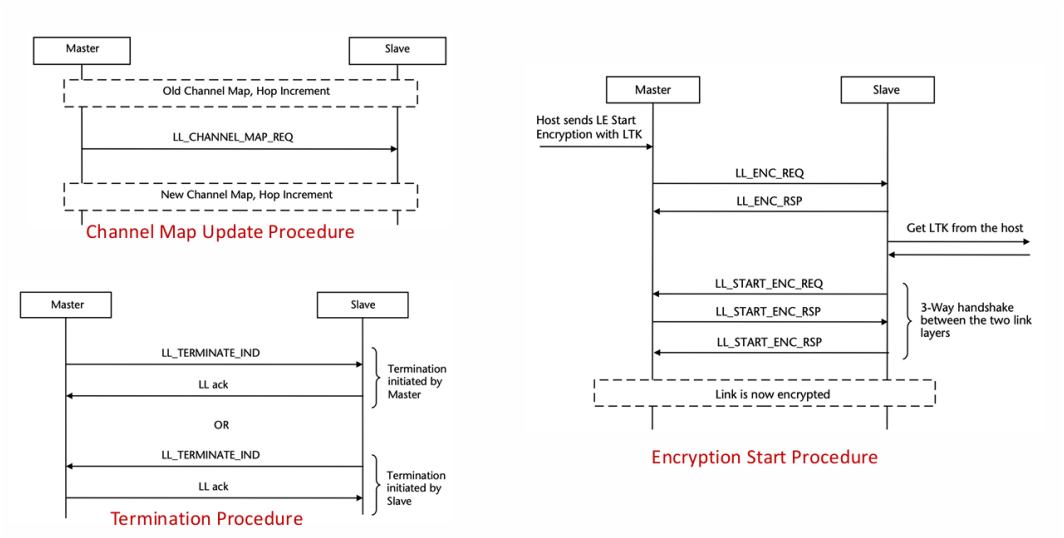


Figura 5.27: Diagrammi di flusso relativi a tre diverse procedure

5.9 BLE Host

Gli Host nel BLE sono eseguiti su una macchina general purpose che è in grado di eseguire anche altri tipi di tecnologie allo stesso tempo.

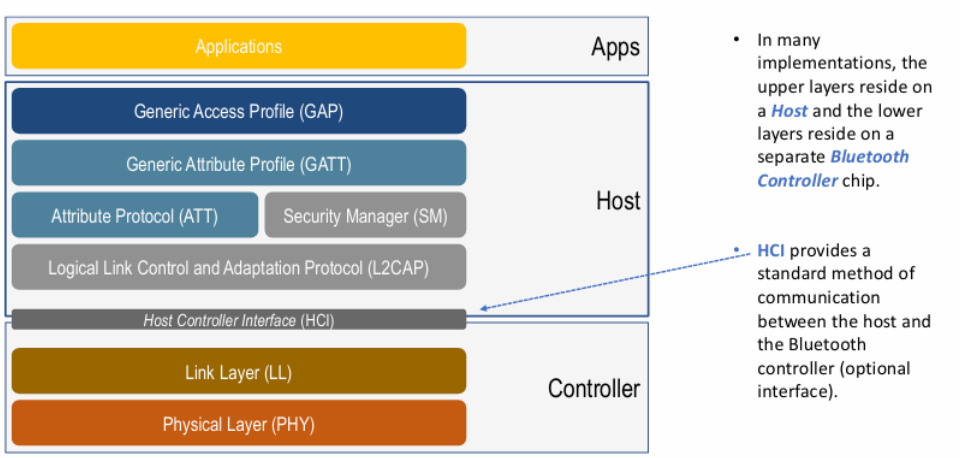


Figura 5.28: Architettura protocollare

5.9.1 Stack Protocollare ed Architettura Dual Mode

In Figura 5.20 possiamo notare le principali differenze tra le pile protocollari di Bluetooth LE e BR/EDR e l'architettura Dual Mode (dove vengono messe insieme le differenze tra i due).

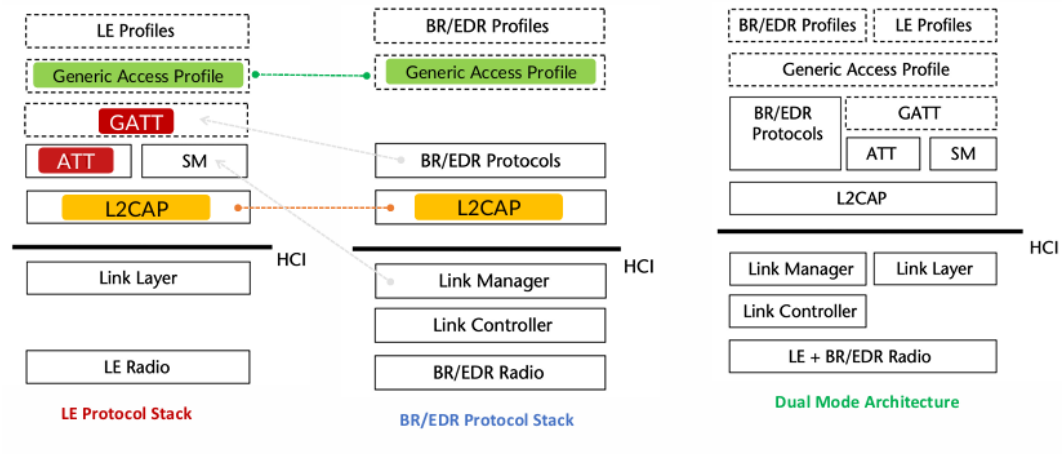


Figura 5.29: Architettura Dual Mode

In giallo è evidenziato il primo livello dell'host, che risulta essere lo stesso per entrambi. Il più grande cambiamento è evidenziato in rosso. Entrambe le pile, invece, condividono il profilo principale evidenziato in verde.

5.9.2 Logical Link Control and Adaptation Protocol (L2CAP)

É usato lo stesso del Bluetooth Classic rimuovendo molte funzionalità. Sono lasciate solamente quelle necessarie a gestire la frammentazione e la ricombinazione.

5.9.3 Security Manager Protocol (SMP)

Protocollo legato alla sicurezza, utilizzato per condividere le chiavi tra master e slave (generalmente alla prima connessione tra i due; a meno che non vengono effettuate procedure di aggiornamento, le chiavi rimangono invariate per tutta la comunicazione).

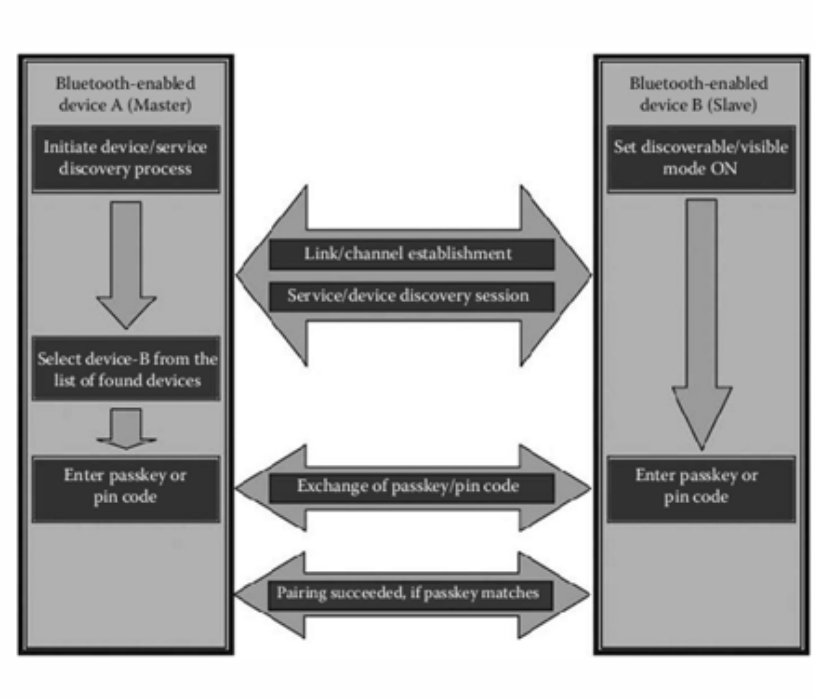
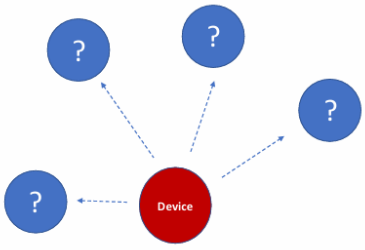


Figura 5.30: Funzionamento del SMP

5.9.4 SDP (Service Discovery Protocol) [BR/EDR]

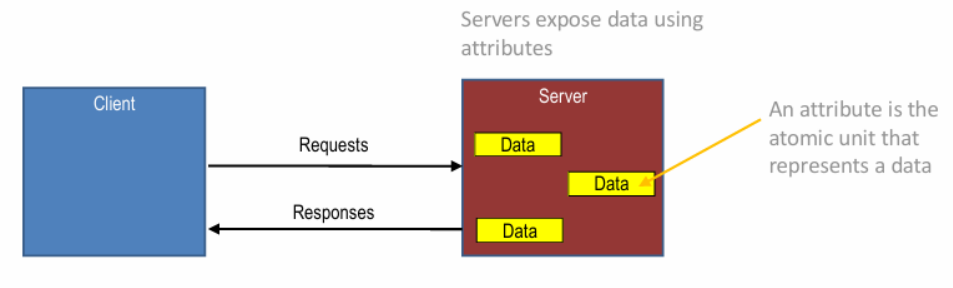
Questo protocollo del bluetooth classic ha lo scopo di scoprire i dispositivi. Ai livelli superiore non è solo importante scoprire il dispositivo fisico ma è necessario sapere anche i suoi servizi e le sue caratteristiche.

In figura, ad esempio, possiamo pensare ai quattro cerchi blu come dispositivi logici ad alto livello che corrispondono allo stesso dispositivi a basso livello (condividono lo stesso indirizzo di livello 2 ma hanno indirizzi diversi a livello più alto).

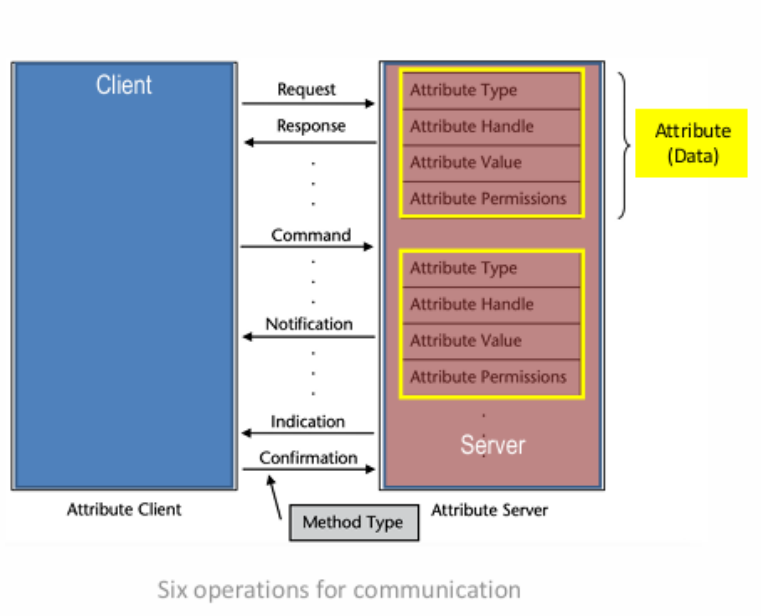


5.9.5 Attribute Protocol (ATT)

É il protocollo BLE più importante da un punto di vista di alto livello, il quale permette di gestire i dati (ad alto livello definiamo "attributo" l'unità atomica che rappresenta il dato).



L'architettura della comunicazione è di tipo client/server (i server posseggono i dati ed i client li richiedono). Generalmente il client è rappresentato dal master ed i server da slave o advertiser



Nella figura sopra sono riportate sei operazioni possibili nella comunicazione tra client e server.

1. Il client può inviare una richiesta al server;
2. questo fornisce una risposta.
3. Il client può inviare un comando al server.
4. Il server può inviare una notifica al client (un'advertisement di dati).
5. Il server può inviare un'indicazione al client;
6. questo può rispondere con una conferma.

[Le indicazioni sono simili alle notifiche, la differenza risiede nell'attesa della conferma].

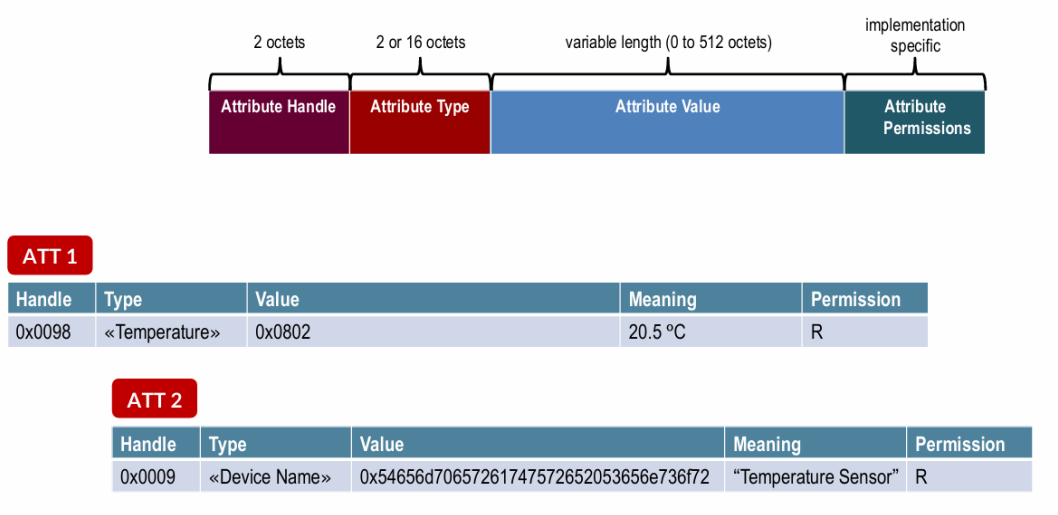


Figura 5.31: Formato di un attributo (in alto) e relativi esempi (in basso)

In un server si distinguono diverse tipologie di attributi, ciascuno dei quali composto da:

- *Handle* - usato dal client per indirizzare i dati;
- *Type* - significato dei dati, determinato attraverso l'Universal Unique IDentifier (UUID);
- *Value* - i dati veri e proprii;
- *Permissions* - si valuta se è possibile accedere o meno.

l'UUID è un campo di 128 bit che può possedere qualsiasi dispositivo. Un numero così elevato di bit permette di minimizzare la probabilità che due dispositivi possano scegliere lo stesso identificativo sulle totali 2^{128} combinazioni.

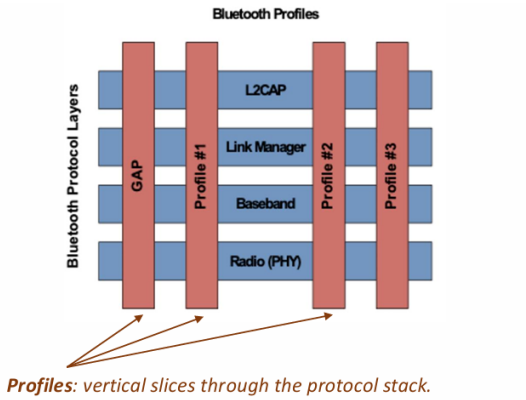
Nell'Internet of Things però è troppo grande, considerando le capacità dei dispositivi coinvolti, quindi se ne utilizza una versione più corta da 16 o 32 bit.

Solo una volta, all'inizio del suo ciclo di vita in quella rete, ciascuna entità scambia il suo indirizzo completo; prosegue poi utilizzando una versione ridotta. Se però due dispositivi scelgono lo stesso indirizzo a 16 bit allora il master si accorge di dover trasmettere pacchetti a due indirizzi aventi lo stesso indirizzo (ma che non sono in alcun modo correlati fra loro) e chiede ad entrambi l'indirizzo completo e li invita a cambiare quello ridotto per evitare problematiche.

Nella tabella sono riportati dei comandi utili a tradurre a livello alto ciò che può essere usato a basso livello.

Name	Description
Error Response	Something was wrong with a request
Exchange MTU Request / Response	Exchange new ATT_MTU
Find Information Request / Response	Find information about attributes
Find By Type Value Request / Response	Find specific attributes
Read By Group Type Request / Response	Find specific group attributes and ranges
Read By Type Request / Response	Read attribute values of a given type
Read Read / Response	Read an attribute value
Read Blob Request / Response	Read part of a long attribute value
Read Multiple Request / Response	Read multiple attribute values
Write Command	Write this – no response
Write Request / Response	Write an attribute value
Prepare Write Request / Response	Prepare to write a value (long)
Execute Write Request / Response	Execute these prepared values
Handle Value Notification	Notify attribute value – no confirmation
Handle Value Indication / Confirmation	This attribute now has this value

5.10 BLE Host Profiles



Dalla figura notiamo come i protocolli sono rappresentati dai blocchi orizzontali (blu) mentre i profili da quelli in verticale (rossi). Un profilo definisce la metodologia di utilizzo dei diversi protocolli e delle loro funzionalità per fornire supporto a servizi specifici. Esso, quindi, è un set di servizi che adempiono ad un caso d'uso. Più profili possono utilizzare lo stesso servizio.

5.10.1 General Attribute Profile (GATT)

In questo modo si raggruppano gli attributi in strutture semplici da gestire.

Nella Figura 5.32 notiamo come abbiamo tre attributi distinti: il primo rappresenta il nome del dispositivo, il secondo lo stato della batteria ed il terzo la temperatura misurata. Da un punto di vista applicativo appartengono tutti allo stesso dispositivo e con il GATT è possibile unire questi attributi in una sola caratteristica (composta dal: nome, stato batteria e temperatura).

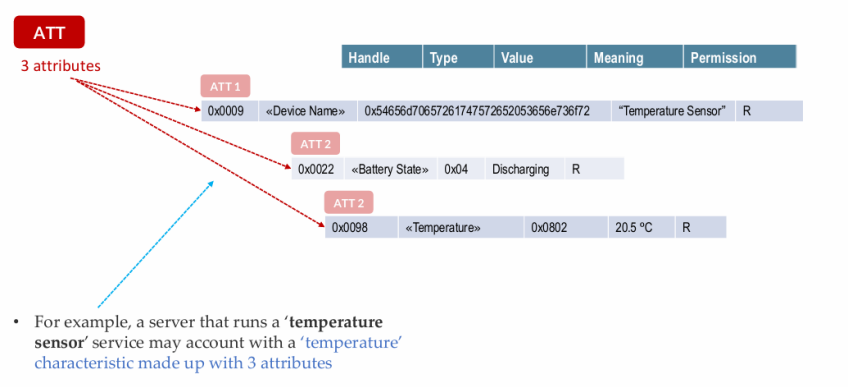
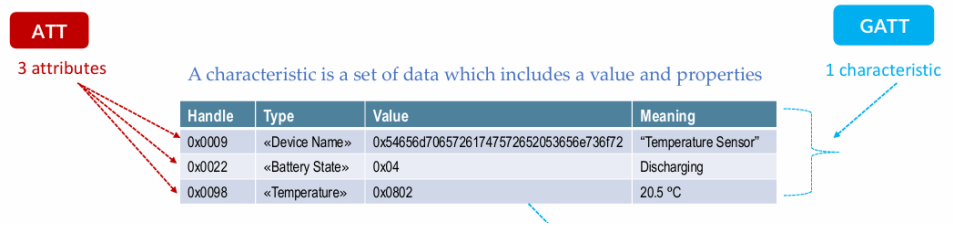
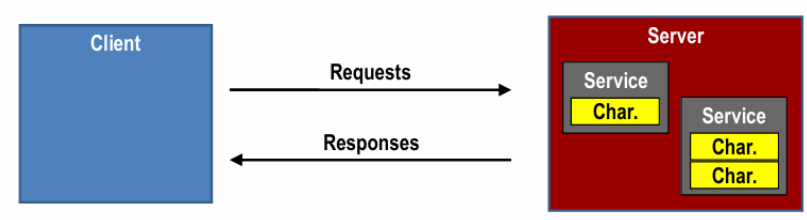


Figura 5.32: Tre attributi relativi ad un sensore di temperatura



L'architettura della comunicazione GATT è di tipo client/server. In particolare: il client inizializza la transazione verso il server e può ricevere risposte da esso; il server riceve i comandi e le richieste dal client ed invia risposte, indicazioni e notifiche. Un dispositivo può comportarsi come client, come server o come entrambi. Esistono tre procedure.

1. *Client-initiated*: lettura o scrittura di una caratteristica.
2. *Server-initiated*: notifica o indicazione del valore di una caratteristica.
3. *Discovery*: scoperta di servizi e caratteristiche.



5.10.2 Profile Dependencies

I profili in BR/EDR sono più numerosi e spesso incorrelati tra loro (quando si vuole produrre una nuova applicazione si può decidere di produrre un nuovo profilo dedicato all'applicazione e sviluppare tutte le funzionalità richiesta da zero). Al contrario nel LE i profili diversi dal GAP e GATT condividono lo stesso livello e non possono essere annidati tra loro.

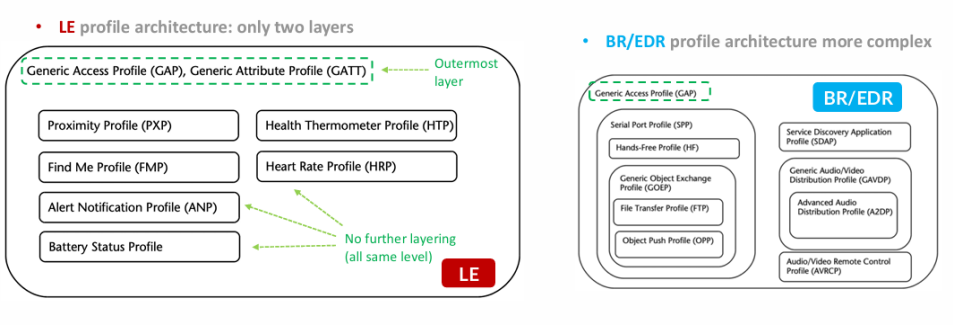
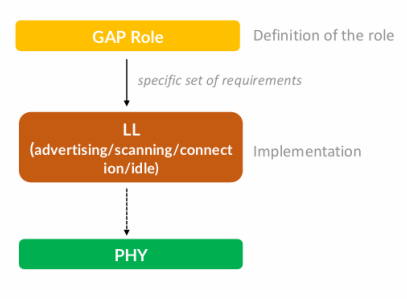


Figura 5.33: Differenze in termini di profile dependencies tra LE e BR/EDR

5.10.3 Generic Access Profile (GAP)

Lo scopo del GAP è quello di gestire l'accesso alle operazioni di livello più basso. Generalmente il Link Layer (livello 2) gestisce gli accesso, ciò comporta che sia strettamente correlato al GAP.

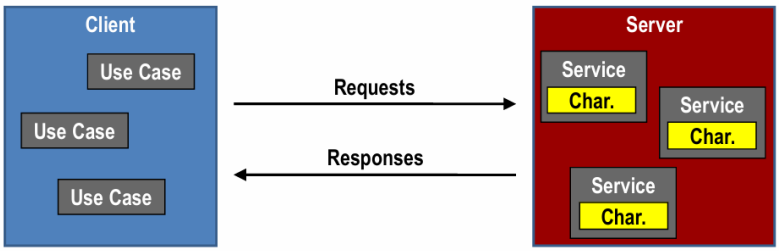


Il GAP definisce il ruolo dei profili (Broadcaster, Observer, Peripheral e Central).

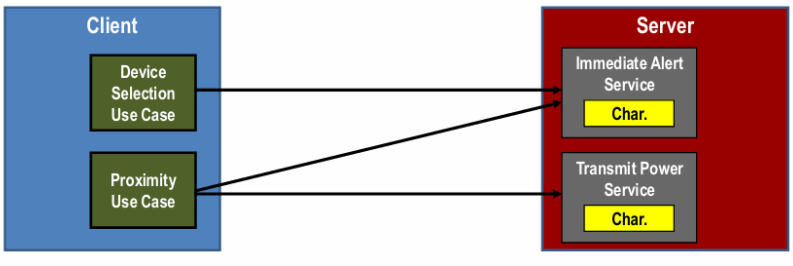
GAP role	Link Layer state
Broadcaster	Advertising
Observer	Scanning
Peripheral	Advertising Connection (Slave)
Central	Scanning Initiating Connection (Master)

Broadcaster	Peripheral	Observer	Central
No need for a radio receiver	Needs both a receiver and transmitter	No need for a transmitter	Needs both a receiver and transmitter
No bi-directional data transfer	Supports bi-directional data transfer	No bi-directional data transfer	Supports bi-directional data transfer
Reduced hardware, reduced BLE software stack	Requires the full BLE software stack	Reduced hardware, reduced BLE software stack	Requires the full BLE software stack

5.10.4 BLE Applications



Le applicazioni in BLE seguono un approccio client/server, in questo tipo di architettura vengono scambiati servizi completi.

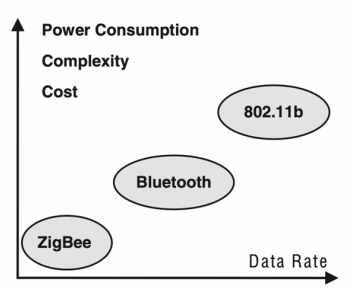


Il mapping non è uno ad uno ma più casi d'uso appartenenti ad applicazioni diverse, implementati dal client, possono fare riferimento ad uno stesso servizio implementato dal server.

6 ZigBee

IEEE 802.15.4 è un insieme di protocolli, procedure e regole che si focalizza sui livelli più bassi della pila protocollare. Sulla base di questa tecnologia sono state realizzate diverse tecnologie, tra le quali ZigBee.

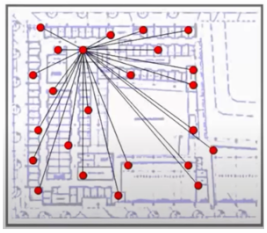
Il Wi-Fi (802.11) è la tecnologia più costosa ma anche quella più veloce in termini di data rate; Bluetooth è una tecnologia intermedia e ZigBee è la tecnologia perfetta per l'IoT in quanto economica ed avente data rate più basso.



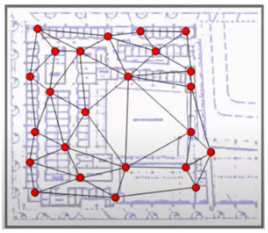
	Data Rate	Typical Range	Application Examples
ZigBee	20 to 250 Kbps	10–100 m	Wireless Sensor Networks
Bluetooth	1 to 3 Mbps	2–10 m	Wireless Headset Wireless Mouse
IEEE 802.11b	1 to 11 Mbps	30–100 m	Wireless Internet Connection

Figura 6.1: Confronto tra ZigBee, Bluetooth e Wi-Fi

Wi-Fi e ZigBee sono entrambi standardizzati dallo stesso ente (IEEE) pertanto condividono molte caratteristiche e funzionalità. Il Bluetooth invece ha più difficolta a coesistere con altre tecnologie.



Star Network (e.g. 802.11)



Mesh Network (e.g. ZigBee)

Notiamo come la topologia del Wi-Fi è di tipo a stella (vi è un solo access point), mentre quella ZigBee è di tipo a maglia (distinguiamo un coordinatore primario ed i dispositivi sono fortemente interconnessi tra loro).

6.1 L1/L2

L'architettura ZigBee prevede complessivamente quatto livelli. Il livello fisico e quello MAC sono stati progettati dall'IEEE mentre il livello di rete e quello applicativo sono stati progettati dalla ZigBee Alliance.

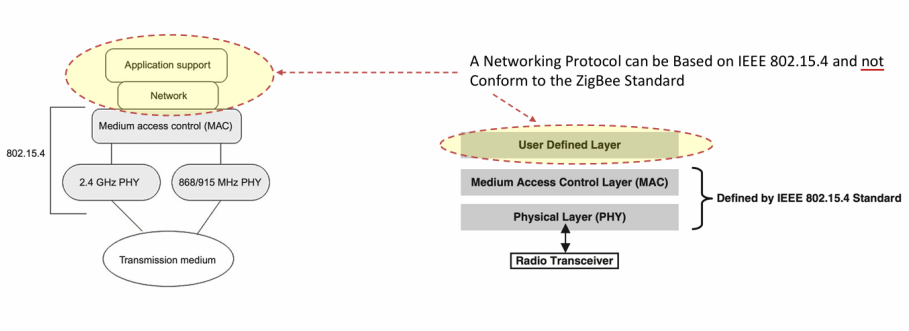


Figura 6.2: IEEE 802.15.4/non-ZigBee Stack

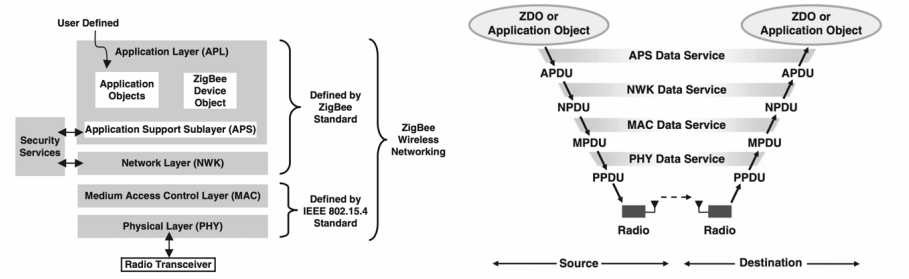


Figura 6.3: IEEE 802.15.4/ZigBee Stack

ZigBee Layer	Description
PHY	Defines the *physical operation* of the device including receive sensitivity, channel rejection, output power, number of channels, chip modulation, and transmission rate specifications (most Zigbee applications operate on the 2.4 GHz ISM band at a 250 kbps data rate).
MAC	Manages RF data transactions between neighboring devices (*point to point*). The MAC includes services such as transmission retry and acknowledgment management, and collision avoidance techniques (CSMA-CA).
NWK	Adds *routing capabilities* that allows RF data packets to traverse multiple devices (*multiple hops*, *Mesh network*)
APS	Application layer that defines various addressing objects including *profiles*, *clusters*, and *endpoints*.
ZDO	Application layer that provides *device and service discovery* features and advanced network management capabilities.

Ad un nodo fisico possono fare riferimento più nodi logici virtuali che prendono il nome di *endpoints* (questi sono tra di loro distinti anche se in esecuzione sulla stessa macchina). Ciascuno di essi è composto da più cluster, ossia record, che permettono di immagazzinare le informazioni.

- EUI (Extended Unique Identifier) è il MAC address del dispositivo, ha però 64 bit invece che 48 (aumenta il numero di dispositivi che possono essere indirizzati).
- Network Address è l'indirizzo logico, composto da 16 bit, assegnato a ciascun dispositivo nel momento in cui si unisce alla rete. Potrebbe cambiare nel tempo per garantire una migliore efficienza e sicurezza rispetto ad un indirizzo statico.
- Endpoint è un indirizzo di 8 bit molto simile a quello di una porta TCP
- Cluster ID è un indirizzo di 16 bit molto simile ad un handle (puntatore).

6.2 Physical Layer

Il livello fisico dell'IEEE 802.15.4 è responsabile dei seguenti task:

- attivazione e disattivazione del ricetrasmettitore;
- Energy Detection (ED) per comprendere se un dispositivo è connesso o meno;
- Link Quality Indicator (LQI) per i pacchetti ricevuti;
- Clear Channel Assessment (CCA) per Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) che ascolta il canale e capisce se altri dispositivi stanno trasmettendo o meno;
- selezione delle frequenze del canale;
- trasmissione e ricezione delle informazioni.

IEEE 802.15.4 – Receiver Sensitivity Definition

Term	Definition	Conditions
Receiver sensitivity	Lowest input power for which the PER conditions are met	• PSDU length: 20 octets • PER: < 1% • Interference: None Note: Power measured at antenna terminals
Packet error rate (PER)	Average fraction of transmitted packets that are not correctly received	Average measured over random PSDU data

PHY (MHz)	Frequency Band (MHz)	Geographical Region	Modulation	Channels	Bit Rate (kbps)	Typical Output Power (dBm)
868/915	868-868.6	Europe	BPSK	1	20	0
	902-928	United States	BPSK	10	40	0
2450	2400-2483.5	Worldwide	O-QPSK	16	250	0

PHY	Band	Data Parameters			Channels	Receiver Sensitivity
		Bit Rate (kb/s)	Symbol Rate (kbaud)	Modulation		
868 MHz	868.0-868.6 MHz	20	20	BPSK	1	$-92dBm$
915 MHz	902.0 - 928.0 MHz	40	40	BPSK	10	$-92dBm$
2.4 GHz	2.4 - 2.4835 GHz	250	62.5	16-ary orthogonal	16	$-85dBm$

TABLE I
IEEE 802.15.4 PHY CHARACTERISTICS.

Modalità di CCA disponibili:

- **CCA Mode 1** (*energy above threshold*) - la potenza del segnale ricevuto è confrontata ad una soglia che permette di determinare se il canale è occupato o meno (sotto la soglia il canale è libero). [Modalità più semplice ed usata].
- **CCA Mode 2** (*carrier sense only*) - il segnale ricevuto viene demodulato per capire se è legato o meno a ZigBee. [La si usa quando ZigBee opera nella stessa banda del Bluetooth].
- **CCA Mode 3** (*carrier sense with energy above threshold*) - unione delle precedenti modalità.

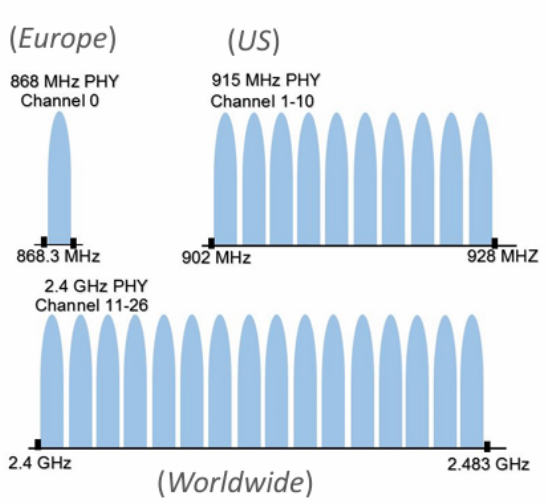
2.4 GHz	Sub-GHz system
larger bandwidth, more channels and worldwide availability	lower bandwidth (and duty-cycle limitations in Europe)
coexistence with other 2.4 GHz systems	for a given output power, subGHz frequencies provide longer range
the higher data-rate at 2.4 GHz reduces the channel occupancy time (it helps avoid interference, reducing the number of retries and the overall power consupmtion)	coexistence with 900-MHz cordless phones. Electromagnetic interference from electrical activity in industrial applications (drives and welding generate noise up to 1 GHz)
antenna sizes are smaller	

La porzione di spettro di interesse è la 2.4GHZ in quanto disponibile in tutto il mondo, al contrario della sub-GHz (868/915 MHz).

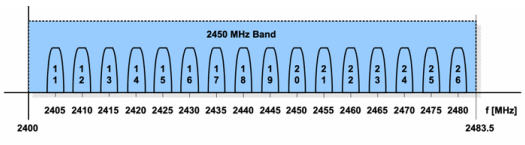
I canali sono numerati da 11 a 26. La frequenza centrale di ciascun canale può essere ricavata come segue:

$$f(k) = 2405 + 5(k - 11)$$

Ciascun canale è ampio 2 MHz ed è spaziato dagli altri di 5 MHz.

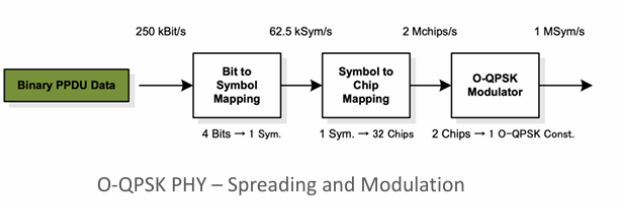


L'architettura standard ZigBee prevede un ordine di modulazioni pari ad M = 16 ed una O-QPSK (Offset-Quadrature Phase Shift Keying) come tecnica di modulazione.



Channel ID	Lower Frequency	Centre Frequency	Upper Frequency
11	2404	2405	2406
12	2409	2410	2411
13	2414	2415	2416
14	2419	2420	2421
15	2424	2425	2426
16	2429	2430	2431
17	2434	2435	2436
18	2439	2440	2441
19	2444	2445	2446
20	2449	2450	2451
21	2454	2455	2456
22	2459	2460	2461
23	2464	2465	2466
24	2469	2470	2471
25	2474	2475	2476
26	2479	2480	2481

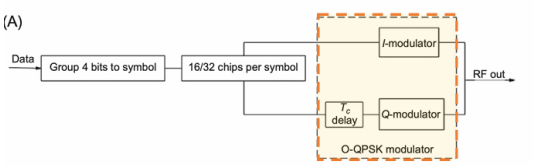
Le informazioni binarie vengono mappate a gruppi di 4 bit in simboli e questi vengono successivamente in una sequenza di chip.



O-QPSK PHY – Spreading and Modulation

O-QPSK PHY – Symbol to Chip Mapping (2450 MHz Band) [12]	
Data Symbols	**Chip values ($c_0..c_{31}$)**
0	11 01 10 01 11 00 00 11 01 01 00 10 00 10 11 10_{bin}
	3 1 2 1 3 0 0 3 1 1 0 2 0 2 3 2_{hex}
1	11 10 11 01 10 01 11 00 00 11 01 01 00 10 00 10
2	00 10 11 10 11 01 10 01 11 00 00 11 01 01 00 10
3	00 10 00 10 11 10 11 01 10 01 11 00 00 11 01 01
4	01 01 00 10 00 10 11 10 11 01 10 01 11 00 00 11
5	00 11 01 01 00 10 00 10 11 10 11 01 10 01 11 00
6	11 00 00 11 01 01 00 10 00 10 11 10 11 01 10 01
7	10 01 11 00 00 11 01 01 00 10 00 10 11 10 11 01
8	10 00 11 00 10 01 01 10 00 00 01 11 01 11 10 11
9	10 11 10 00 11 00 10 01 01 10 00 00 01 11 01 11
10	01 11 10 11 10 00 11 00 10 01 01 10 00 00 01 11
11	01 11 01 11 10 11 10 00 11 00 10 01 01 10 00 00
12	00 00 01 11 01 11 10 11 10 00 11 00 10 01 01 10
13	01 10 00 00 01 11 01 11 10 11 10 00 11 00 10 01
14	10 01 01 10 00 00 01 11 01 11 10 11 10 00 11 00
15	11 00 10 01 01 10 00 00 01 11 01 11 10 11 10 00

O-QPSK PHY – Symbol to Chip Mapping (2450 MHz Band)

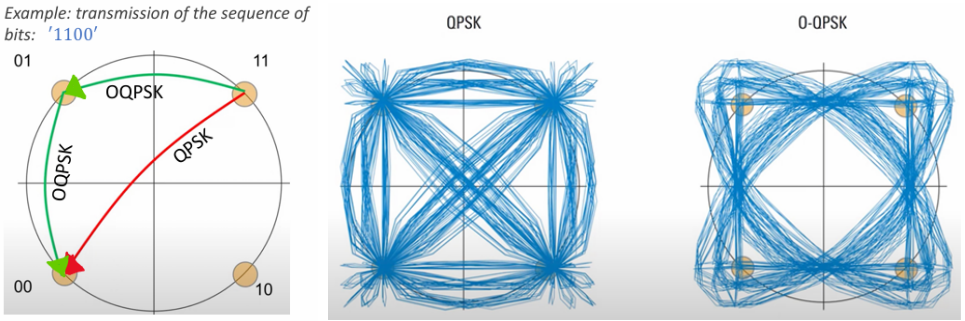


La Direct Sequence Spread Spectrum (DSSS) è progettata per promuovere la coesistenza (non si trasmettono 4 bit ma 32). Viene, quindi, utilizzata più banda di quella strettamente necessaria in maniera tale da distribuire il segnale, ciò comporta un data rate inferiore ma in compenso il sistema sarà più robusto al rumore e alle interferenze.

6.2.1 Modulazione

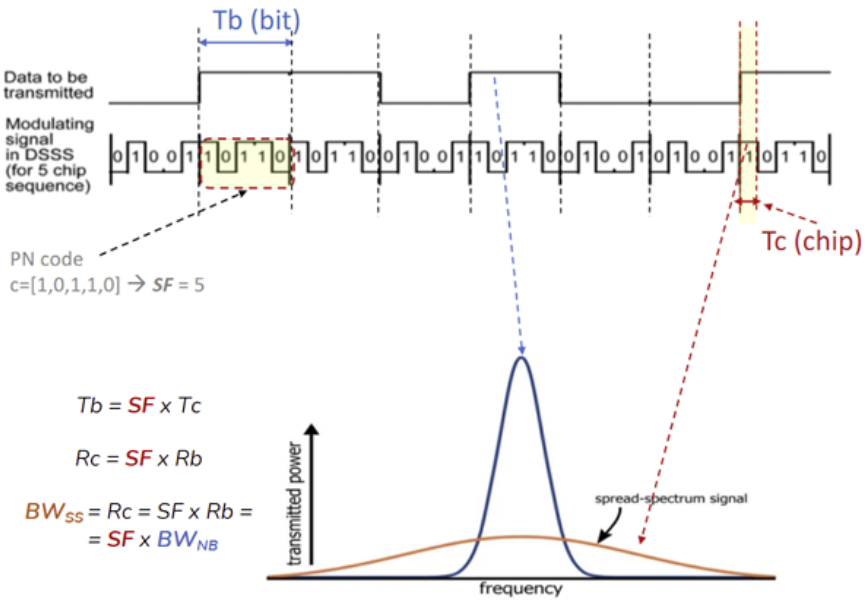
OQPSK

Una componente della transizione è ritardata di mezzo simbolo, questo per evitare di passare dall'origine.



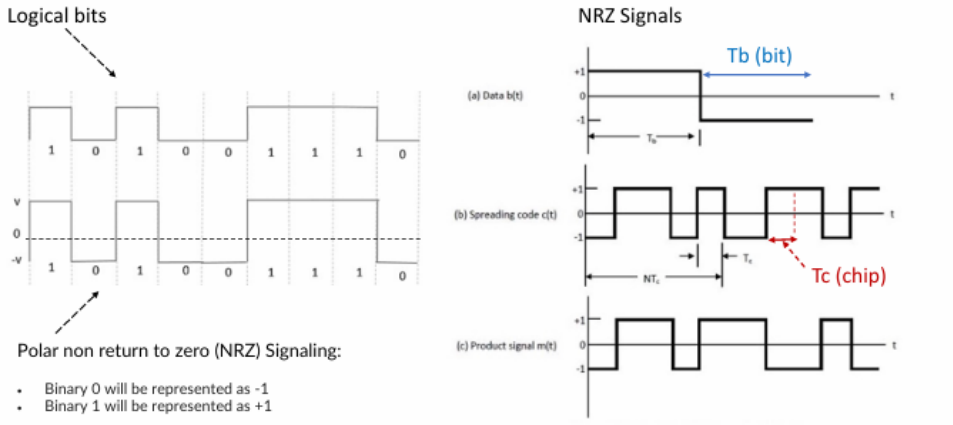
DSSS

L'informazione viene divisa e trasmessa simultaneamente sul maggior numero di frequenze concesse all'interno di una particolare banda.



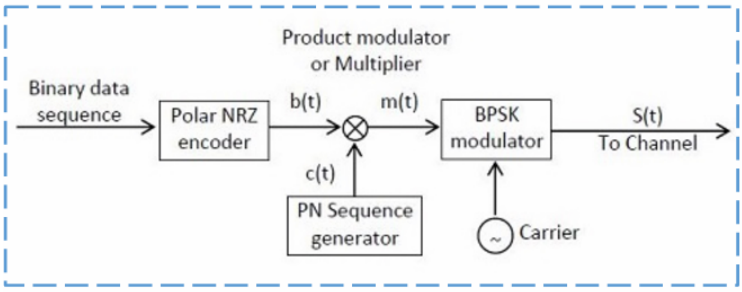
Supponendo di avere una sequenza $c = [1, 0, 1, 1, 0]$; se volessimo trasmettere il bit 1 il segnale modulato sarà pari alla parola c, altrimenti (per trasmettere il bit 0) sarà pari al suo complemento $\bar{c} = [0, 1, 0, 0, 1]$.

Le sequenze di bit (0 ed 1) devono essere poi convertite in tensione elettrica, da un punto di vista fisico. Per fare ciò solitamente è usata la NRZ (Non-Return to Zero) che associa al bit 1 una tensione +1V ed al bit 0 una di -1V.



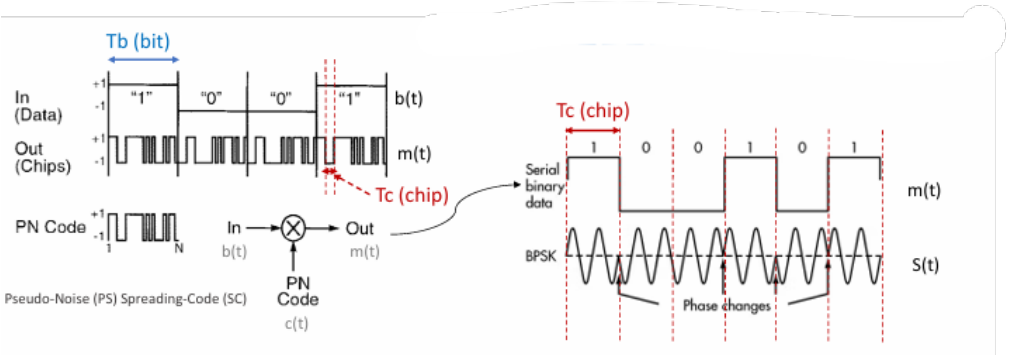
Nella modulazione DS-BPSK il segnale modulante è il segnale ottenuto come risultato della moltiplicazione tra la sequenza binaria (con codifica NRZ) e lo spreading code:

$$m(t) = b(t) \cdot c(t)$$

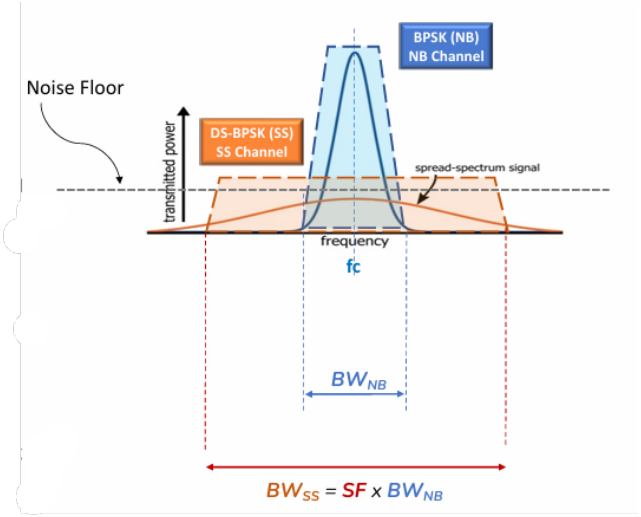


		Polarity of Data Sequence b(t) at time 't'	
		+	-
Polarity of PN Sequence C(t) at time 't'	+	0	π
	-	π	0

Truth table for binary phase modulation (BPSK)



Può accadere che la densità spettrale di potenza del segnale modulato secondo tecniche Spread Spectrum si abbassi al di sotto della soglia del rumore. Ciò non rappresenta un problema se parte della sequenza può essere ancora recuperata.



In fase di ricezione occorre risalire al bit appartenente alla sequenza originaria conoscendo la code word.

$$m(t) \cdot c(t) = b(t) \cdot c(t) \cdot c(t) = b(t) \cdot c^2(t) = b(t)$$

[Il quadrato della sequenza è sempre unitario perché può assumere solo valori +1 e -1]. Questa strategia non è adatta all'IoT perché risulta molto costosa da realizzare per ottenere i risultati delle operazioni in tempi accettabili. Nella realtà la sequenza da ricodificare è del tipo:

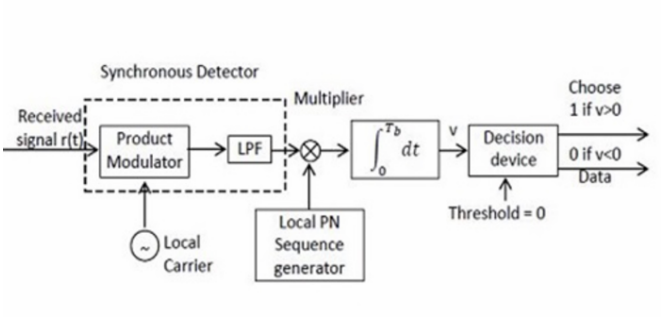
$$b_p \cdot c_p + n$$

ossia soggetta ad una componente aggiuntiva di rumore che introduce, quindi, un'interferenza.

$$(b_p \cdot c_p + n) \cdot c_p = b_p + n \cdot c_p$$

Notiamo come però questo tipo di segnali siano resistenti alle interferenze e difficili da disturbare, anche in presenza di *jamming*.

Una volta ricodificata la sequenza, questa viene integrata in un intervallo di tempo T_b. Se il risultato ottenuto è maggiore di 0 verrà associato il bit 1, altrimenti il bit 0.



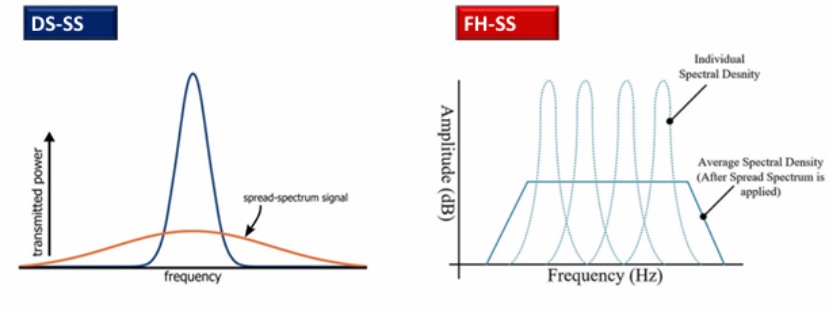
Se la sequenza da ricodificare è del tipo:

$$b_p \cdot c_p + b_i \cdot c_i$$

Dove il secondo contributo rappresenta l'interferenza, si avrà:

$$\int_0^{T_b} (b_p \cdot c_p + b_i \cdot c_i) \cdot c_p \cdot dt = b_p \int_0^{T_b} dt + b_i \int_0^{T_b} c_i \cdot c_p \cdot dt \approx b_p T_b$$

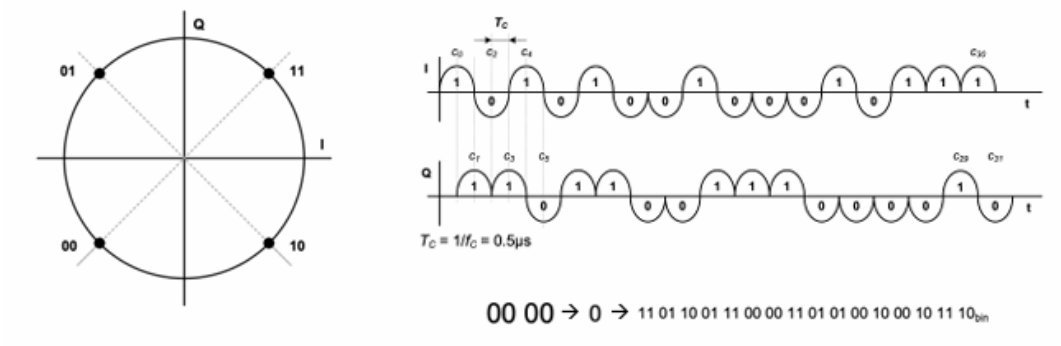
Frequency-Hoping Spread Spectrum (FHSS)



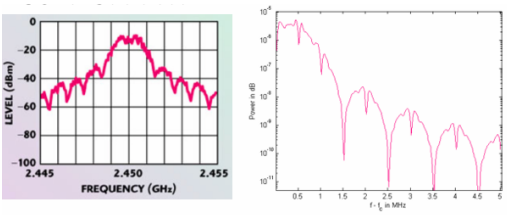
É maggiormente impiegata nel mondo IoT, perché più semplice da implementare (da un punto di vista logico ed elettronico). Prevede che la frequenza del segnale trasmesso, modulato secondo una qualsiasi tecnica a banda stretta, venga cambiata quanto più possibile secondo un pattern noto sia al trasmettitore che al ricevitore.

6.2.2 Pulse Shaping

Dopo aver prodotto i chips, si utilizzano due forme d'onda per modularli: il coseno (componente in fase) ed il seno (componente in quadratura). L'offset (pari alla durata del chip) implica che le due forme d'onda non sono modulate contemporaneamente.

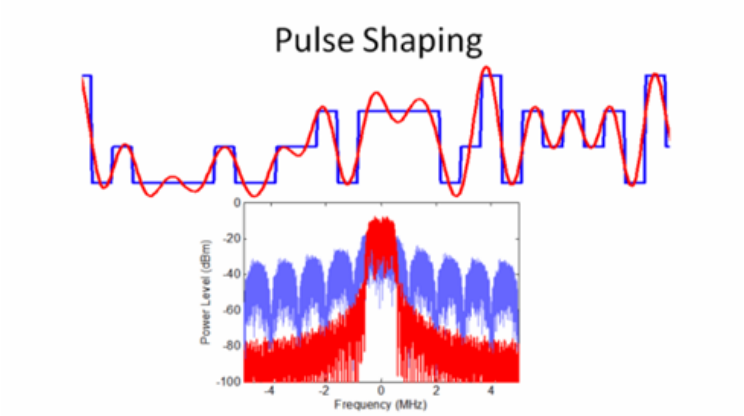


É possibile modellare l'impulso, quindi non trasmettere la forma d'onda così com'è ma applichiamo una procedura (detta *pulse shaping*) in modo da incrementare il livello di potenza del segnale e di conseguenza la sua densità spettrale di potenza

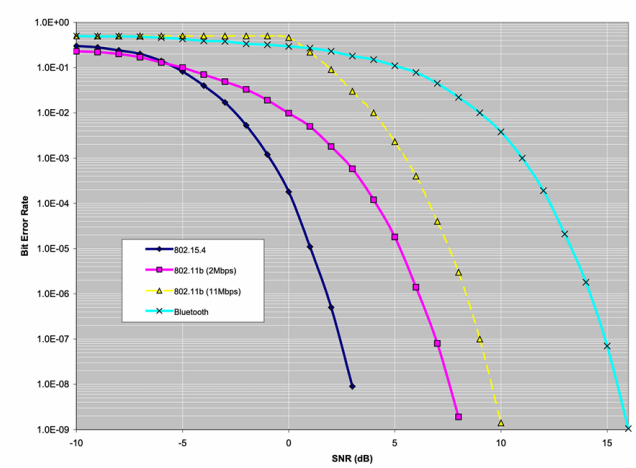


Power spectral density of IEEE 802.15.4

Il pulse shaping permette di ridurre le interferenze tra canali adiacenti ma rende più complessa la sincronizzazione.

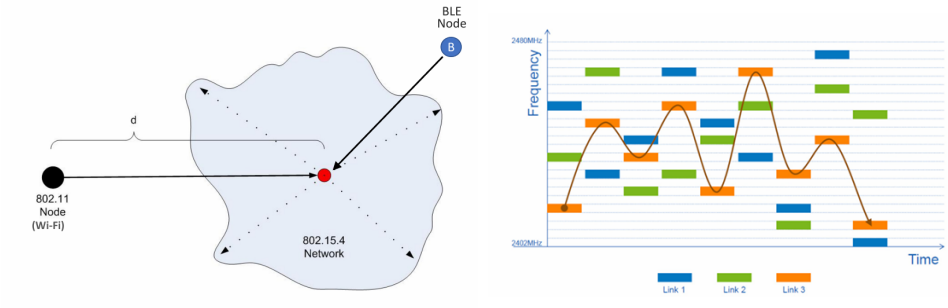


Possiamo notare le differenze, in termini di BER, tra le tecnologie: ZigBee, Bluetooth e Wi-Fi.



Si osserva che lo ZigBee ha prestazioni in trasmissione dai 7 ai 18 dB migliore rispetto alle altre. Ciò si traduce in una copertura che aumenta dalle 2 alle 8 volte in termini di distanza a parità di energia per bit.

6.2.3 Wi-Fi



Nel corso degli anni abbiamo avuto numerose versioni di Wi-Fi. Nelle nostre case ormai è diffusa la versione 6 che raggiunge un throughput massimo di 10 Gbps. Wi-Fi 4 e 5 impiegavano la tecnica di modulazione OFDM (Orthogonal Frequency Division Multiplexing) mentre a partire dal 6 è usata la OFDMA (Orthgonola Frequency Division Multiple Axis).

802.11b/g

Possiamo notare nella successiva figura le risposte in frequenza del Wi-Fi 1 (in alto) e del Wi-Fi 3 (in basso) e come queste differiscono molto tra loro.

L'802.11b ha un segnale che ricorda un seno cardinale. Questo perché lo spettro è sincronizzato, dato che viene usata la DSSS.

Il segnale dell'802.11g, invece, è più o meno piatto e sfrutta la modulazione OFDM.

PHY (MHz)	Frequency Band (MHz)	Geographical Region	Modulation	Channels	Bit rate (Mbps)	Typical Output Power (dBm)
2450	2401-2483	Europe* Japan	DBPSK DQPSK	13	1, 2, 5.5, 11	20
	2401-2473	United States, Canada		11		
	2446-2483	France		4		
	2446-2473	Spain		2		

* Excluding France and Spain

Table 2: IEEE 802.11b Frequency Bands and Data Rates

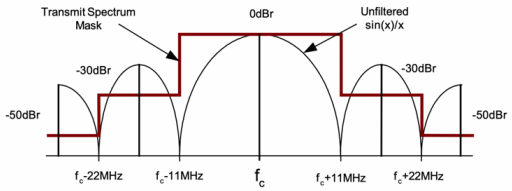


Figure 1: 802.11b Spectral Mask

PHY (MHz)	Frequency Band (MHz)	Geographical Region	Modulation	Channels	Bit rate (Mbps)	Typical Output Power (dBm)
2450	2401-2483	Europe* Japan	DBPSK DQPSK QAM-16 QAM-64	13	1, 2, 5.5, 11, 6, 9, 12, 18, 24, 36, 48, 54	20
	2401-2473	United States Canada		11		
	2446-2483	France		4		
	2446-2473	Spain		2		

* Excluding France and Spain

Table 3: IEEE 802.11g Frequency Bands and Data Rates

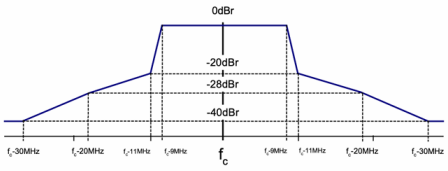
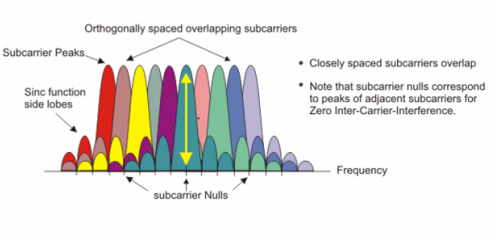


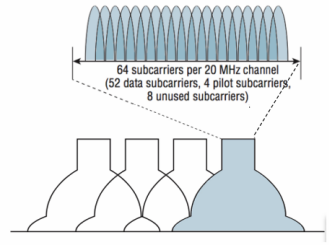
Figure 2: 802.11g Spectral Mask

L'802.11b ha 14 canali nella banda 2.4 GHz (numerati da 1 a 14), ciascuno di ampiezza 22 MHz, separati l'uno dall'altro di 5 MHz. L'802.11g è come il Wi-Fi 1, l'unica differenza sta nell'ampiezza di ciascun canale: pari a 20 MHz.

802.11a/g/n/ac

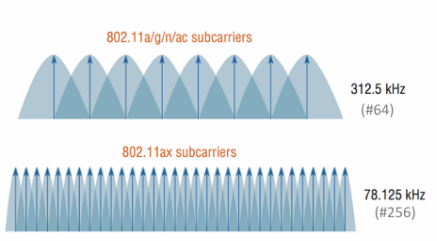
Con lo schema OFDM si può cambiare tecnica di modulazione al fine di proteggere il canale in caso di inferenza. Le tecniche di modulazione generalmente impiegate sono BPSK, QPSK, 16-QAM e 64-QAM.





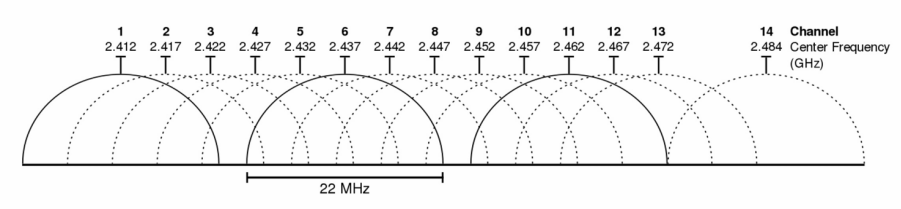
802.11ax

Nelle generazioni più recenti di Wi-Fi i canali sono più stretti. Nella figura notiamo come siamo passati da 64 sotto-canali a 256. Se il canale è più stretto anche il bit rate diminuisce.



6.2.4 Canali 802.11

In Europa è possibile usare 13 canali (su 14 da cui è formato il Wi-Fi, l'ultimo non è utilizzato), negli Stati Uniti, invece, si possono usare solo i primi 11.



Notiamo in Figura 6.4 una differenza nell'ampiezza dei canali tra Wi-Fi 1 e 3.

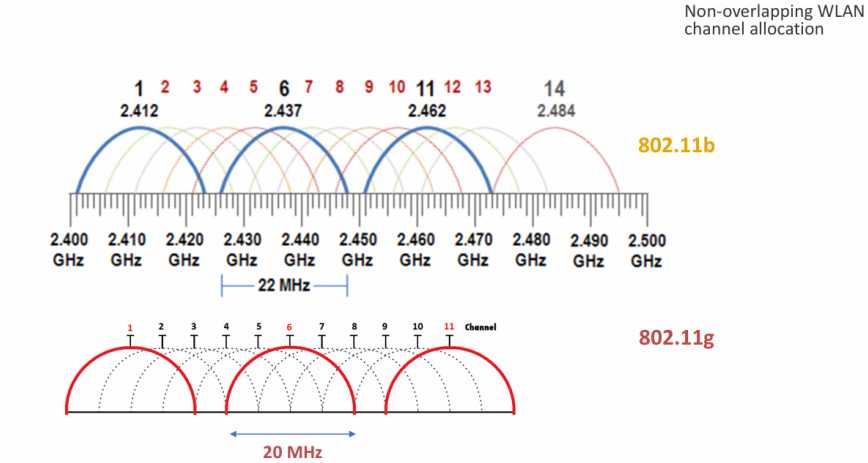


Figura 6.4: Sequenza di canali non sovrapposti

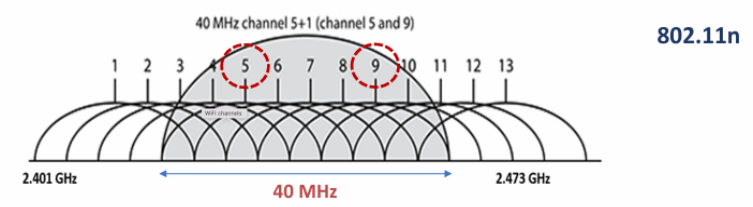


Figura 6.5: Channel bonding in Wi-Fi 4

Dal Wi-Fi 4 è stato introdotto il channel bonding, che permette di formare canali più grandi. Usando un canale da 40 MHz ci si aspetta che anche il bit rate raddoppi.

6.2.5 Coesistenza tra tecnologie

Possiamo notare in Figura 6.7 che un punto di accesso Wi-Fi copre vari canali Bluetooth. Quando il dispositivo Bluetooth salta dal canale 4 al canale 16 si accorge dell'interferenza e per evitarla salta il doppio.

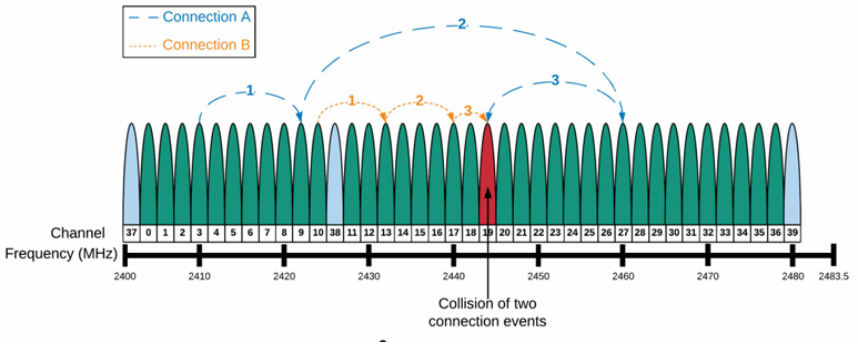


Figura 6.6: Coesistenza tra Bluetooth e Bluetooth

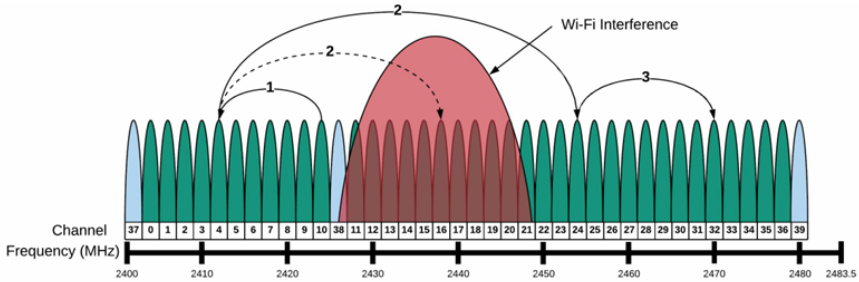


Figura 6.7: Coesistenza tra Wi-Fi e Bluetooth

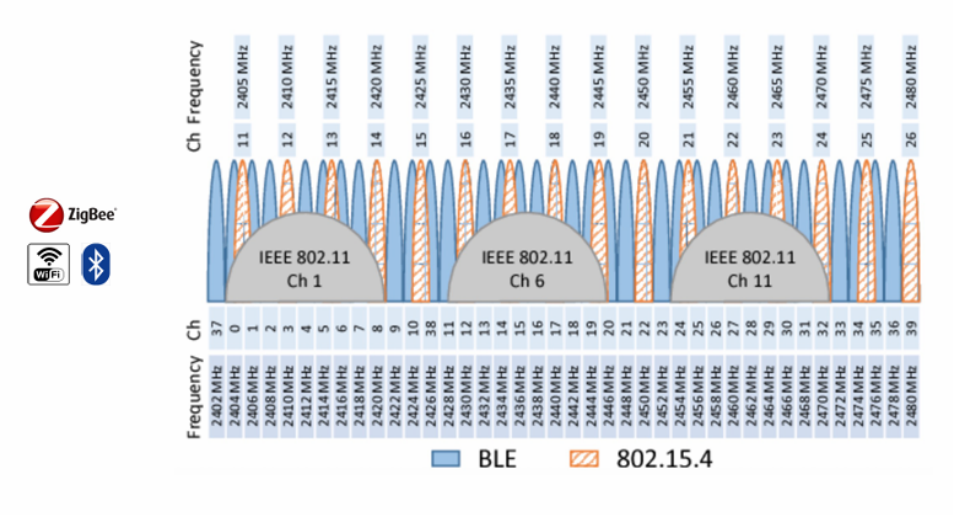
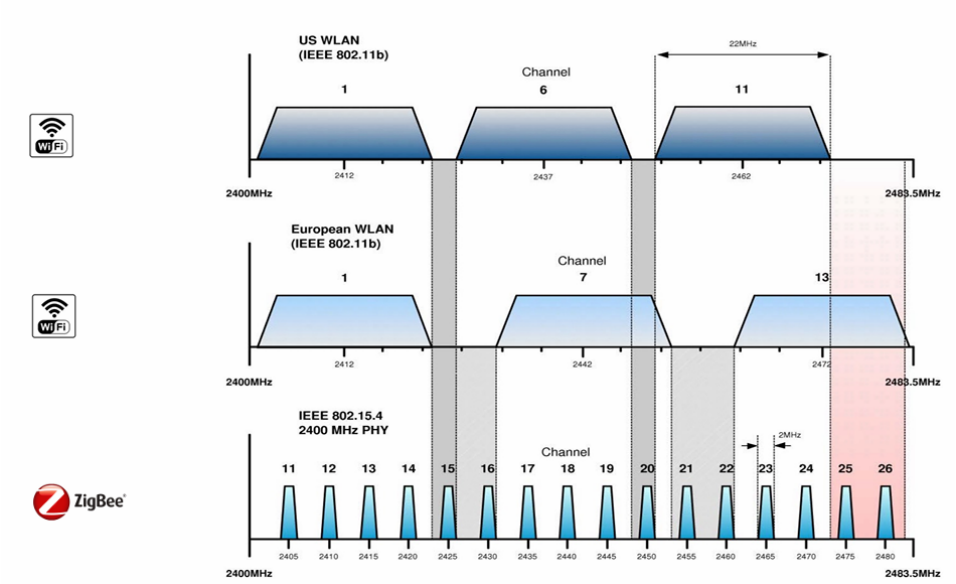


Figura 6.8: Coesistenza tra ZigBee, Wi-Fi e Bluetooth

In Europa è conveniente scegliere i canali 15, 16, 21 e 22 per ZigBee; mentre negli Stati uniti sceglie 15, 16, 25, 26. Per realizzare una buona coesistenza tra Wi-Fi e ZigBee è necessaria una distanza tra le rispettive frequenze centrali di 7 MHz. Con una distanza tale la sovrapposizione dei canali può essere trascurata (perché il Wi-Fi usa OFDM mentre ZigBee DSSS). Il Bluetooth, come abbiamo visto, stabilisce una nuova sequenza di hopping per evitare le interferenze. Quando ciò accade il Wi-Fi ne risente (si abbassa il data rate effettivo).



Network Layer Frequency Agility

In caso di interferenze troppo forti (rispetto ad una soglia predeterminata), il dispositivo può spostarsi su un altro canale.

ZigBee prevede un servizio chiamato LQI (Link Quality Indicator) che valuta lo stato corrente del canale, analizzandolo pacchetto per pacchetto.

6.3 Livello MAC

L'IEEE 802.15.4 ha un'architettura LR-WPAN (Low Rate Wireless Personal Area Network).

Il livello MAC (2) dello stack protocollare è responsabile dei seguenti compiti: associazione, dissociazione, acknowledge frame delivery, meccanismi di accesso al canale, frame validation, Guaranteed Time Slot (GTS) management e beacon management.

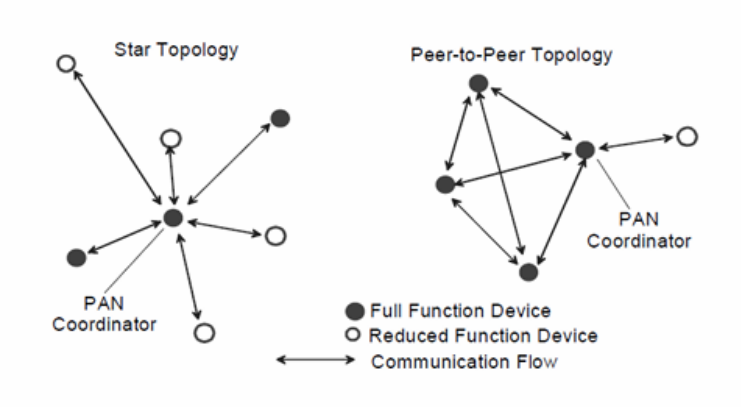
Sono possibili diversi data rate: 851 Kb/s, 250 Kb/s, 100 Kb/s, 40 Kb/s e 20 Kb/s. Di nostro interesse sono i 250 Kb/s, ovvero il massimo data rate concesso per la banda 2.4 GHz.

Tipologie di nodi

Full-Function Device (FFD): può parlare con qualsiasi altro dispositivo, può trasmettere messaggi (ed in caso viene detto coordinator), quando è responsabile dell'intera rete è chiamato PAN coordinator. [Ogni rete ha bisogno di almeno un FFD che lavori come coordinatore].

Reducted-Function Devidce (RFD): può comunicare solo con FFDs (uno alla volta), non può mai agire da coordinatore.

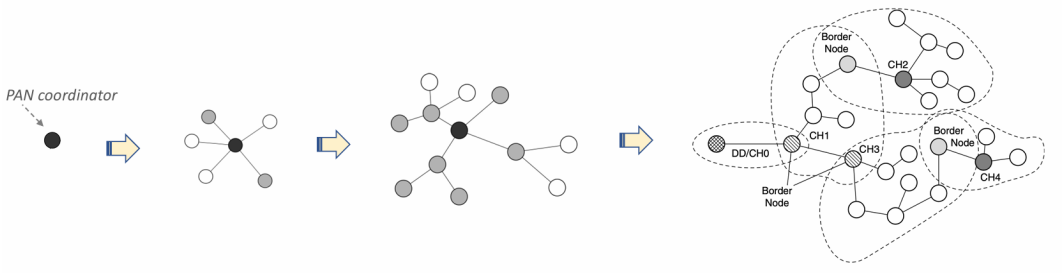
Topologie di rete



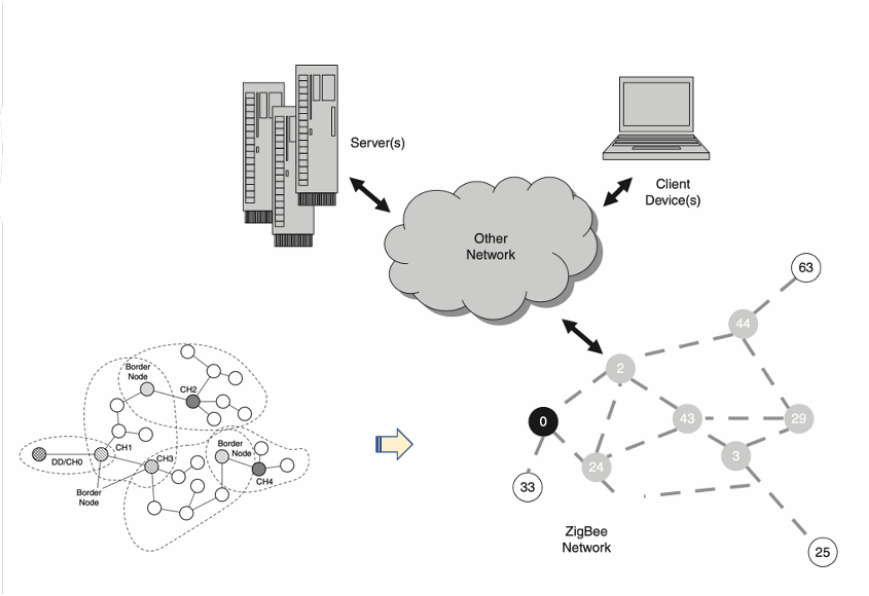
- **Star networks (strutturata)**: ha origine quando un FFD decide di creare una propria PAN, dichiarandosi coordinatore della stessa (dopo aver scelto un identificativo unico per la PAN). Altri dispositivi possono unirsi alla rete. Il coordinatore corrisponde al nodo centrale.
- **Peer-to-Peer (point-to-point) networks**: i dispositivi possono formare schemi arbitrari di connessioni. É una base per le reti ad hoc (autogestione e organizzazione). Notiamo come il cordinatore PAN è presente anche per queste reti. Da un punto di vista logico non è necessario ma viene utilizzato per garantire la sicurezza e l'accesso ad internet tramite gateway.

Self forming networks

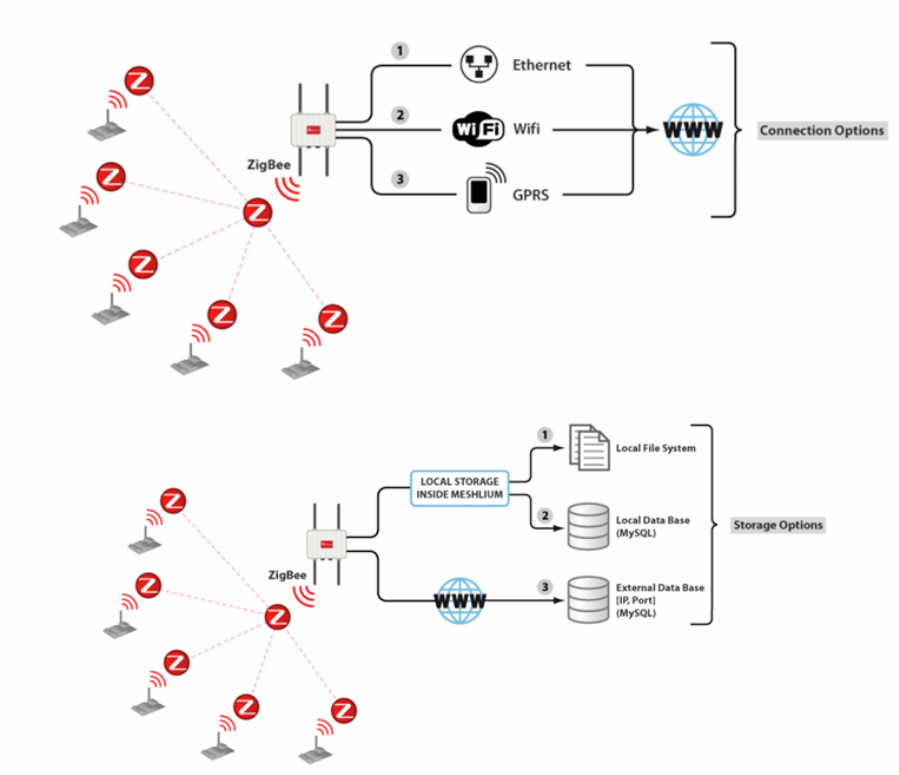
Nelle reti mesh il primo dispositivo FFD che inizia a comunicare sarà il coordinatore PAN, gli altri dispositivi si uniscono alla rete inviando richieste di associazione. La rete si auto-configura in base allo scenario.



Gateway



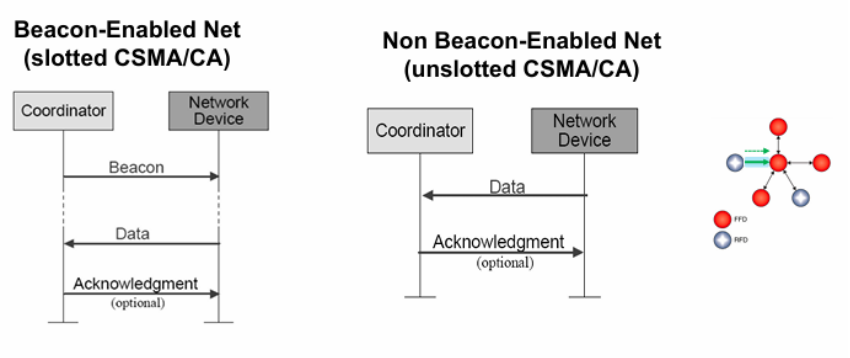
Il gateway è un FFD (quindi non un dispositivo ZigBee), solitamente molto costoso. Il suo target principale è quello di tradurre i protocolli e permettere ad una rete ZigBee di comunicare attraverso Internet.



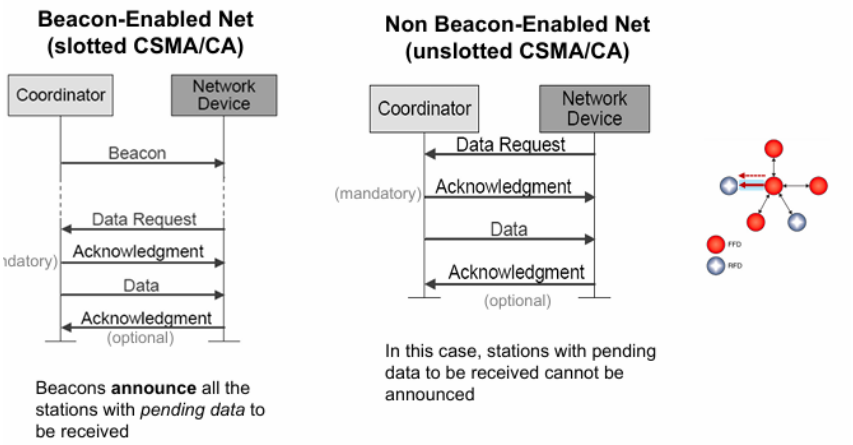
6.3.1 Data transfer

Si distinguono tre tipologie di trasferimento dati:

1. *da device a coordinator* - si possono avere due approcci (beacon based e non-beacon based). Nel primo caso il coordinator invia un beacon con il quale effettua un advertising di tutti i suoi figli, ossia i nodi che vi dipendono. Una volta ricevuto un beacon, il device connesso può trasmettere i dati. Dopo ciò il coordinator può decidere se rispondere o meno con acknowledgment. Nel secondo caso, invece, il device trasmette i dati senza sapere se il coordinator è attivo o meno. L'accesso ai dati è gestito, quindi, attraverso un rilevamento del canale slotted nel beacon based ed unsloted nel non-beacon based.



2. *da coordinator a device* - anche in questo caso abbiamo i due apporcci beacon based e non-beacon based. Nel primo caso il coordinator trasmette per primo il beacon per far capire al device che è attivo. Il dispositivo che vuole ricevere dei dati manda una richiesta ed il coordinator, una volta ricevuta, risponde obbligatoriamente con un acknowledgement e poi trasmette i dati richiesti. Il device potrebbe rispondere con un acknowledgement. Nel secondo caso i dispositivi avanzano sempre una richiesta al coordinator, il quale risponde con un acknowledgement (obbligatorio) e poi con i dati. Il device può rispondere con un acknowledgement (opzionale).



3. *peer-to-peer*.

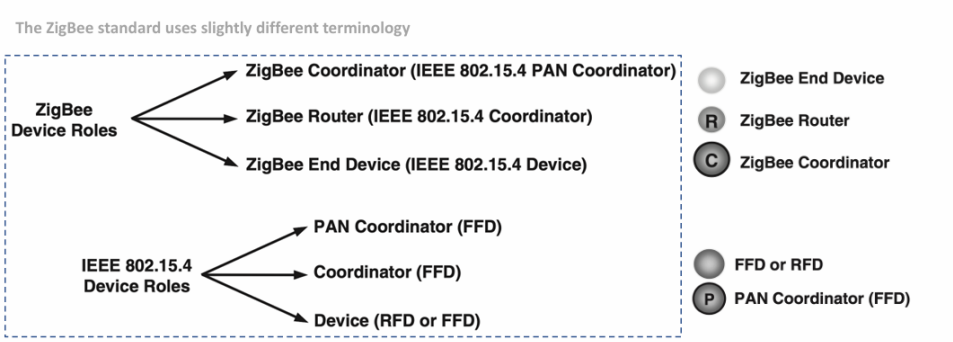
6.3.2 Data verification

La verifica dei dati è svolta grazie ad una Frame Check Sequence (FCS) a 16 bit basata su Cyclic Redundancy Check (CRC) per rilevare eventuali errori nel pacchetto dati.

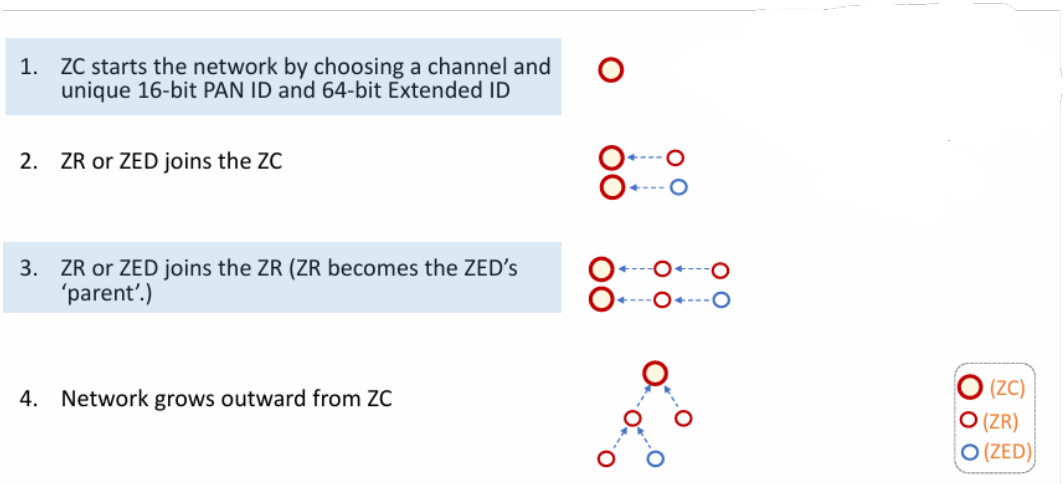
Il ricevitore esegue un calcolo indipendente dalla FCS e lo confronta con quello ricevuto (operazione che prende il nome di *checksum* e viene eseguita su ogni bit). In caso di corrispondenza viene inviata una conferma di ricezione all'autore del messaggio. In mancanza di un ACK viene ripetuta la trasmissione.

6.3.3 Node Types

- *ZigBee Coordinator (ZC)*: router speciale che forma la rete. Se ne distingue uno per rete e rappresenta un FFD con il ruolo di PAN coordinator.
- *ZigBee Router (ZR)*: un device che può instradare l'informazione e che resta sempre attivo. Rappresenta un FFD con il ruolo di coordinator.
- *ZigBee End-Device (ZED)*: un device con funzionalità ridotte che richiede il ZC e ZR per partecipare alla rete.



6.3.4 Creare una rete



6.3.5 Unirsi ad una rete

Un dispositivo Router o End-Device scansiona il canale per scoprire tutti i nodi locali (distanti 1 hop) ai quali connettersi, i quali possono essere esclusivamente Router o Controller. Il dispositivo che ha eseguito la scansione sceglie un dispositivo target e sottomette una richiesta, se il dispositivo target accetta la richiesta risponde con una conferma (che includerà l'indirizzo di rete e le principali caratteristiche di essa).

Il processo opposto, di dissociazione, è più semplice. Il device invia un messaggio al router avvisandolo di non voler più far parte della rete in questione.

[Per i device IoT in modalità sleep è complicato unirsi ad una rete ed effettuare uno scambio di messaggi].

6.3.6 Scanning mode

La procedura di scanning è costruita su funzionalità di livello 1 quali: Energy Detection (ED), Link Quality Indicator (LQI) e channel switching.

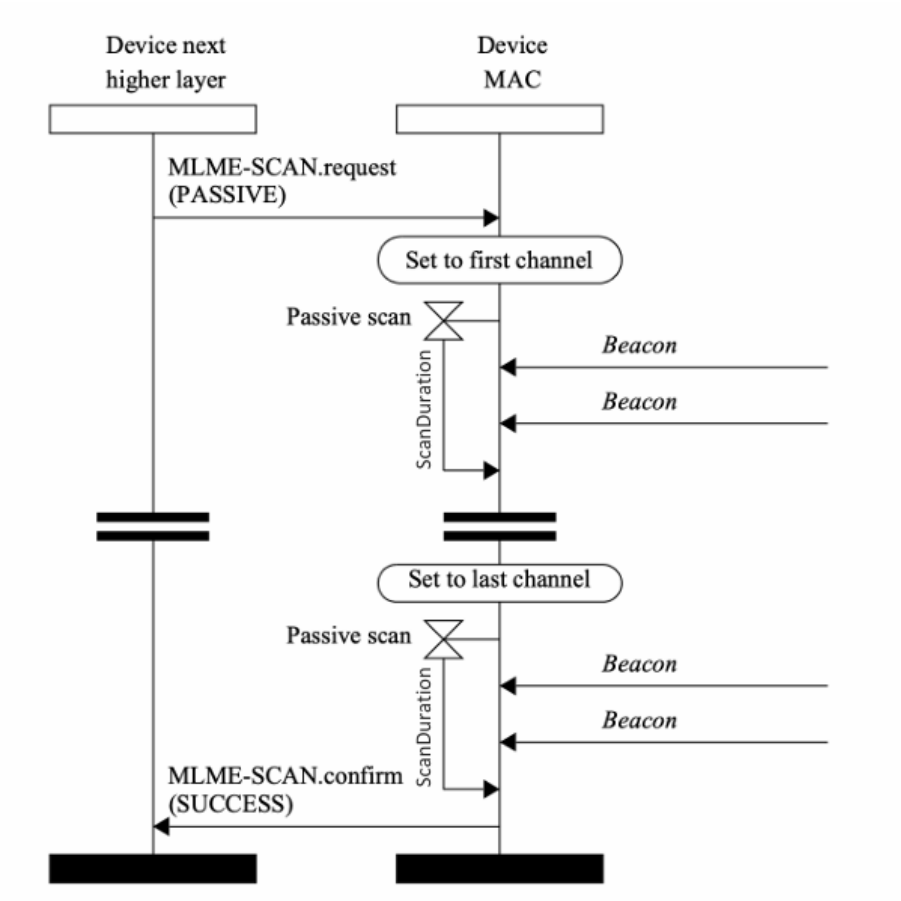


Figura 6.9: Passive scan message sequence chart

Nella procedura di scanning **passiva** il livello più alto del device invia un messaggio al livello più basso (MAC) dopo aver deciso di connettersi alla rete.

Ricevuto il messaggio, il MAC layer avvia la procedura di scanning (imposta la frequenza radio nel livello fisico al primo canale ed ascolta quest'ultimo per la durata dello scan). Se riceve qualche beacon allora ottiene le informazioni volute. Successivamente cambia canale attendendo ulteriori beacon (eventuali). Viene deciso se il processo di scan ha avuto successo o meno (a seconda se ha ricevuto almeno un beacon) inviando una conferma al livello superiore che sceglierà se connettersi o meno alla rete.

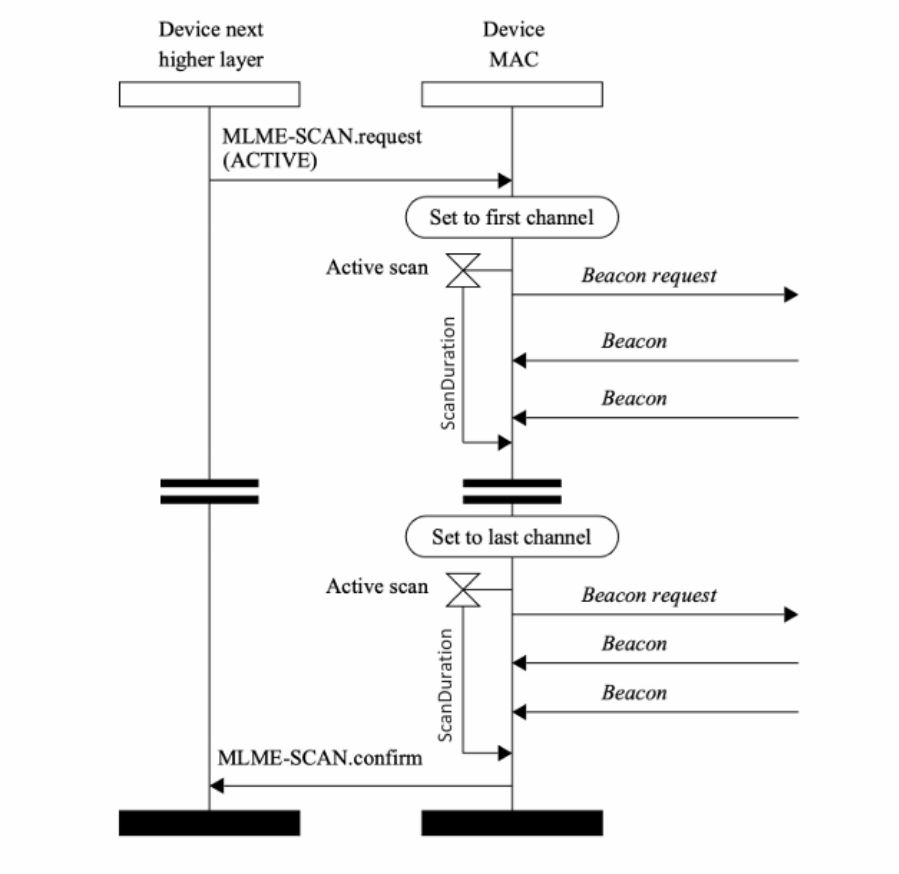


Figura 6.10: Active scan message sequence chart

Nell'**active** mode abbiamo il medesimo diagramma, con l'aggiunta del *beacon request* (con la quale è possibile ottenere un meccanismo più efficiente, perché gli altri dispositivi sono forzati a rispondere).

Una volta stabilito a che router connettersi viene trasmesso un pacchetto di Joining Request. Se a seguito di un certo intervallo di tempo il router accetta la richiesta, trasmette un pacchetto di Joining Response in risposta.

[Entrambi i dispositivi hanno un timeout al seguito del quale dichiarano se quest'ultima è fallita ed un nuovo processo di scanning può iniziare].

Nella rete i dispositivi potrebbero non trasmettere alcun beacon. In tal caso è possibile trasmettere un Orphan Notification Command e si attende un riallinameno del coordinator.

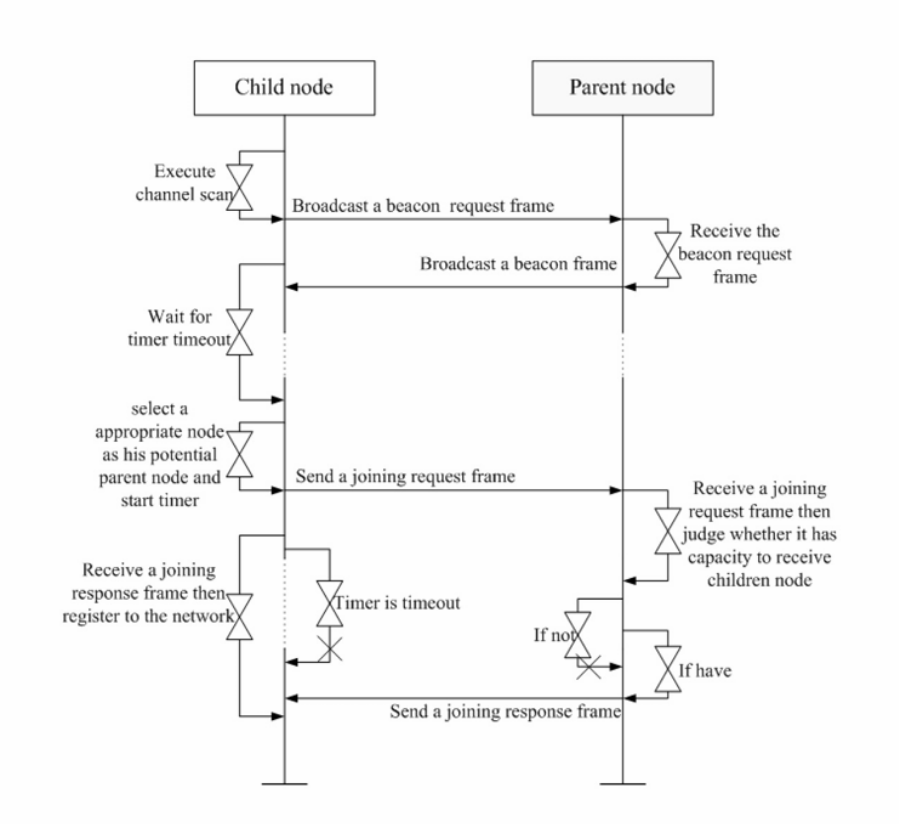
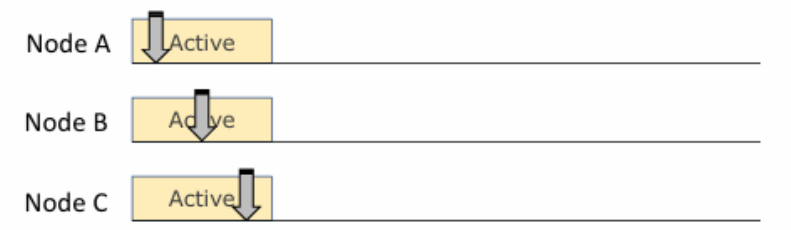
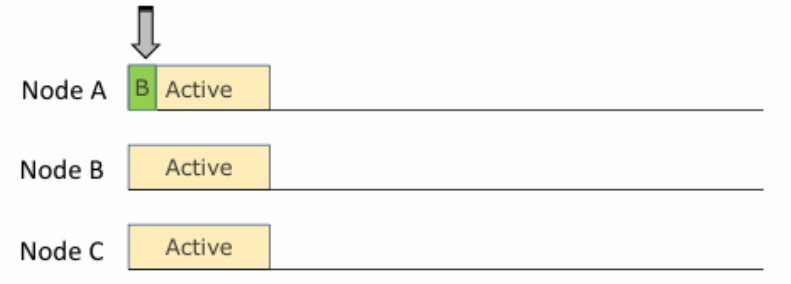


Figura 6.11: Joining request/response message sequence chart

6.4 Power Saving Algorithms



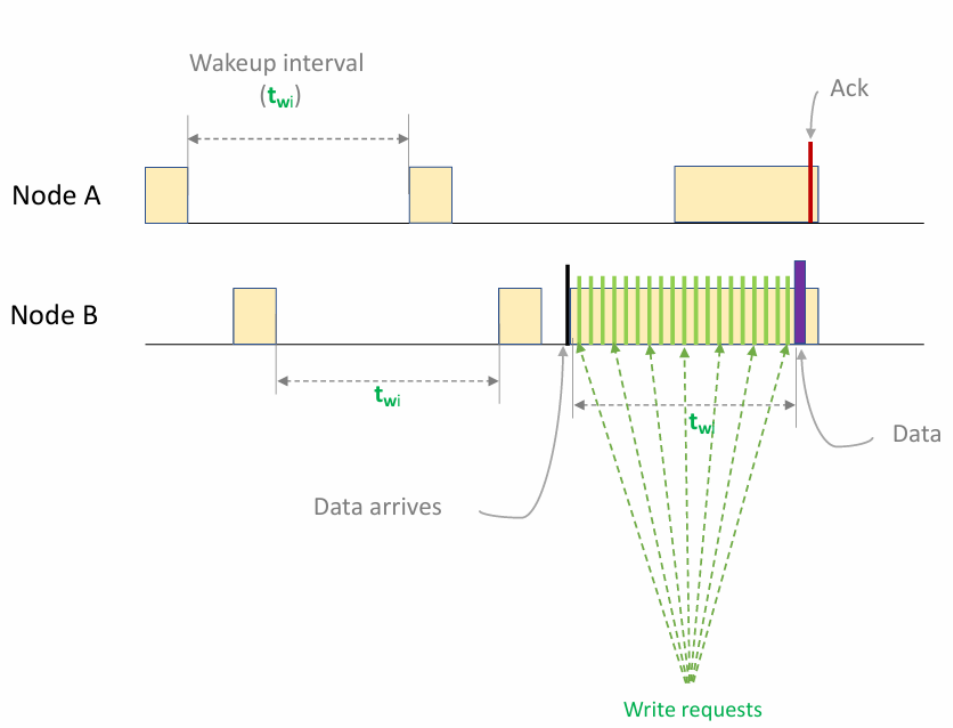
Il primo approccio possibile prevede che tutti i nodi della rete siano sincronizzati in modo da condividere lo stesso schedule di sleep e wake time.



Un secondo approccio prevede l'ausilio dei beacon. Un dispositivo inviamo un beacon agli altri dispositivi e li informa che è attivo e può ricevere pacchetti; i dispositivi in ascolto possoo eventualmente decidere di trasmettergli delle informazioni.

6.4.1 Long Preamble Emulation (LPE)

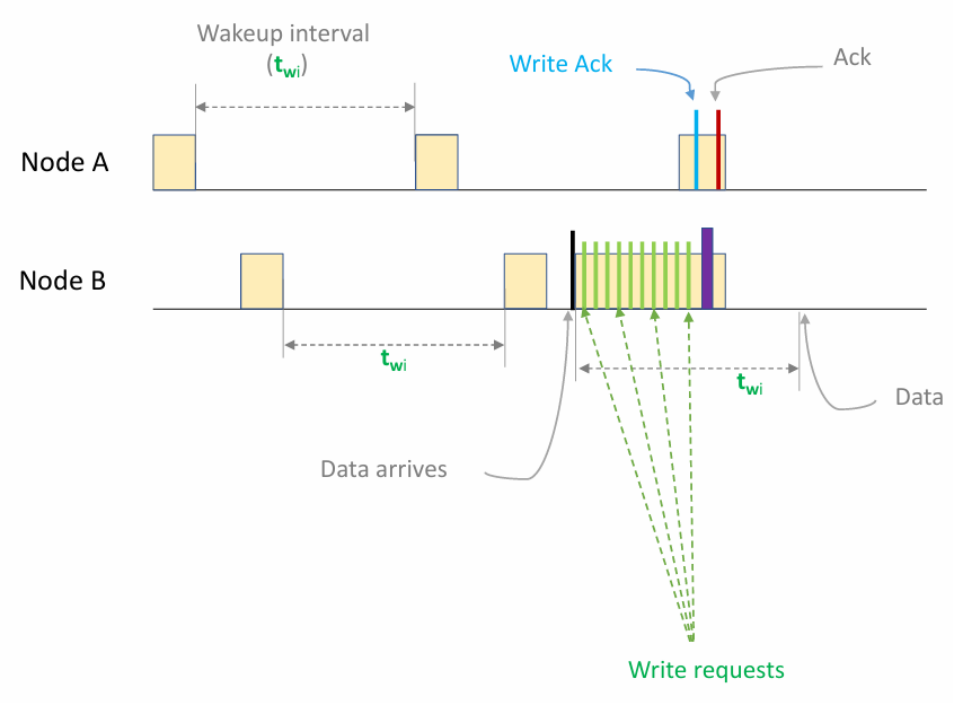
Protocollo non-beacon nel quale i nodi condividono ed utilizzano tutti lo stesso wake-up interval. Notiamo nell'esempio della figura seguente come i nodi A e B abbiano degli schedule diversi. Una volta arrivati dati al nodo B (sbarretta nera), questo si attiva ed inizia ad inviare dei messaggi sul canale (write requests) per una durata pari al suo wake-up interval (perché non sa quando il nodo A è attivo). Una volta sveglio il nodo A riceve la write request e resta attivo per ricevere anche i dati ai quali risponde con un ack.



Non è un approccio efficiente perché l'informazione di interesse (contenuta nella sbarretta viola) è molto breve rispetto al periodo di attività. Viene sprecata, quindi, molta energia.

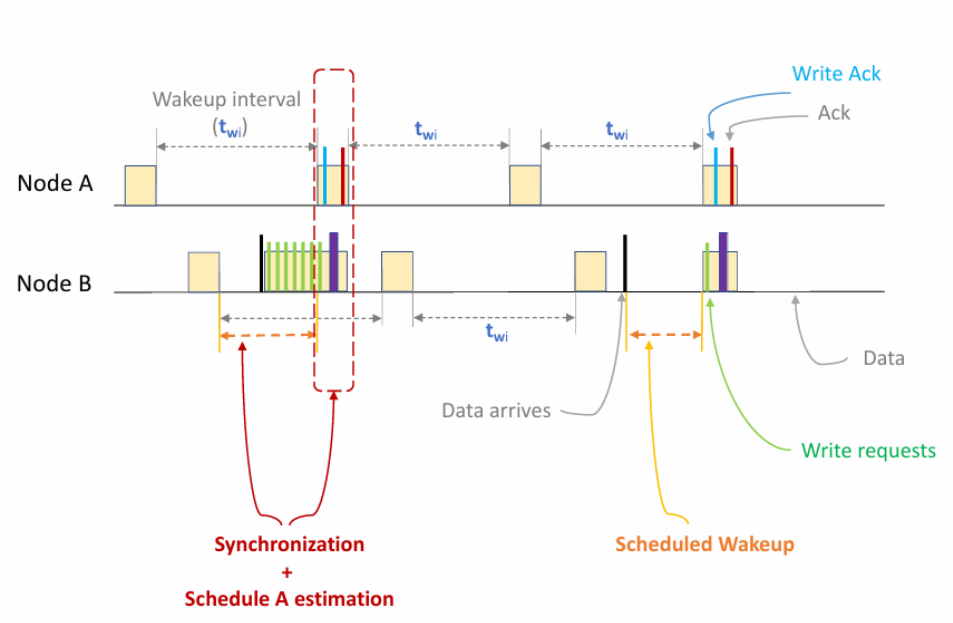
6.4.2 Long Preamble emulation with ACK (LPA)

Una seconda versione del precedente protocollo che a seguito della ricezione della prima write request, una volta sveglio il nodo A, questo risponde immediatamente con un ack. In questo modo il nodo B può risparmiare energia ed il nodo A rimane in attesa per meno tempo.



6.4.3 Long Preamble emulation with ACK after local Synch (LPAS)

Un'ulteriore versione del medesimo protocollo. Dopo lo scambio, eseguito esattamente come per LPA, il nodo B riceve una nuova informazione ed ormai ha compreso il wake-up pattern del nodo A, non ha quindi necessità di trasmettere molte write requests ma si sveglierà direttamente assieme ad A e ne invierà una sola (questo viene definito una 4-way handshake). É la variante più efficiente perché il nodo A e il B sono attivi per il minor tempo possibile. [ZigBee solitamente non fa uso di questo protocollo, ne sceglie uno più semplice].

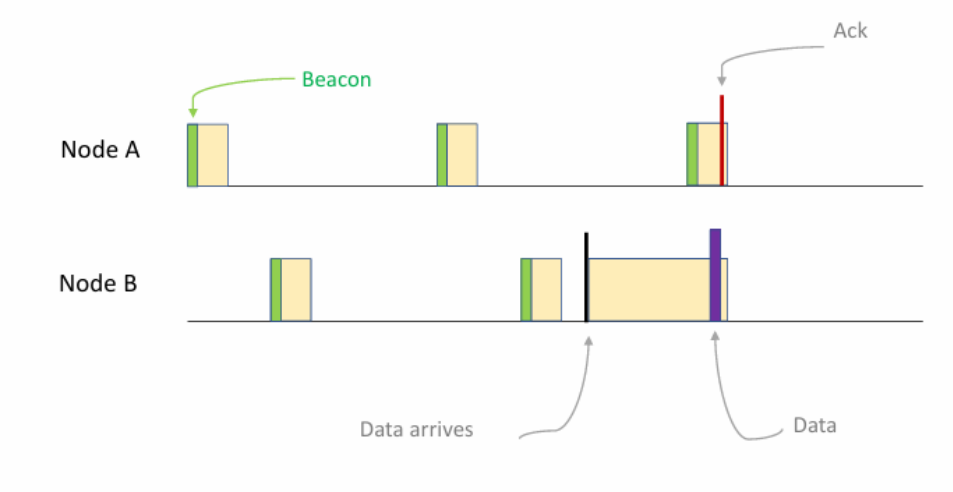


6.4.4 Non-Beacon Tracking (NBT)

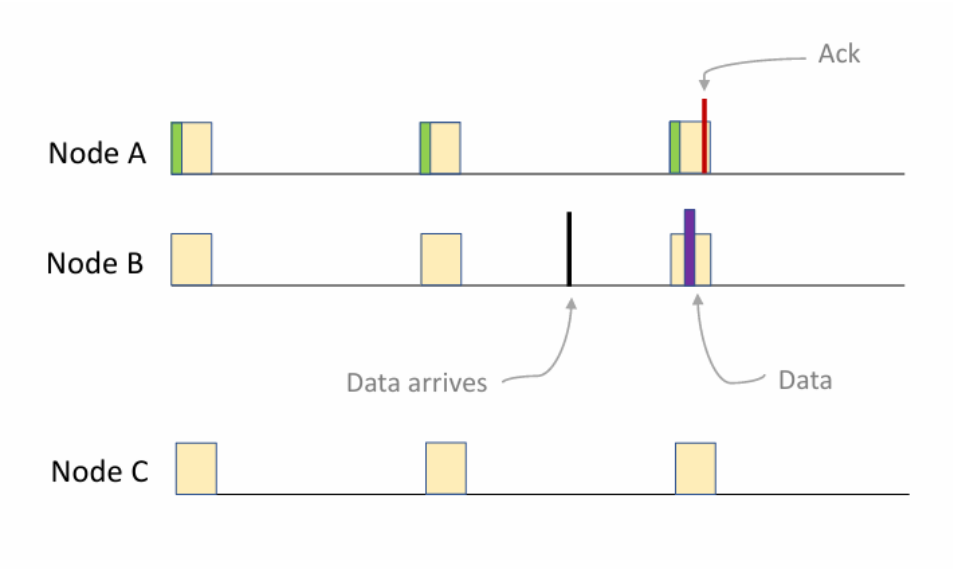
É una possibilità di utilizzo dei beacon per la gestione della comunicazione. Notiamo dalla figura seguente che il nodo A è un controller ZigBee (per il suo pattern), il nodo B funziona invece come un ricevitore.

Quest'ultimo non conosce il nodo A quindi resta in ascolto attendendo un beacon (prima operazione eseguita al risveglio dal nodo A). Dopo averlo ricevuto può trasmettere i dati e ricevere l'ack.

In questo modo non vengono trasmesse delle write requests ma è possibile inviare i dati una volta sola direttamente. I consumi sono equivalenti circa agli altri metodi (le differenze dipendono dalla tecnologia: alcuni dispositivi possono consumare più in ricezione che in trasmissione).



6.4.5 Beacon Tracking (BT)



Il nodo A anche in questo caso è il coordinator ma a differenza di prima gli altri nodi sono ad esso collegati (conoscono il suo pattern e si svegliano assieme).

Questo modo di operare è del tutto sincronizzato e conviene dal punto di vista dei consumi.

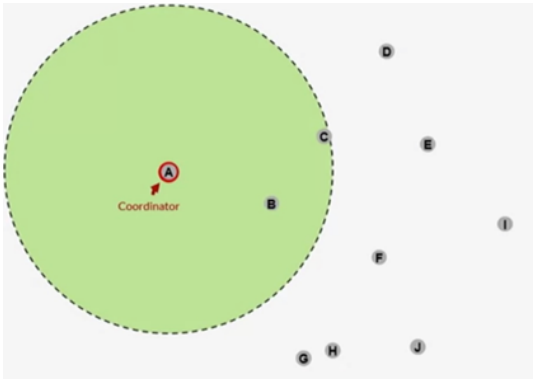


Figura 6.12: Single-hop

Nell'esempio fatto sopra ci trovavamo in una situazione 1-hop in cui tutti i nodi sono all'interno dell'area di copertura del coordinator e possono quindi ricevere i suoi beacon. Nella realtà capita che i device si trovano al di fuori di quest'area.

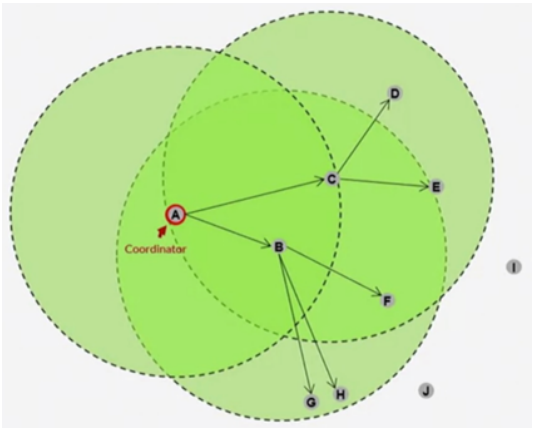


Figura 6.13: Multi-hop

Una soluzione a questo problema prevede che anche gli altri nodi inizino ad inviare i beacon, si forma quindi un *association tree*.

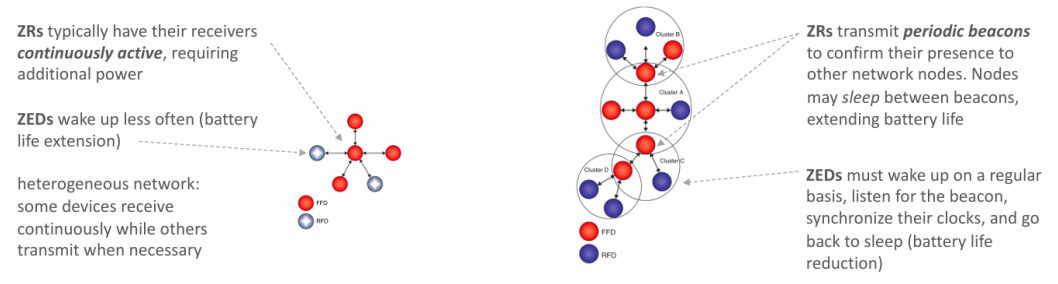
6.4.6 Global Synchronization (GS)

É simile al BT ma la trasmissione dei beacon non è obbligatoria.

6.4.7 Consumo Energetico

Gli algoritmi appena visti sono quelli dell'IEE 802.15.4 ma solo alcuni di essi sono selezionati da ZigBee, il quale può scegliere tra due possibilità:

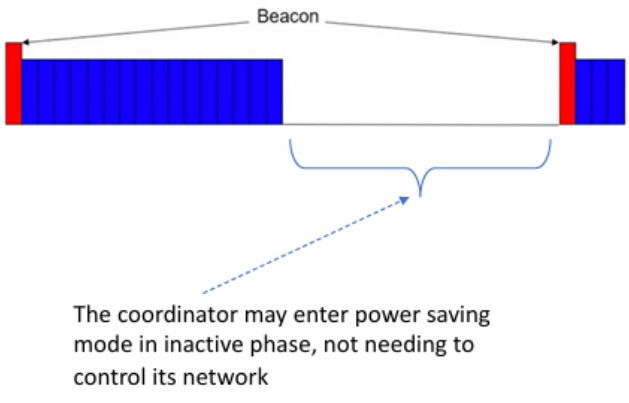
- *non-beacon-enabled*, alcuni dispositivi sono più attivi di altri che passano maggior parte del tempo in modalità sleep (i consumi sono asimmetrici). É più semplice e statico. Unslotted CSMA/CA (no GTSs).
- *beacon-enaboled*, i nodi sono attivi solo quando un beacon è trasmesso. É più complesso e dinamico. Slotted CSMA/CA (+ GTSs).



Se si ha un solo coordinator ma numerosi dispositivi la migliore soluzione è quella non-beacon, al contrario se si vuole una rete fortemente connessa che migliori l'area di copertura è preferibile la beacon-enabled.

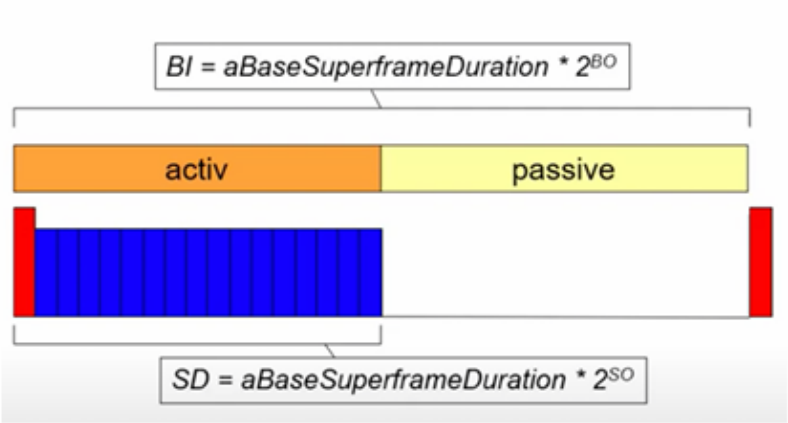
6.4.8 Beacon Mode and Superframe Structure

ZigBee prevede che il tempo sia organizzato in SuperFrame (SF). Una struttura di questo tipo è opzionale (possiamo scegliere tra beacon e non-beacon mode, ma la prima è più diffusa). Il coordinatore invia i beacon (rappresentati in rosso), i quali hanno lo scopo di indicare l'inizio della fase di attività e di annunciare la durata della fase di attivazione (rappresentata in blu). Durante la fase d attività il coordinatore è in attesa di possibili pacchetti.



Possiamo quindi distinguere una fase attiva e una passiva. La SD (Superframe Duration) è la porzione attiva del bacon interval dove tutti i dispositivi sono attivi (trasmettono o ricevono). BI (Beacon Interval). Il coordinatore può

scegliere due parametri: Beacon Order (BO) e Superframe Order (SO); questi sono dei numeri interi compresi tra 0 e 14.

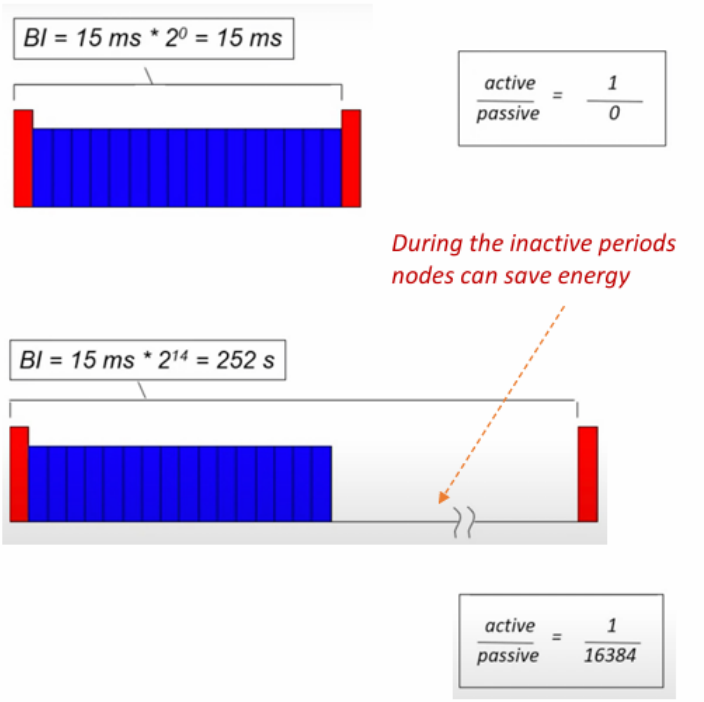


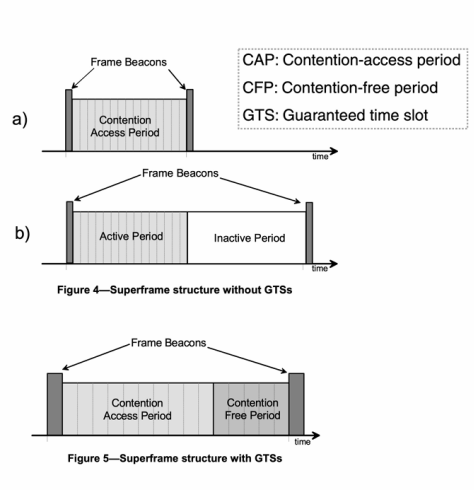
Built-in trade-off mechanisms that give the ZEDs the option of prolonging network lifetime at the cost of lower throughput or higher transmission delay.

Nello specifico:

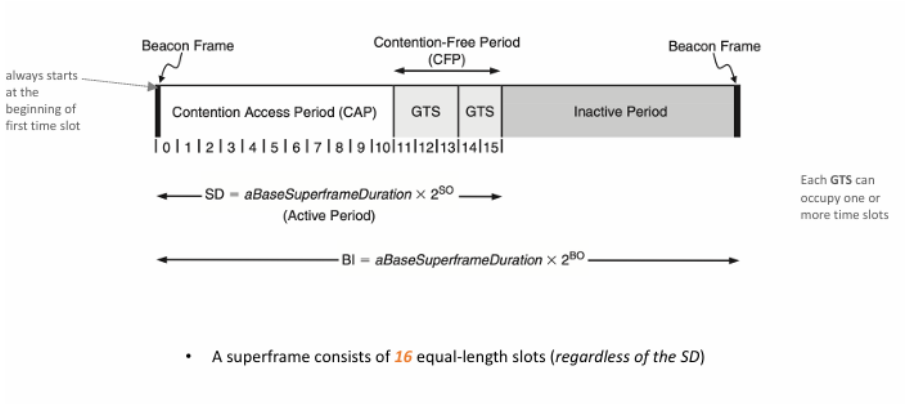
$$0 \leq SO \leq BO \leq 14$$

É preferibile scegliere il BI più lungo possibile, invece, si sceglie una SD piccola se sono connessi pochi dispositivi e grande quando i dispositivi gestiti dal coordinatore sono tanti.

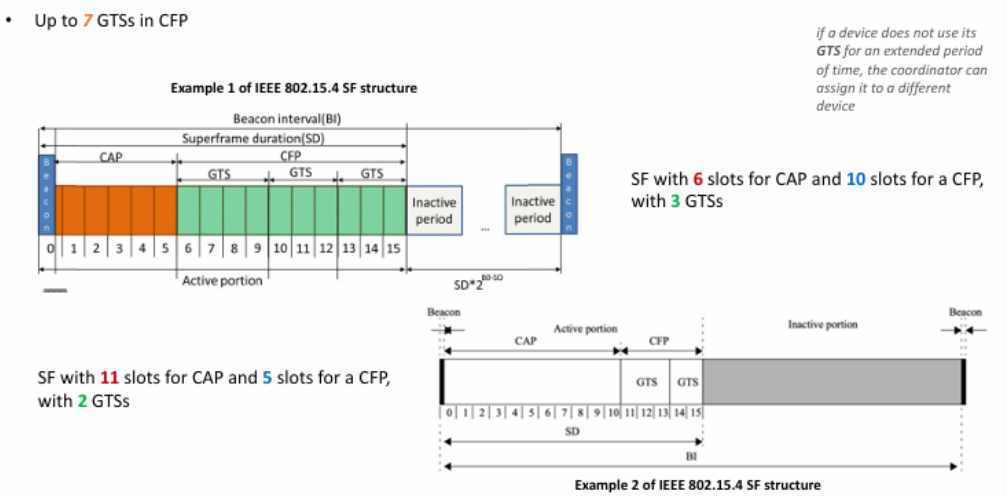




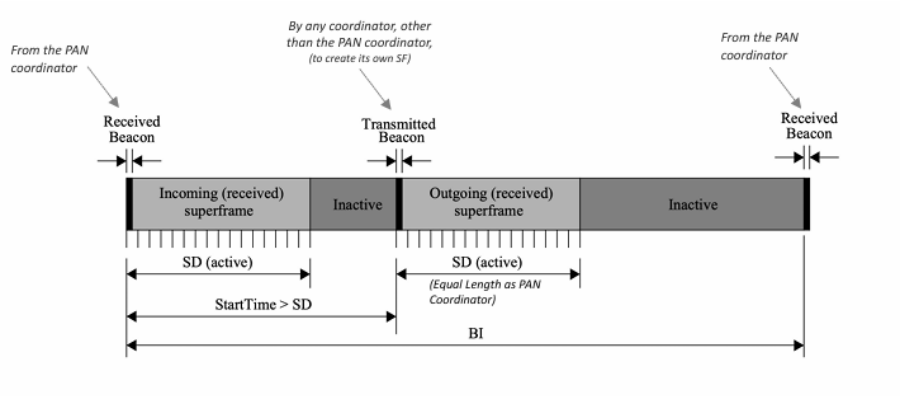
Il CFP è opzionale, contiene degli slot che possono essere riservati a determinati dispositivi. Per supportare i sevizi in tempo reale è possibile allocate il GTS durante la CFP (usato quando i dispositivi usati hanno un pattern specifico).



Il superframe è diviso in 16 slots. I dispositivi solitamente trasmettono durante il CAP. Nella figura in basso possiamo notare invece il modo in cui ZigBee integra il beacon tracking.



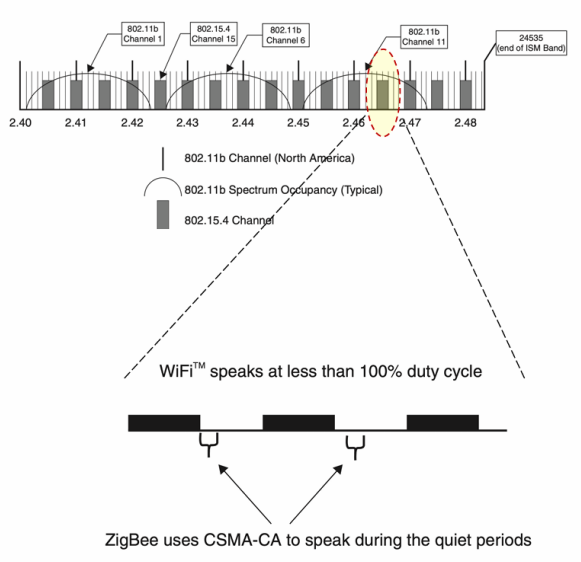
Un router può decidere di creare il suo superframe, quindi invia il suo beacon. Il secondo coordinatore può trasmettere il suo beacon solo durante il periodo inattivo. Un altro vincolo imposto da ZigBee è che la SD deve avere la stessa lunghezza del PAN coordinator.



6.5 CSMA/CA

Per evitare che due o più stazioni trasmettano nello stesso momento, come per il wi-fi, lo standard IEEE 802.15.4 attua la procedura Carrier Sense Multiple Access with Collision Avoidance. Viene ascoltato il canale, se questo è un idle, ossia è libero (nessun altro dispositivo sta trasmettendo), può iniziare una trasmissione. Se invece il canale è occupato si deve aspettare un certo intervallo di tempo casuale, noto come back-off time, in modo da evitare di collidere com la trasmissione in corso.

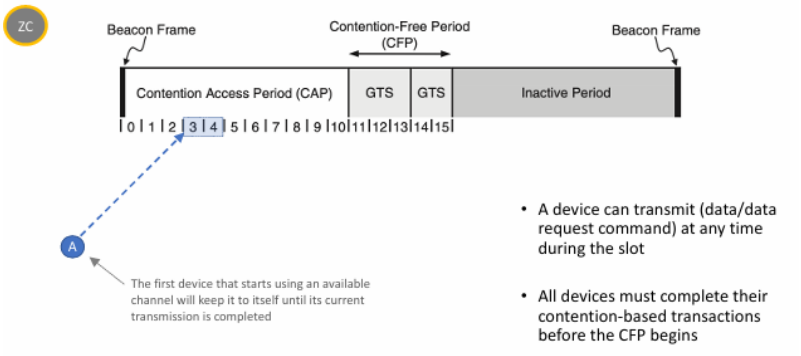
6.5.1 Clear Channel Assessment (CCA)



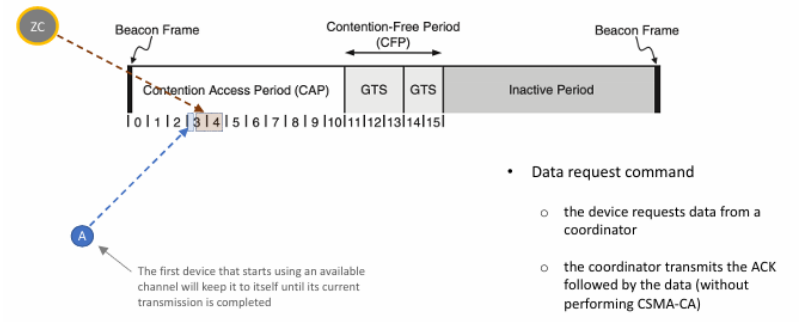
Eseguita ogni qualvolta un dispositivo desidera trasmettere, per assicurarsi che il canale non sia in uso.

6.5.2 Slotted (Beacon Mode)

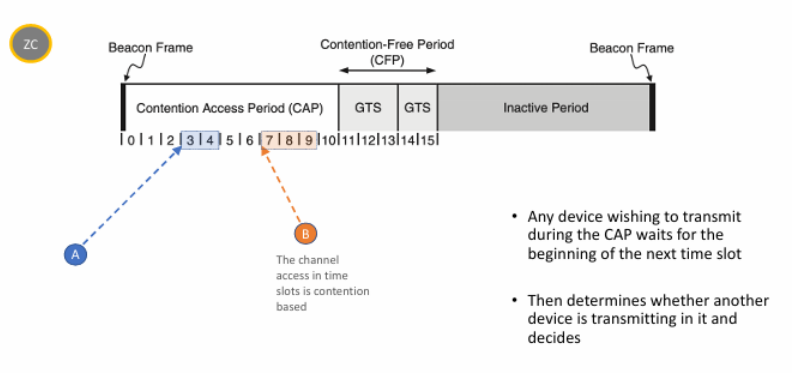
Si supponga che un dispositivo A voglia trasmettere un pacchetto al suo coordinator, compete per il canale ed in particolare lo occupa per gli slot 3 e 4 del CAP (Contention Access Period), In generale un dispositivo può trasmettere in qualsiasi momento nello slot occupato, tutti i dispositivi però devono concludere le transazioni contention-based prima che il Contention-Free Period (CFP) inizi.



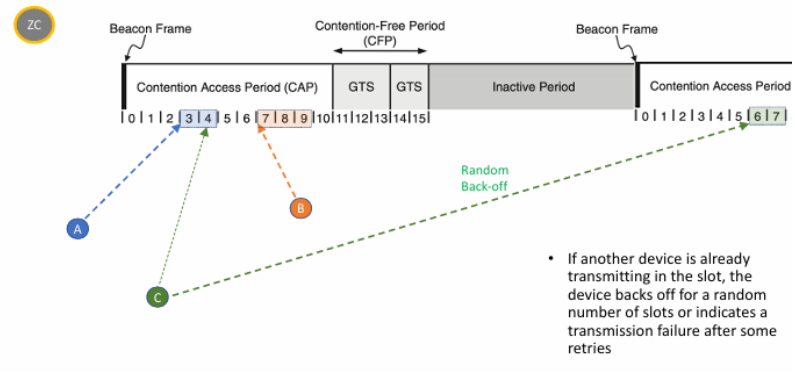
Il dispositivo A potrebbe non avere un pacchetto da trasmettere ma che vuole ricevere. In questo caso, dopo aver concorso per il canale, invia una richiesta al coordinator, il quale risponde (negli stessi slot) con un acknowledgment e poi con i dati veri e propri, senza eseguire il CSMA/CA.



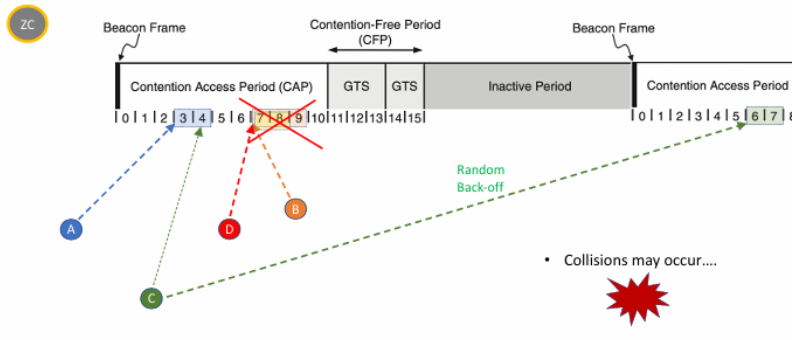
Se ci sono altri dispositivi che competono per il canale, come il dispositivo B che sceglie casualmente lo slot 7 ed inizia a trasmettere il suo pacchetto per un tempo totale di tre slot. In questo caso non vi è alcun problema perché i due dispositivi hanno scelto slot diversi e rilevano libero il canale.



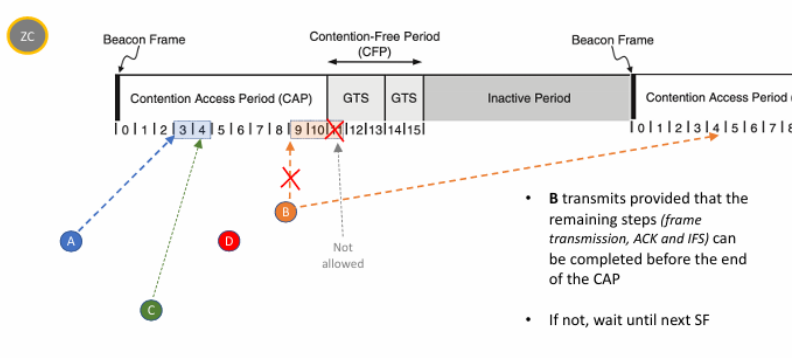
Se supponiamo di aggiungere un ulteriore dispositivo C che vuole trasmettere un pacchetto nello slot 4. In questo caso C esegue il CCA e si accorge che il canale è occupato ed esegue il back-off per un numero casuale di slot. Ritenta quindi la sua trasmissione nello slot 6 del superframe successivo.



Il CSMA Slotted non implica però la mancanza di collisioni. Infatti, i canali B e D ascoltano il canale nello stesso momento, lo rilevano libero ed iniziano a trasmettere entrambi nello slot 7.

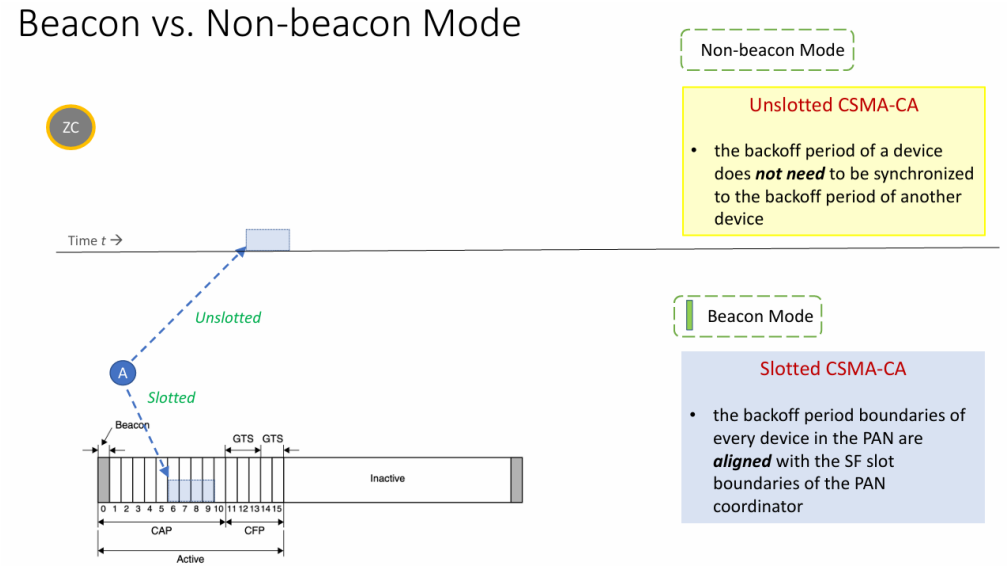


In quest'altro caso il dispositivo B ascolta il canale, lo rileva libero ma inizia a trasmettere un pacchetto di lunghezza tre slot a partire dal 9. Poiché non è in grado di completare la trasmissione entro la conclusione del CAL la re-schedula per il superframe successivo.

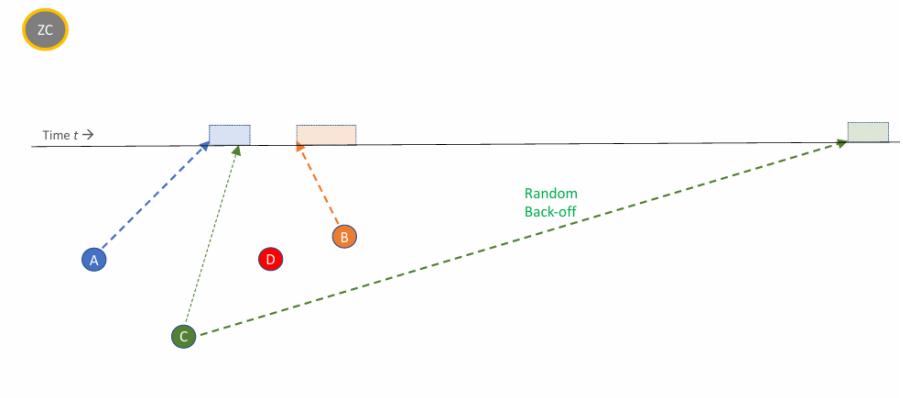


6.5.3 Unslotted (Non-Beacon Mode)

In questo caso i dispositivi non condividono lo stesso clock (i dispositivi non sono tutti sincronizati), pertanto il tempo non è suddiviso in slot.



A parte questa differenza, i procedimenti sono identici.



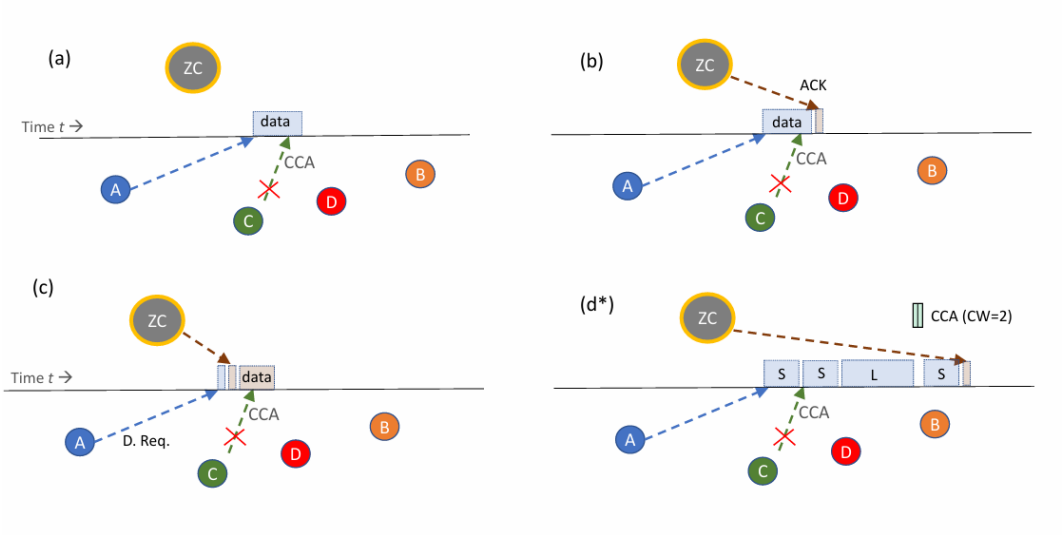
6.5.4 Timing

Nell'esempio (a) il dispositivo A trasmette dati al Coordinator mentre i dispositivi B e D sono in sleeo e non hanno nulla da trasmettere. Il dispositivo C vorrebbe trasmettere ma il CCA glielo impedisce.

Nell'esempio (b) il dispositivo A trasmette dati al Coordinator e questo risponde subito dopo con un ack. Il dispositivo C vuole trasmettere ma il CCA lo impedisce.

Nell'esempio (c) il dispositivo A trasmette un comando di richiesta al Coordinator che risponde con un ack e successivamente i dati veri e propri.

Nell'esempio (d) il dispositivo A trasmette più pacchetti per volta. Il Coordinator risponde con un ack solo alla fine dei quattro pacchetti (viene usato quindi un Cumulative Aggregate Acknowledgement, volendo però si potrebbe scegliere di rispondere dopo ogni singolo pacchetto).

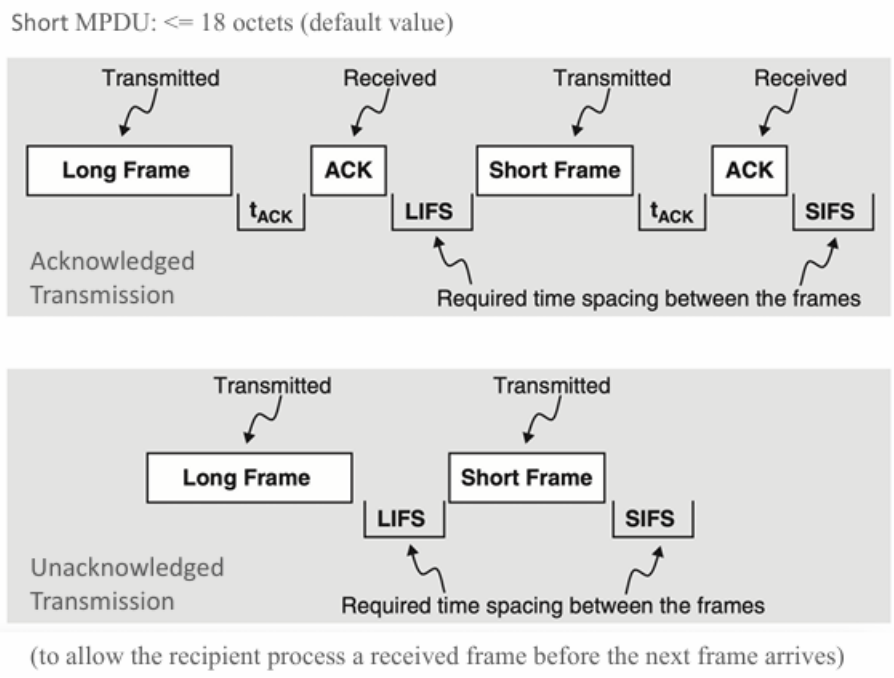


Un MAC Protocol Data Unit è definito Short Frame se il numero di byte è minore o uguale a 18 ottetti (valore di default), altrimenti è detto Long Frame. Un Coordinator potrebbe però anche di suddividere i pacchetti utilizzando una soglia diversa da quella di default.

Si definiscono quindi Short InterFrame Spacing (SIFS) e Long InterFrame Spacing (LIFS) due intervalli di tempo di attesa (per un breve periodo e per un lungo periodo).

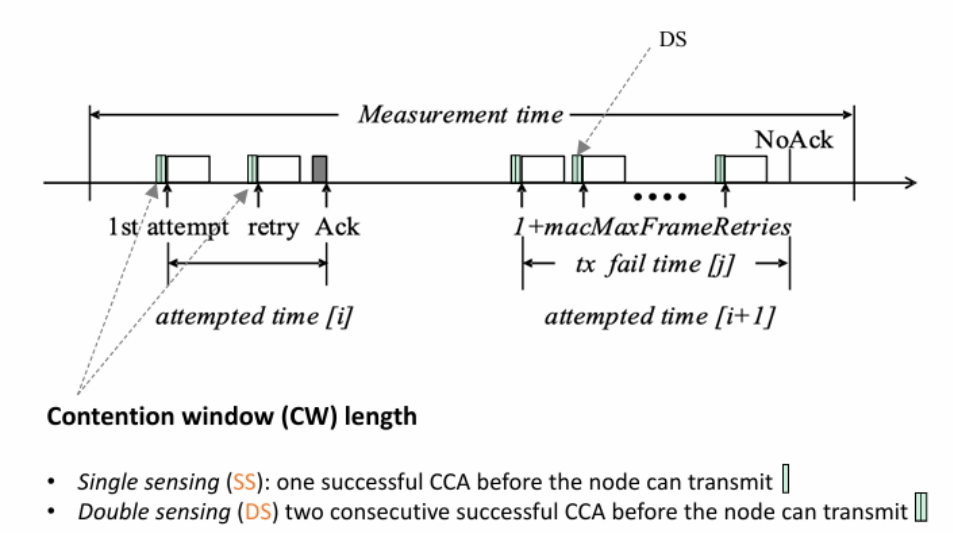
L'IFS (InterFrame Spacing) è il tempo che il coordinator necessita per processare uno short packet.

Per rendere più sicura la trasmissione possiamo imporre che un dispositivo deve aspettare un certo intervallo di tempo e se per l'intera durata dell'intervallo il canale è risultato libero può iniziare a trasmettere.



6.5.5 Contention Window (CW)

Questo parametro controlla per quanto tempo un dispositivo deve valutare se il canale è in idle prima di trasmettere. Il valore standard è 2 (si vuole che il canale sia in idle per due periodi di back-off prima di essere considerato libero).

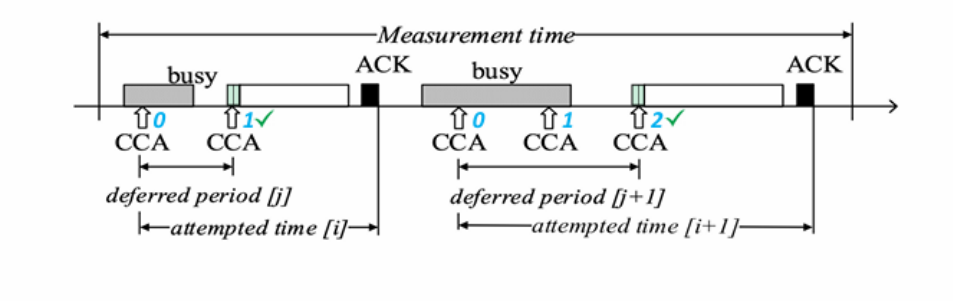


Un dispositivo può aspettare per due CCA positivi consecutivi prima di dichiarare il canale libero: si parla di meccanismo Double Sensing (DS). Il Coordinator ZigBee può anche decidere di aspettare un solo periodo di back-off prima di dichiarare il canale libero: si parla di meccanismo Single Sensing (SS).

Il primo è certamente più sicuro ma spreca più tempo, mentre il secondo è più sensibile alle collisioni (perché si ascolta il canale per un intervallo di tempo più breve) ma la comunicazione è più rapida per via dell'idle ridotto

6.5.6 Number of Back-off stages (NB)

É un altro parametro coinvolto nel CSMA.

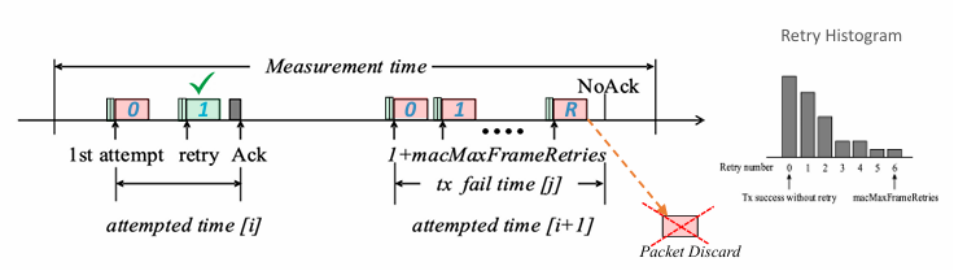


Nell'esempio notiamo a sinistra un dispositivo che vuole accedere al canale, il quale però è occupato. All'inizio il parametro NB è uguale a zero perché si tratta del primo tentativo di accesso. Poiché il canale è occupato si esegue il back-off e la trasmissione viene posticipata. Quando riparte NB sarà uguale ad uno (in quanto ha eseguito una volta il back-off) e trova il canale libero per trasmettere.

A destra invece notiamo che il dispositivo prova a trasmettere più volte prima di ottenere il canale, ad ogni tentativo il parametro NB viene incrementato di 1. Raggiunto il limite, il processo viene dichiarato come fallito a livello di rete.

6.5.7 Retry limit (R)

Parametro che assume valore zero non appena la trasmissione inizia. Ogni vola che una trasmissione di un pacchetto fallisce (evento segnalato dall'assenza di acknowledgement) viene incrementato di uno, fino al raggiungimento del valore massimo. Una volta raggiunto quest'ultimo si trasmette un'informazione a livello di rete e si decide cosa fare.



Nell'esempio a sinistra un dispositivo trasmette un pacchetto ma non riceve alcun riscontro capendo così che si è verificato un errore per il quale il pacchetto

si è corrotto e sceglie un nuovo intervallo di tempo per iniziare nuovamente a trasmettere. Riascolta, quindi, il canale capisce che è libero e trasmette lo stesso pacchetto che viene riscontrato. Riassumendo: abbiamo trasmesso due volte uno stesso pacchetto (la prima volta in modo errato e la seconda in modo corretto).

A destra lo stesso dispositivo prova a trasmettere sul canale per R volte ma non riceve nessun ack. Significa che il coordinator è spento. In questo caso il problema è a livello fisico e viene suggerito un cambiamento del canale.

6.5.8 Back-off Period (BP)

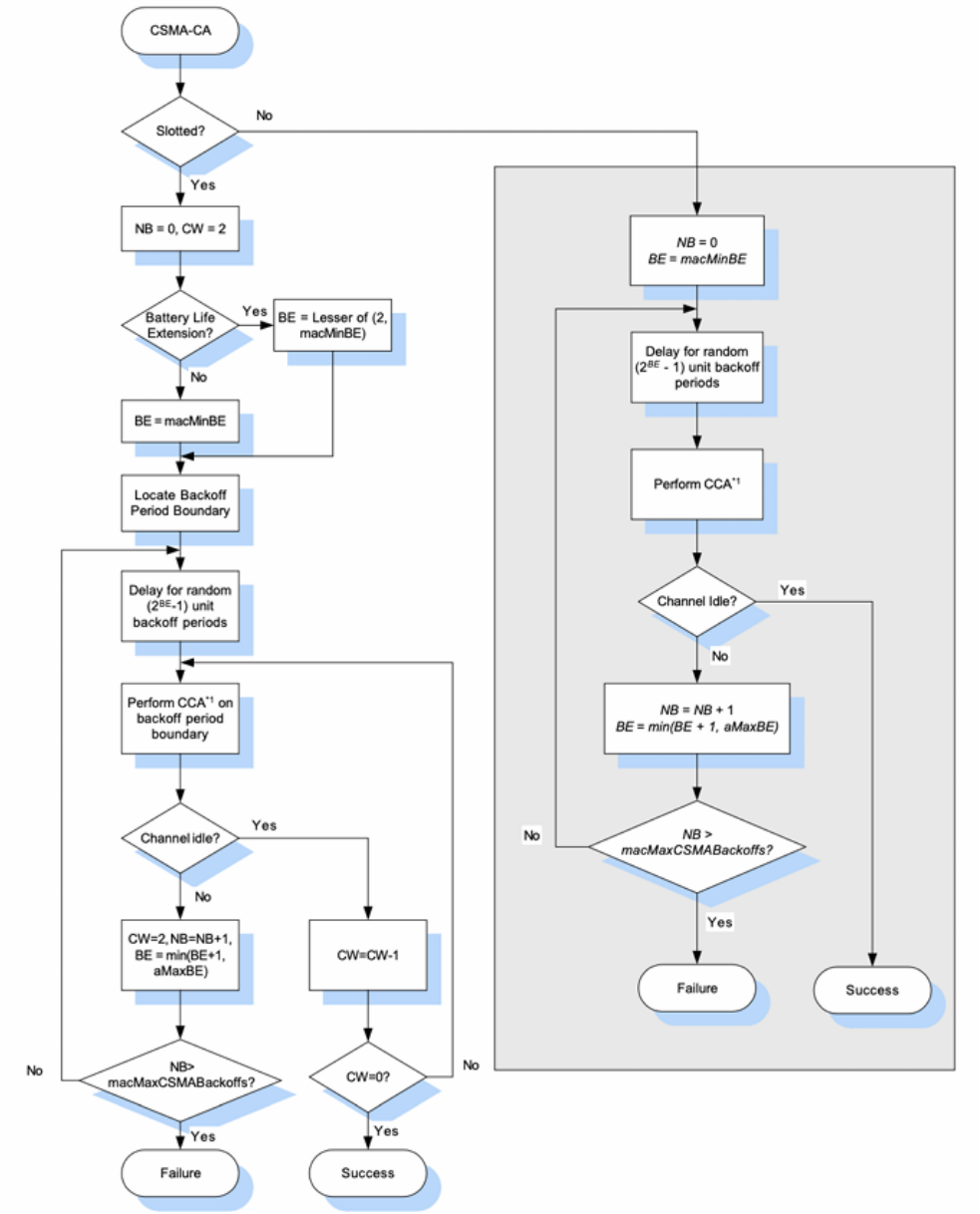


Figura 6.14: Differenza tra Slotted ed Unslotted CSMA in IEEE 802.15.4

É il più importante parametro del Carrier Sensing. In modalità unslotted il periodo di back-off deve sottostare ai vincoli imposti dal controller. Ciascun dispositivo ha un suo clock interno ed attiva o disattiva il BP in base ad esso, mentre nella slotted mode tutti i dispositivi fanno riferimento ad un clock comune.

Il periodo di back-off dura $320\mu s$. Il bitrate di ZigBee sul canale 2.4 GHz è di 250 kbps, quindi divideno per 4 si ottiene il symbol rate pari a 62.5 ksym/s. La velocità di trasmissione di un simbolo è quindi pari all'inverso del symbol rate, cioè 16 ms. Il periodo di back-off corrisponde a 20 simboli, ossia ciascun dispositivo ascolta 20 simboli sul canale prima di ritentare la trasmissione. Altri parametri importanti sono:

- $aTurnaroundTime = 192\mu s$, ossia il tempo necessario per passare da ricezione a trasmissione.
- $maxAckWaitDuration = 560\mu s$ ossia il tempo dalla fine della trasmissione del pacchetto per completare la ricezione dell'acknowledgement.

Si osserva che la durata complessiva di un pacchetto, con buona approssimazione, è pari a 54 simboli. Trattasi di un valore ben più alto rispetto al numero di simboli necessari per completare la ricezione dell'acknowledgement, ossia il rapporto tra *maxAckWaitDuration* e la durata di un simbolo.

il back-off è un numero casuale scelto tra $[0, 2^{BE}]$ con BE (Back-off Exponent).

6.5.9 Distribution Coordination Function (DCF)

É l'equivalente in wi-fi del protocollo CSMA/CA. La finestra di contesa viene chiamata Contention Window:

$$W_i = \begin{cases} 2^i W_0, & 0 \leq i \leq m \\ 2^m W_0, & m \leq i \leq L \end{cases}$$

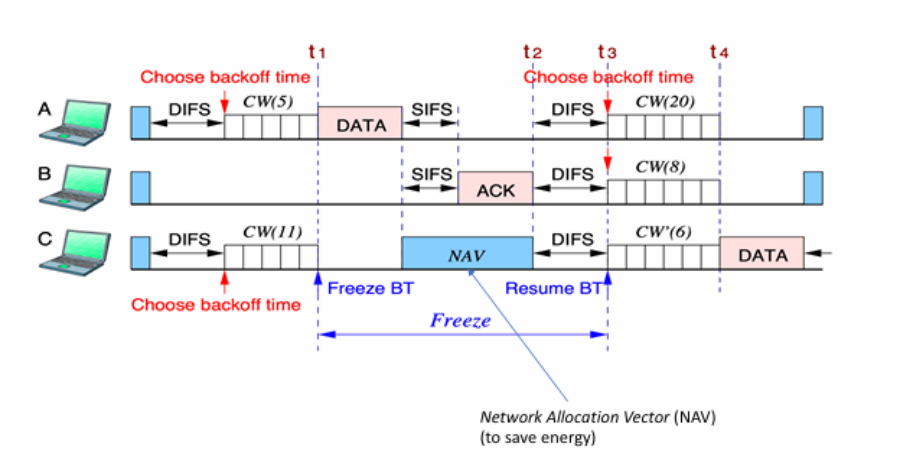
dove m indica il *back-off stages* ed L il numero massimo di tentativi che il dispositivo può eseguire (superati questi, il pacchetto viene scartato e si inoltra il problema ai livelli superiori per decidere come comportarsi).

La finestra di collisione raddoppia quando si verifica una collisione e viene resettata ad ogni successo.

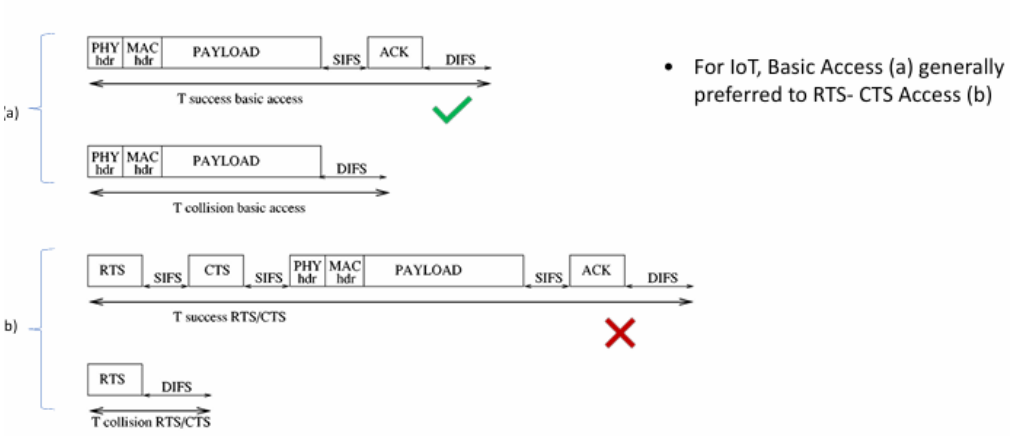
Nella figura seguente notiamo un dispositivo A che ha stabilito di attendere per 5 slot di contesa, un dispositivo B che non è interessato a trasmettere ed un dispositivo C che attende invece 11 slot. Il dispositivo A, quindi, è il primo a trasmettere. Il dispositivo C si accorge della trasmissione ma non va in sleep (il wi-fi non mira a risparmiare energia), piuttosto interrompe il contatore fermandosi a 6 per riprendere non appena il canale si è liberato.

Lo Short InterFrame Space (SIFS) è un tempo morto per dichiarare che il canale è libero. In ZigBee il tempo necessario affinché il canale potesse essere considerato libero era pari a due volte la CW, mentre in wi-fi è pari al

Distributed InterFrame Space (DIFS). Questo tempo è dimensionato sempre superiore al SIFS (il quale rapresenta il tempo massimo entro il quale un dispositivo deve rispondere ad un pacchetto ricevuto).



I consumi energetici sono più elevati ma anche le prestazioni. L'unico tentativo di risparmio energetico è rappresentato dal NAV (Network Allocation Vector) che è un temporizzatore attivato prima della trasmissione di un dato il quale indica il tempo minimo di occupazione del canale. Quindi è un informazione che il dispositivo può estrarre per capire se andare in sleep.



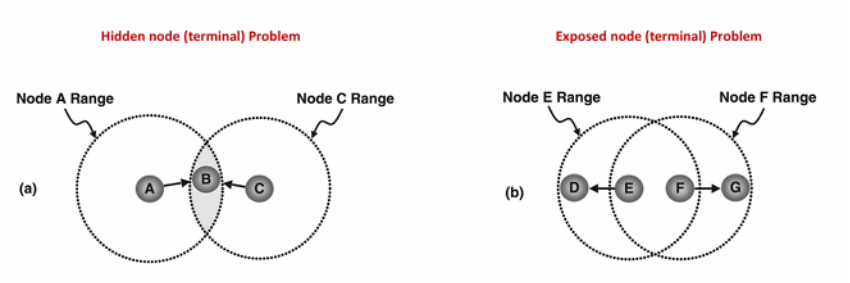
Nel caso (a) si ha un protocollo con handshaking a 2 vie, in cui viene trasmesso direttamente il pacchetto e (in caso di successo) viene ricevuto l'acknowledgment. Nel caso (b) si ha un protocollo con handshaking a 4 vie, in cui precedono due pacchetti di controllo, l'RTS (Request-To-Send) ed il CTS (Clear-To-Send), prima di iniziare il meccanismo visto nel caso precedente.

Se non vi fossero collisioni non avrebbe senso l'utilizzo del protocollo a 4 vie (perché questo impiega più tempo ed energia).

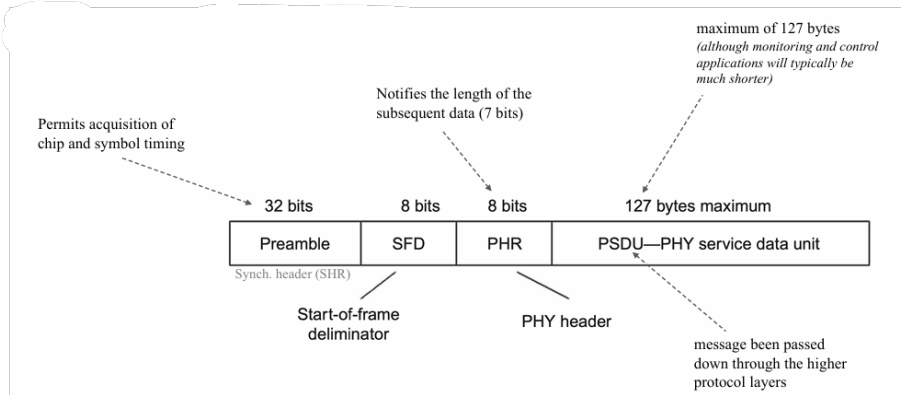
Weaknesses

Il meccanismo di handshake RTS/CTS (Request-To-Send/Clear-To-Send) non è supportato dall'IEEE 802.15.4.

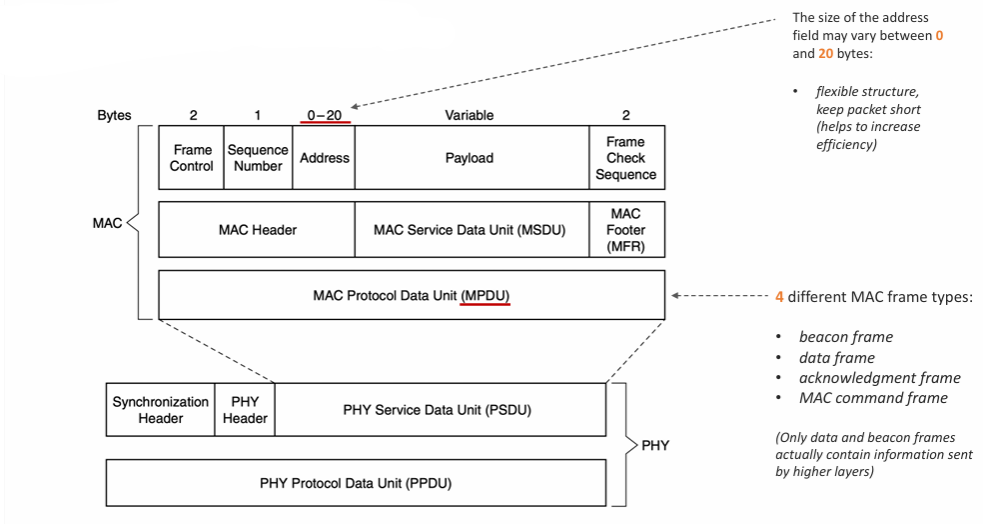
Questo lo espone a due problematiche: nodo nascosto e nodo esposto.



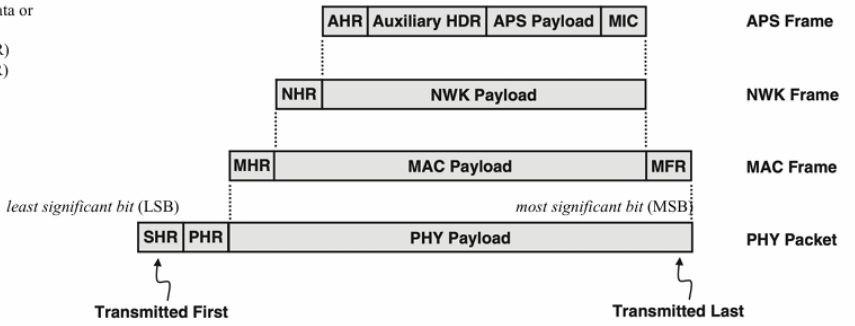
6.6 Packet Structure



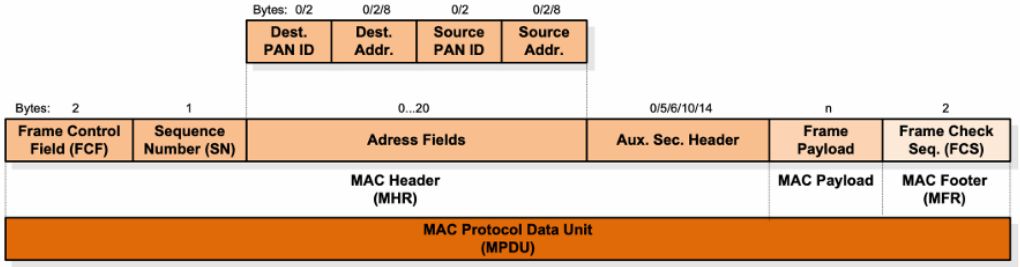
802.15.4 frame construction and channel access are *similar* to those of WLAN 802.11 (Wi-Fi) but are *less complex*



- Payloads contain data or commands
- MAC header (MHR)
- NWK header (NHR)
- APS header (AHR)

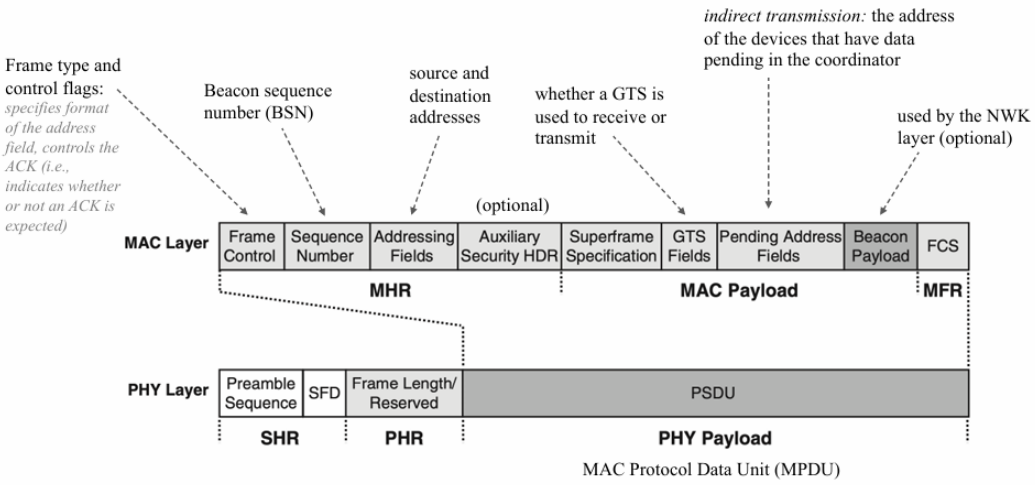


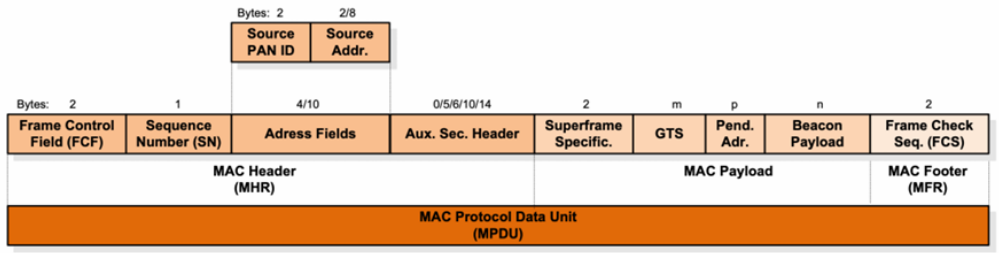
- MAC footer (MFR): 16-bit Frame Check Sequence (FCS) for data verification
- auxiliary HDR: contains the mechanism used to add security and the security keys (NWK and MAC frames can also have optional auxiliary headers for additional security)
- Message Integrity Code (MIC): security feature to detect any unauthorized change in the content of the message



IEEE 802.15.4 – General MAC Frame Format

6.6.1 Beacon Frame Structure

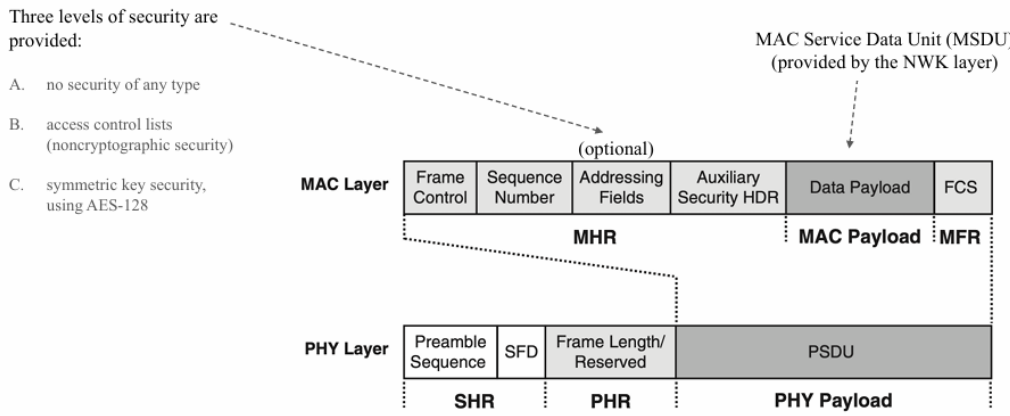


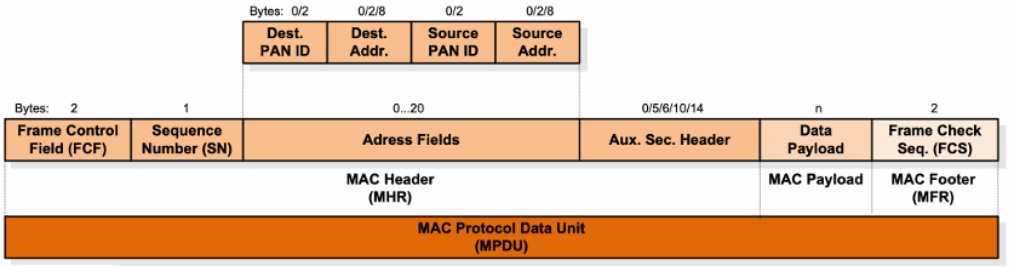


Note: The number of GTS, pending address and beacon payload bytes is variable

IEEE 802.15.4 – MAC Beacon Frame Format

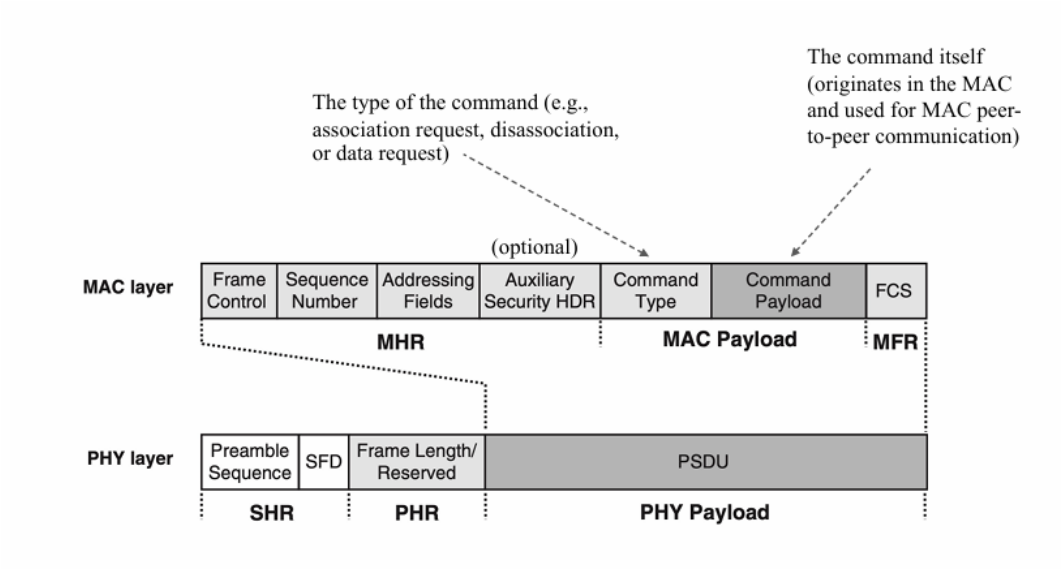
6.6.2 Data Frame Structure

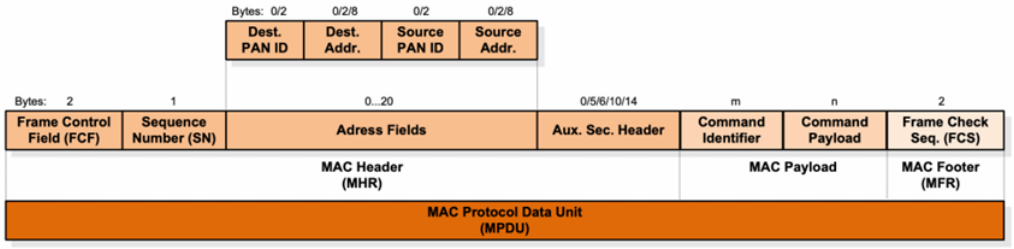




IEEE 802.15.4 – MAC Data Frame Format

6.6.3 Command Frame Structure

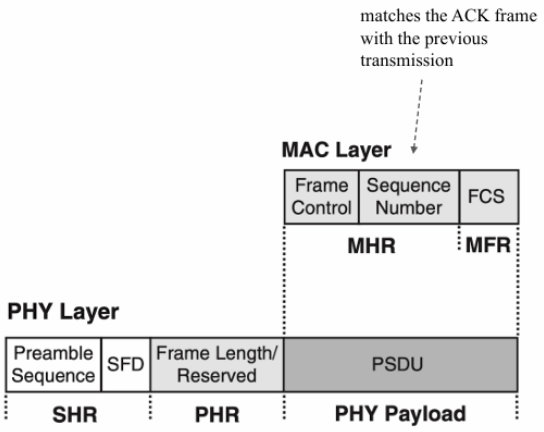


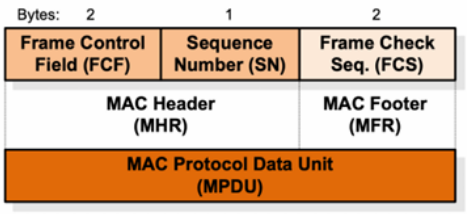


IEEE 802.15.4 – MAC Command Frame Format

6.6.4 ACK Frame Structure

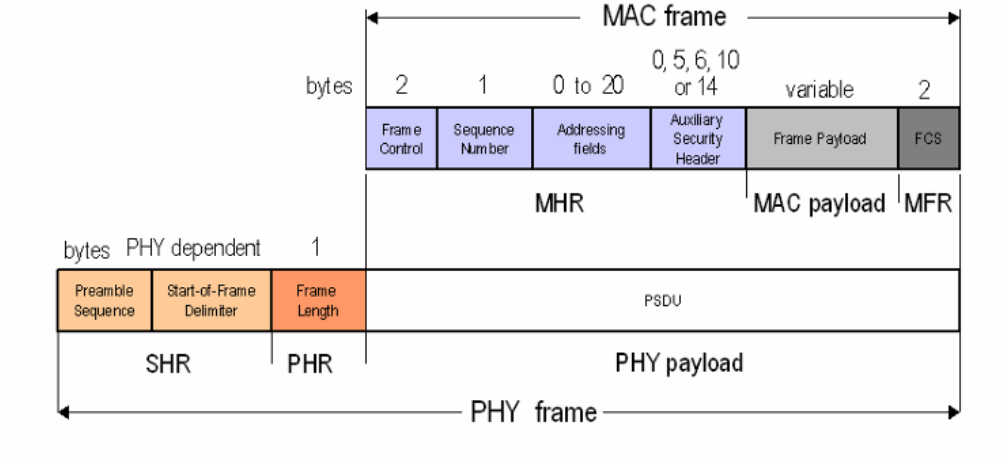
- Successful reception and validation of a a data or MAC command frame is confirmed with an ACK
  - ❖ If the receiving device is unable to handle the incoming message for any reason, the receipt is ***not acknowledged***
  - ✓ The acknowledgment frame is sent ***immediately after*** successful validation of the received frame



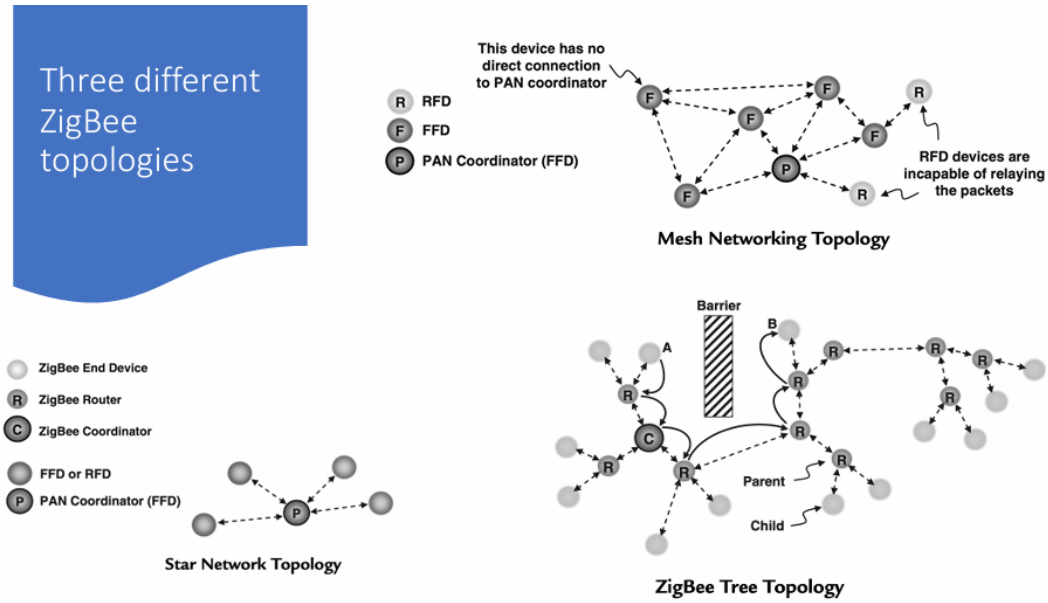


IEEE 802.15.4 – MAC Acknowledgment Frame Format

6.6.5 MAC Frame (bytes)



6.7 L3



Cosa succede quando si vuole parlare con un dispositivo che non rientra nella copertura del canale radio? C'è bisogno dell'utilizzo di algoritmi distribuiti per scoprire e mantenere percorsi tra più nodi.

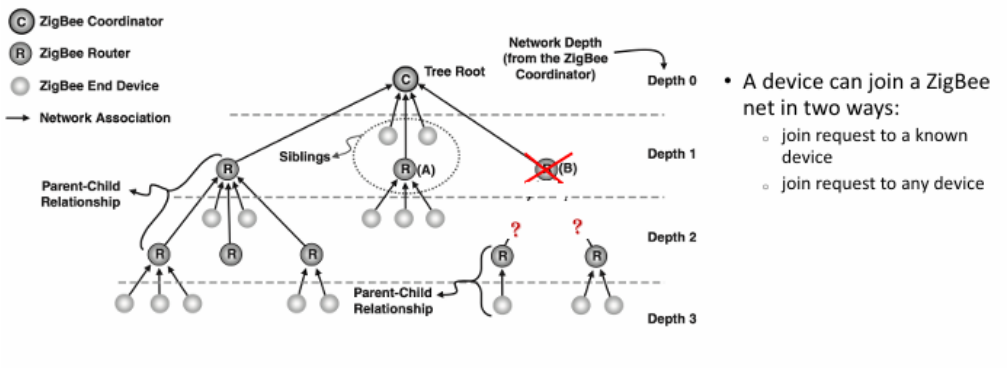
6.7.1 Topologia a Stella

Il Coordinatore ZigBee sceglie un PAN ID (valore univoco a 16 bit) per individuare l'intera rete e lo comunica a tutti i suoi dispositivi in broadcast, formando una struttura centralizzata.

Ciascun dispositivo è identificato da un indirizzo univoco a 64 bit (che in contesti limitati può essere ridotto ad una versione a 16 bit, per motivi di overhead).

6.7.2 Topologia Cluster-Tree

É una struttura priva di loop, composta esclusivamente da genitori e figli. Ciascun dispositivo può connettersi soltanto ad un genitore univoco. Il ZigBee Router si unisce ad un ZigBee Coordinator e permette ad altri Router o End Device di connettersi.



Ad una rete possono unirsi più di 60,000 dispositivi, ma la sua profondità massima (distanza massima Router-Router o Coordinator-Router) deve essere pari a 6; quindi, in ognuno di questi livelli ciascun nodo non può avere più di 20 figli (dei quali massimo 6 possono essere Router).

Le reti ad albero però presentano un problema. Essendoci un unico collegamento tra un nodo e la sua radice, se uno dei nodi intermedi cede, crolla l'intero sotto-albero (tutti i nodi ad esso connessi, di gerarchia inferiore).

6.7.3 Topologia Mesh

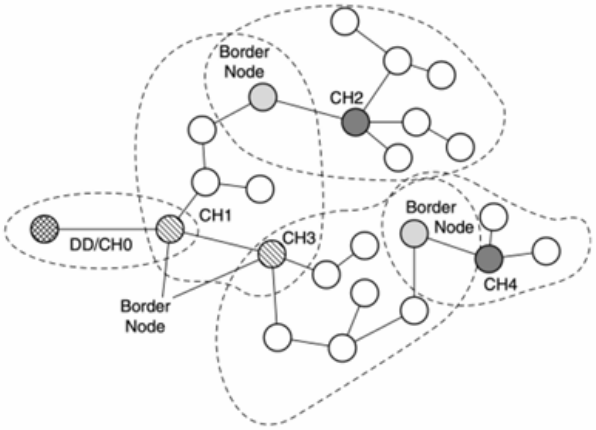
Le reti mesh sono densamente connesse (molto ramificate). I nodi intermedi possono fungere da router, ossia da instradatori dell'informazione. Rispetto alle precedenti topologie si ha una robustezza maggiore, dato che sono previsti più percorsi colleganti due nodi (se uno di questi cede è sempre possibile utilizzare l'altro, evitando il collasso della rete). Data la sua struttura complessa non è possibile usare il TCP/IP ma richiede dei protocolli di rete ad-hoc.

6.8 Routing Approach

I protocolli di routing si dividono in due categorie:

- *Proactive (Table-driven)*, tutti i nodi instradatori hanno il controllo di tutta la rete tramite le tabelle di routing. Sono protocolli a bassa latenza ma con grandissimo overhead, indicati per le reti statiche.
- *Reactive (Source-Initiated on-demand-driven)*, l'instradamento viene fatto su richiesta, rendendoli ideali per ambienti dinamici.

6.9 Cluster Tree Protocol



Clusters not bordering cluster 0 use intermediate clusters to get a CID

L'albero in ZigBee si forma partendo dalla *Cluster Selection*, in cui definiamo il DD (Designed Device) corrispondente al coordinatore principale della rete. [Si vuole dare la possibilità alla rete ad albero ZigBee di essere adattiva, sbilanciata e asimmetrica, in grado di sistemarsi da sola in caso di guasti].

Il Cluster-Head (CH) si elegge in base a delle specifiche (range di trasmissione, capacità di potenza...), una volta ricevuto il suo identificativo, il CH lo comunica in broadcast a tutti i suo sotto-nodi. I nodi che ricevono questo messaggio inviano una *Connection Request* al CH, a cui questo risponde con una *Connection Response* (contenente il node ID), una volta ricevuta. Per finire il nodo risponde con un ack.

Il DD assegna un CID (ID Cluster univoco) a ciascun CH e forma una rete multicluster alla quale si unisce, fungendo da CH del cluster 0, ed invia un messaggio HELLO ai nodi vicini. Se un CH ha ricevuto questo messaggio invia un messaggio di CONNECTION REQUEST con la quale si unisce al cluster 0, richiedendo un CID. Una volta ottenuto il nuovo CID, il CH informa i suo nodi membri attraverso il messaggio HELLO. Il CID e i node ID (assegnati dal CH a ciascun nodo interno al suo cluster) formano la coppia di indirizzi logici utilizzata per instradare i pacchetti.

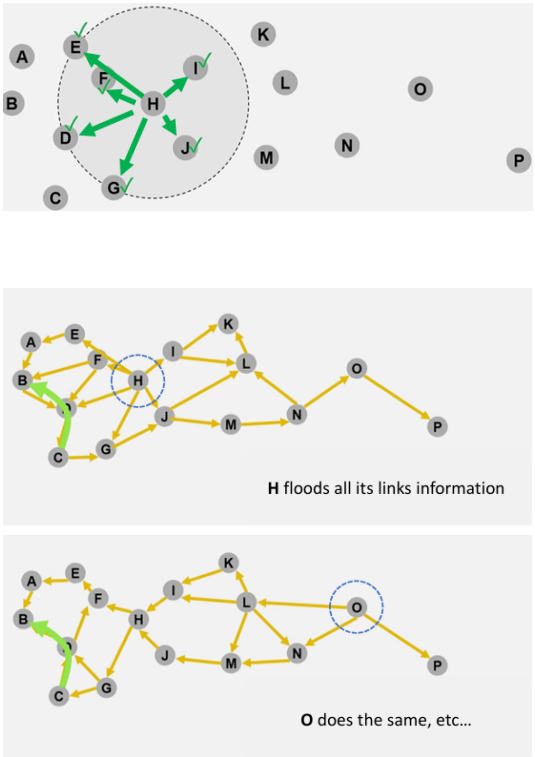
L'ultimo passaggio consiste nel nominare un nodo di frontiera (border node) che comunica con tutti i nodi del cluster genitore. Questo viene deciso dal relativo CH in base al broadcast del dispositivo principale DD (che è l'unico a conoscere la posizione di tutti i CH), per cui i border node vengono scelti e selezionati dopo la selezione dei CH.

In questa configurazione è evitato il problema del *flooding* (inondazione di pacchetti di controllo). Essendo una struttura ad albero il flusso è sempre unidirezionale (dalla radice verso le foglie, ciascun nodo intermedio non può fare altro che propagare quell'informazione). Non si creano inoltre fenomeni di loop (in cui i nodi trasmettono in broadcast lo stesso messaggio senza arrivare ad una soluzione).

6.9.1 Problematiche del routing

I protocolli di instradamento classico (quelli TCP/IP) presentano alcune problematiche da risolvere che li rendono inadatti al caso wireless.

Nelle reti cablate l'informazione viene inviata attraverso delle comunicazioni unicast. Se il nodo vuole comunicare un dato ai suoi nodi vicini è costretto a trasmettere il pacchetto uno ad uno. Nelle reti wireless, invece, basta inviare il pacchetto una sola volta (broadcast), sarà il canale radio a farlo arrivare contemporaneamente a tutti i suoi vicini.



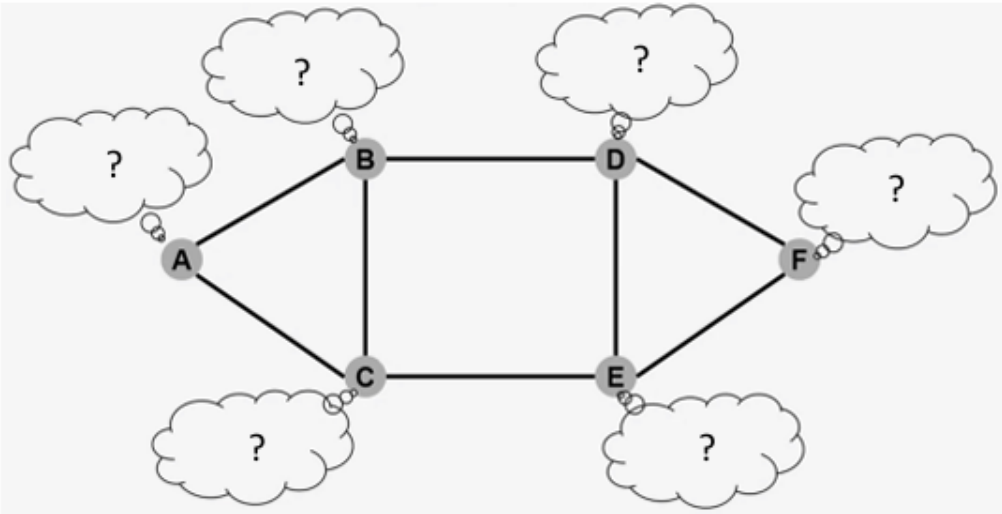
Se consideriamo l'esempio sopra, in cui il nodo C vuole trasmettere un dato al nodo B. Impiegando un protocollo tradizionale accade che ogni nodo genera pacchetti di controllo ridondanti ed inutili. Questo si traduce in problemi di banda, consumi energetici e di memorizzazione.

6.10 Mesh Networking Protocols

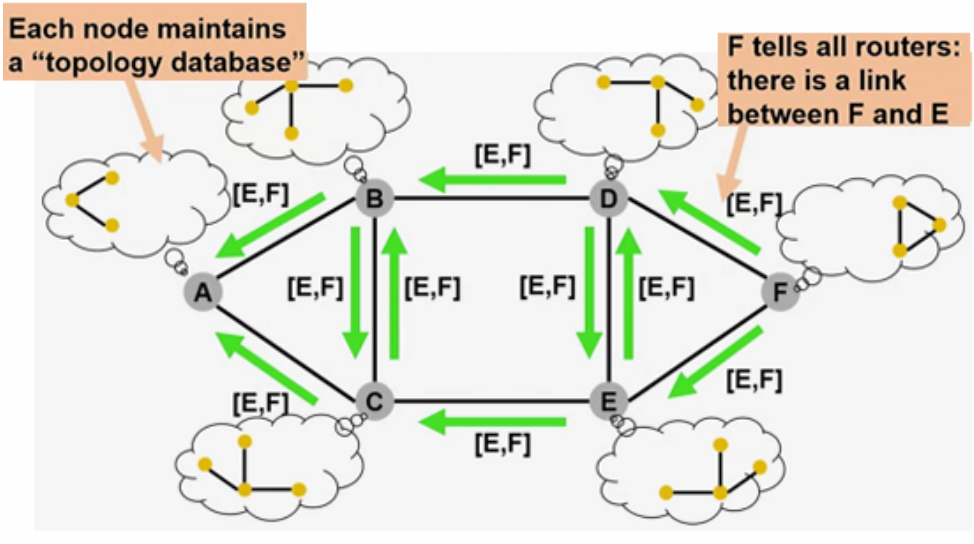
Per risolvere le problematiche di routing sono stati introdotti diversi protocolli.

6.10.1 Link State Routing (LSR)

Tutti i nodi, che non hanno la conoscenza della rete, la vogliono ottenere. Per ottenerla, scoprono attraverso l'utilizzo di protocolli di livello 2, quali sono i propri vicini e trasmettono in flooding su tutta la rete.



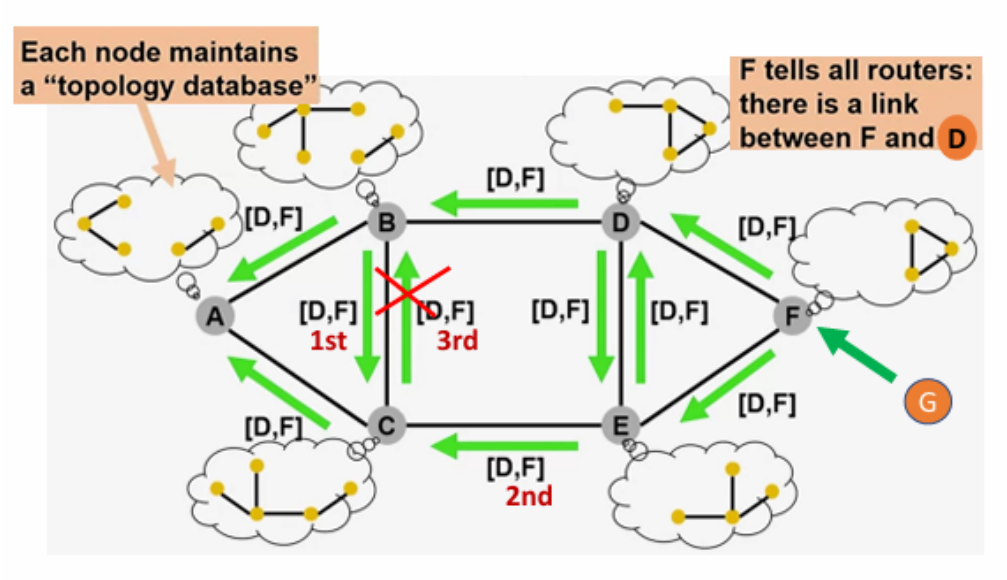
Ad esempio, il nodo F scopre che ha il nodo D come vicino e tutti gli altri nodi verranno a conoscenza di questa informazione, aggiungendola al proprio diagramma.



Se un nuovo nodo G si connette alla rete, scopre i suoi vicini (F in questo caso) e chiede a questi come è fatta l'intera topologia. A seguito di questa operazione

vengono trasmessi altri messaggi di flooding con i quali si comunica al resto della rete del nuovo link tra i nodi G ed F.

Se però il nodo G, dopo aver trasmesso per pochi minuti, si disconnette (per un qualunque motivo). Verrebbero propagati tanti messaggi inutili che arriverebbero ai nodi quando G già non c'è più.



Esistono vari meccanismi per evitare l'inondazione di pacchetti inutili.

Il Sequence Number, utilizzato negli header dei pacchetti permette di evitare loop di aggiornamenti (se un nodo ha già ricevuto il Link State agiornato non lo inoltra nuovamente ai suoi vicini). Per trasmettere nuove informazioni il sequence number viene inrementato di uno.

Dopo che i nodi hanno scoperto l'intera rete inizia la fase di *forwarding* (trasmissione dei dati).

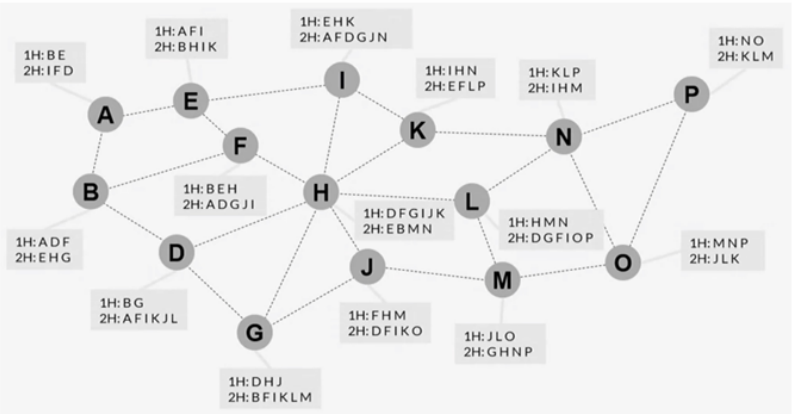
Quando un nodo vuole trasferire un pacchetto ad un altro nodo, sceglie il percorso a distanza minima utilizzando l'algoritmo Dijkstra (questo è utilizzabile però solo se si ha una conoscenza completa dela rete, non è adatto quindi all'IoT). Il protocollo può assegnare un costo ai collegamenti per controllare la selezione del percorso. [Il percorso può essere scelto in base a diverse metriche, ad esempio a ciascun link può essere associato un costo unitario (se ogliamo minimizzare il numero di hop intermedi)].

6.10.2 Optimized Link State Routing (OLSR)

Un modo ottimizzato per trasferire il minor numero di informazioni possibili per raggiungere lo stesso obiettivo finale. [Non tutti i nodi inoltrano pacchetti nella costruzione della topologia, ma si scelgono dei nodi specifici detti MultiPoint Relays (MPR)].

Per esempio, se il nodo H esegue il Neighbor Discovery (procedimento di livello 2 per il quale manda in broadcast un messaggio HELLO, segnalando la sua presenza). Tutti i nodi vicini vengono a conoscenza ed aggiungono il

nodo H come nodo distante 1 hop da essi. Terminato questo processo, il nodo H invia nuovamente un messaggio di HELLo con il quale però comunica le informazioni riguardanti i suoi vicini (in questo modo K, F, J capiscono di essere distanti 2 hop tra loro).



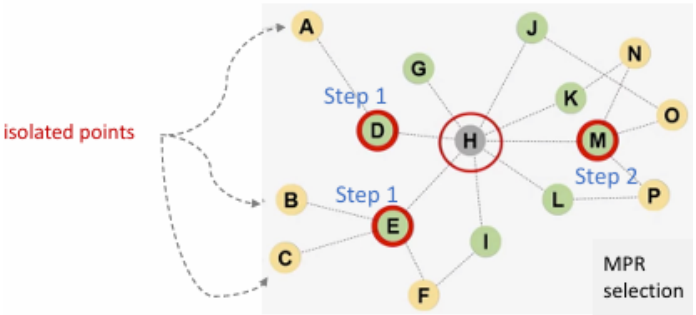
Per propagare la tipologia completa, l'OLSR non fa inoltrare a tutti i nodi la loro informazione, ma fa in modo che ciascuno di essi scelga un subset di vicini ai quali inviare lo stato dei collegamenti, attraverso dei pacchetti chiamati Link State. Il nodo H (esempio in basso) elegge in modo ottimizzato i propri MPR, attraverso un algoritmo.

1. Fra i propri nodi di livello 1, il nodo H sceglie i nodi che servono punti isolati. [Ad esempio il nodo A è raggiungibile soltanto dal nodo D, quindi quest'ultima va scelto necessariamente, essendo l'unico attraverso il quale il nodo H può raggiungere A. Stesso dicasi per il nodo E (che permette di raggiungere i nodi isolati B e C)].

2. Seleziona i nodi in grado di massimizzare la copertura. [Sceglierà quindi M, perché permette di raggiungere N, O, P].

3. Ripete lo step 2 finché tutti i nodi non sono coperti [nell'esempio di figura nostro non c'è bisogno].

Non è detto che questo algoritmo raggiunga l'ottimo matematico, ma il procedimento è molto veloce e otteniamo comunque buone prestazioni.

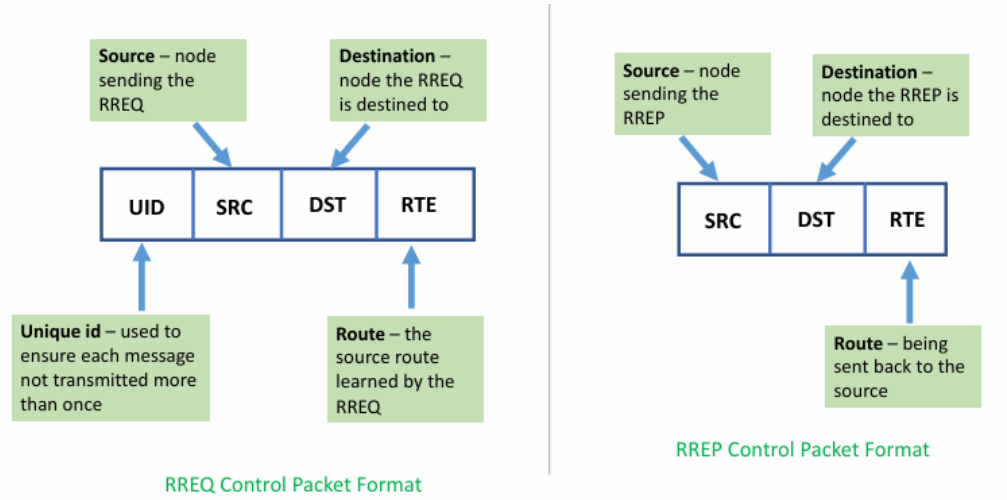


Costruita la topologia della rete, il forwarding è lo stesso del tradizionale Link State. Ma si sceglie di utilizzare il Dijkstra solo su gli MPR per instradare il traffico dati.

ZigBee non fa però uso del protocollo OLSR perché in ogni istante tutti i collegamenti possibili tra i nodi sono mantenuti, quindi si potrebbero avere problemi in termini di privacy.

6.10.3 Dynamic Source Routing (DSR)

É un protocollo reattivo utilizzato da ZigBee. Il suo funzionamento prevede l'utilizzo di due pacchetti principali: il RREQ (route request), ossia la richiesta di instradamento; ed il RREP (route reply), ossia la risposta della destinazione alla richiesta di instradamento della sorgente.



Il campo RTE (route) contiene l'intero percorso appreso dal pacchetto (è importante al fine di evitare le inondazioni).

Il campo UID (Unique ID) è usato per assicurare che ciascun messaggio non venga trasmesso più volte (simile al Sequence Number).

Si supponga che il nodo A voglia inviare un pacchetto al nodo M.

Naturalmente, prima di poter inviare il suo pacchetto dati, A deve costruire la conoscenza della rete necessaria per raggiungere M (non dell'intera rete). Invia, quindi, ai suoi vicini un pacchetto con il quale si identifica e specifica la destinazione. I nodi B ed E (vicini) ricevono la RREQ e si aggiungono come secondi nodi nel campo RTE.

Il nodo B, a sua volta, inoltra il pacchetto, il quale raggiunge F che memorizza l'informazione (A vuole trasmettere verso M passando da B) ed aggiunge se stesso come nodo intermedio nel campo RTE. F però può essere raggiunto sia da B che da E, però se viene raggiunto prima dal nodo B, non farà nulla una volta che sarà raggiunto anche da E, perché avrà già ricevuto un pacchetto con lo stesso UID.

La situazione si evolve fino a quando il nodo H esegue il broadcast (tutti i nuovi nodi inoltrano il pacchetto, quelli che lo avevano già ricevuto non fanno niente).

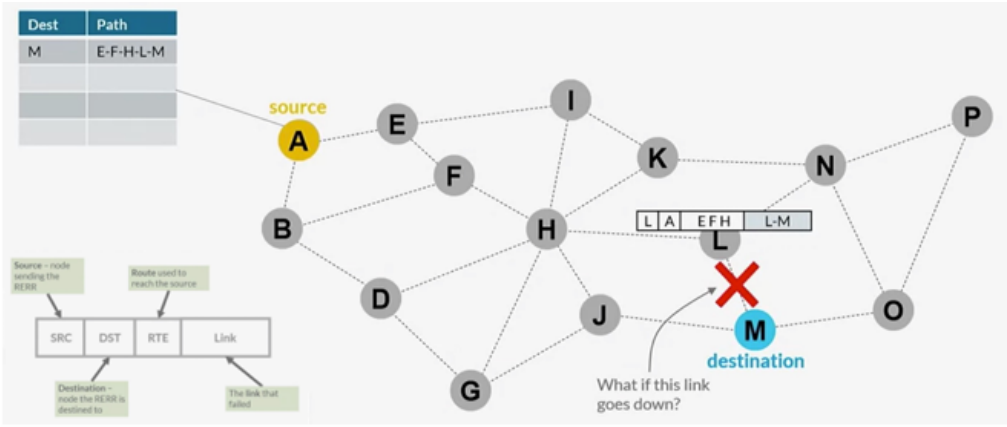
Il meccanismo procede fino a quando L raggiunge M che è il destinatario.

Una volta ricevuto il RREQ, il nodo M risponde ad A con il pacchetto RREP in maneria più semplice perché nel campo RTE è riportato l'intero percorso fra i due nodi.

Una volta che il nodo A riceve il RREP ha individuato M e conosce il percorso per raggiungerlo, allora può iniziare la trasmissione dei dati secondo la sequenza di nodi determinata.

Si definisce un certo intervallo di tempo per il quale un nodo mantiene in memoria il percorso per raggiungere un altro nodo. Trascorso tale tempo deve liberare la memoria per via delle sue capacità limitate. Se un nodo riceve, entro il timeout impostato per un percorso, una richiesta relativa a quello specifico percorso, allora aggiorna il timeout (azzerandolo). Se, invece, il timeout impostato per un percorso scade prima che possa essere ricevuta una richiesta relativa a quel percorso, l'informazione viene persa e deve essere ricostruita.

[Il route caching crea delle problematiche come, ad esempio, le tempeste di risposta. Se tutti memorizzano il percorso, andrebbero a inondare la rete avvisando di sapere il path per arrivare a M. Per risolvere il problema si utilizza una specie di CSMA, scegliendo un tempo di delay casuale].



Se c'è una failure del link tra L ed M, entra il gioco il pacchetto RERR (Route Error). Questo pacchetto viene trasmesso al nodo A, il quale una volta ricevuto cancella il percorso registrato ed avvia una nuova ricerca.

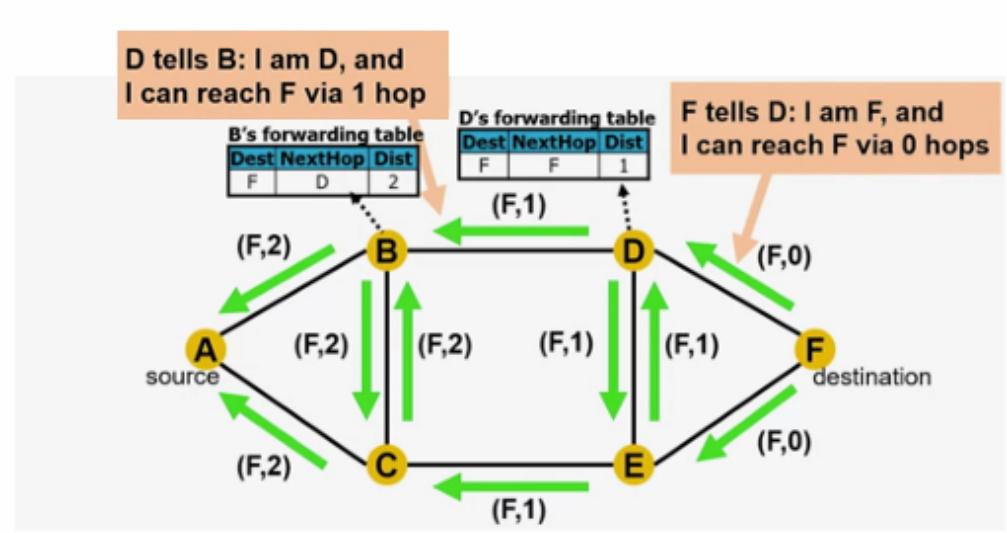
In una rete ZigBee molto lunga questo metodo non è adatto.

6.10.4 Distance Vector Routing

Non viene conservato l'intero cammino ma un puntatore al nodo precedente dal quale riceve il pacchetto.

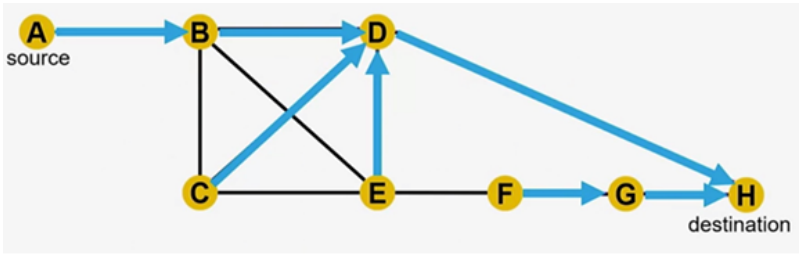
Per esempio se il nodo F vuole raggiungere il nodo A, invia un pacchetto di broadcast con il quale indica di essere a distanza 0 da se stesso. Il nodo D riceve

il pacchetto, comprende di essere a distanza 1 dal nodo F e comunica questa informazione in broadcast. Il nodo B, una volta ricevuta questa informazione, indica nella sua tabella di forwarding di essere a distanza 2 dal nodo F e che vi può arrivare tramite il nodo D. Questa informazione raggiunge il nodo A che riconoscerà di essere a distanza 3 da F e saprà come arrivarci.

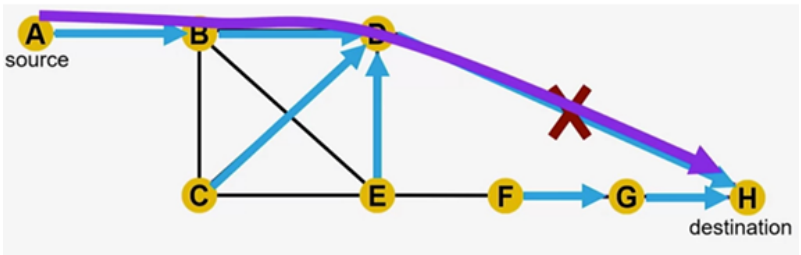


Advertisements flooding

Consideriamo il cammino in figura che parte dal nodo A ed arriva al nodo H passando dai nodi intermedi B e D.

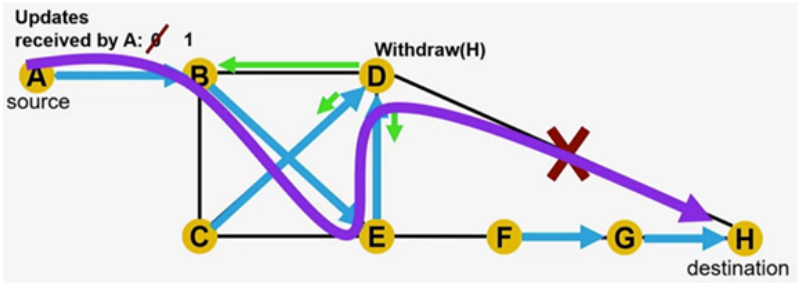


Se il collegamento D-H cade, il nodo D se ne accorge (perché non è più a distanza uno da esso) e deve comunicare questa informazione con un pacchetto, nel quale afferma di essere a distanza infinita dal nodo H. Questo pacchetto raggiungera i nodi vicini B, C ed E (in tempi diversi).

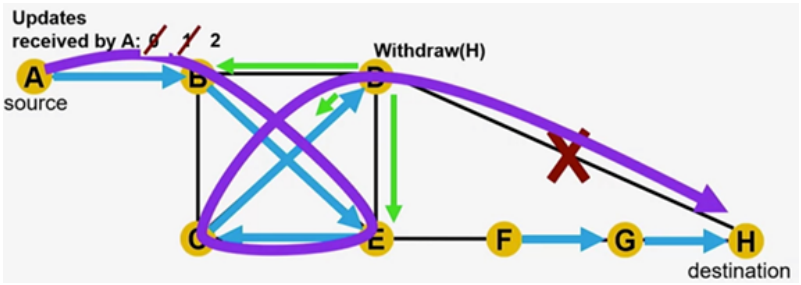


Link Failure issue

Se il pacchetto raggiunge prima il nodo B, questo ricodifica tutto (sa di non poter contare sul nodo D per raggiungere H). B si affida ad E (che risulta a distanza 2 dal nodo H nella sua tabella), ma non sa che questo era permesso attraverso il nodo D. I pacchetti quindi percorrendo la strada riportata in viola saranno persi.

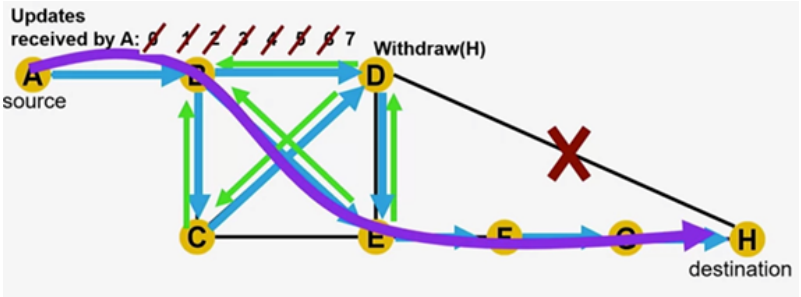


L'informazione della distanza infinita del nodo H dal nodo D raggiunge anche il nodo E, Questo invia il pacchetto al nodo C, che non aveva ricevuto l'informazione dal nodo D (quindi inoltra i pacchetti su un percorso che prevede ancora il nodo D e che porta a fallimenti).



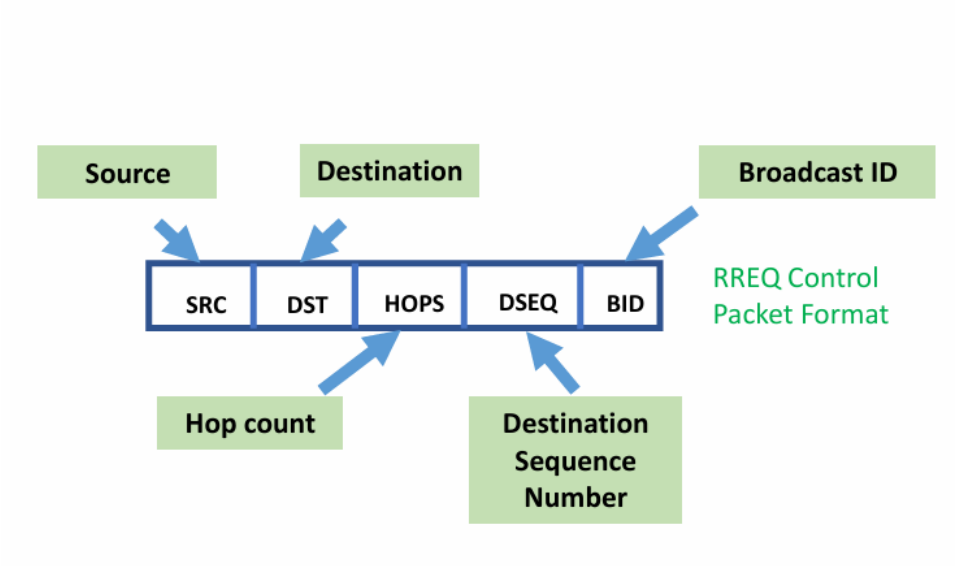
Prima di arrivare a convergenza su un cammino ancora attivo trascorre tanto tempo, vengono inviati tanti pacchetti e conseguentemente viene sprecata tantissima energia. Per risolvere questo problema si usano i sequence number.

[Se il diagramma fosse stato ad albero e non a maglia non avremmo avuto questo problema].



6.10.5 Ad-hoc On-demand Distance Vector (AODV)

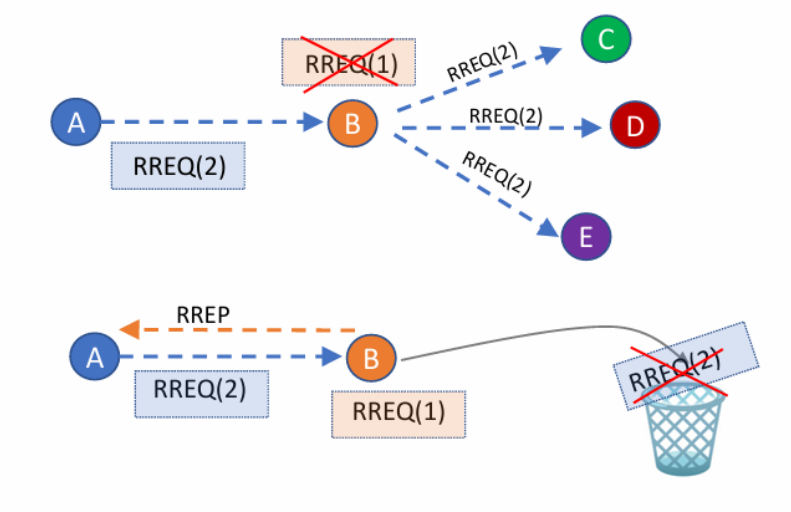
É la versione reattiva del Distance Vectore. Come nel DSR, l'instradamento avviene esclusivamente quando necessario e vengono introdotti i pacchetti di RREQ d RREP.



Un RREQ è identificato univocamente attraverso la coppia (SRC, BID).

Il protocollo AODV è suddiviso in cinque step:

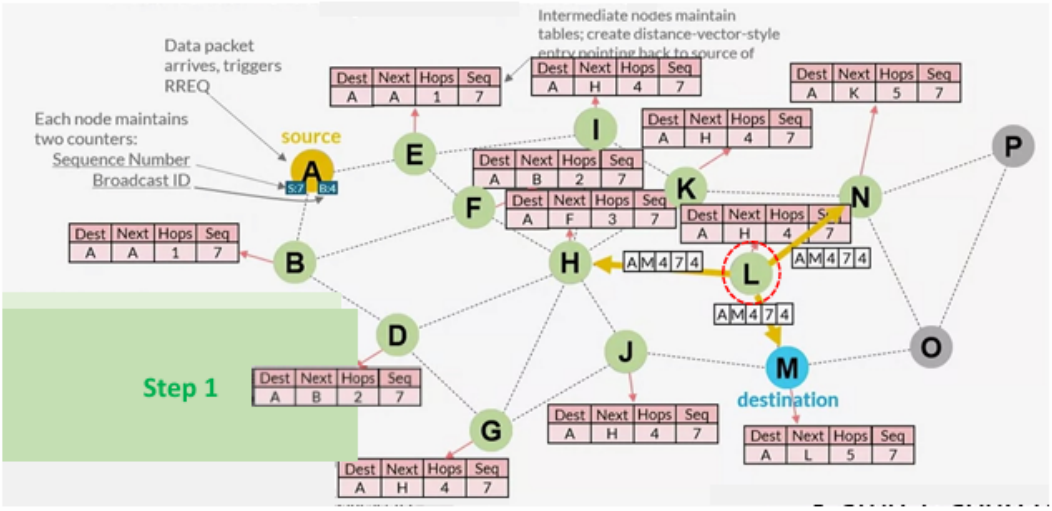
1. la sorgente trasmette un pacchetto RREQ
2. i nodi intermedi ricevono la richiesta. Controllano la coppia (source, identificativo) per capire se è un richiesta già ricevuta ed in caso affermativo la ignorano; altrimenti incrementano l'hop count inoltrando la richiesta in avanti e mantenendo alcune informazioni (DST, SCR, BID, DSEQ ed expiration time).
3. la richiesta raggiunge un nodo intermedio che possiede il cammino verso la destinazione. Questo compara il DSEQ che conserva con il campo DSEQ del RREQ ricevuto. Se quest'ultimo è maggiore del primo, il RREQ introduce un informazione nuova rispetto a quella conservata ed il nodo intermedio non fornisce il cammino alla destinazione, ma inoltra la richiesta. Se, invece, accade il contrario allora risonde con un RREP illustrante la sequenza di nodi da percorrere per raggiungere la destinazione desiderata.
4. Il pacchetto RREP raggiunge la sorgente. Lungo la propagazione all'indietro viene eseguito il catching e viene fatto uso di un timeout (allo scadere del quale il percorso viene scartato).
5. Si effettua ul mantenimento del percorso (ciascun nodo mantiene una lista dei propri nodi attivi). Quando un nodo intermedio manda un pacchetto e non ritorna nulla, o riceve un messaggio di allerta (comprende che qualcosa è andato storto ed invia un unsolicited RREP indietro alla sorgente). Facendo questo pone ad infinito il numero di hop necessari a raggiungere il nodo per il quale è caduto il link ed una volta stabilito il nuovo percorso si incrementa di uno la *freschezza* dell'informazione (il DSEQ), in maniera tale che l'aggiornamento dell'informazione abbia massima priorità.



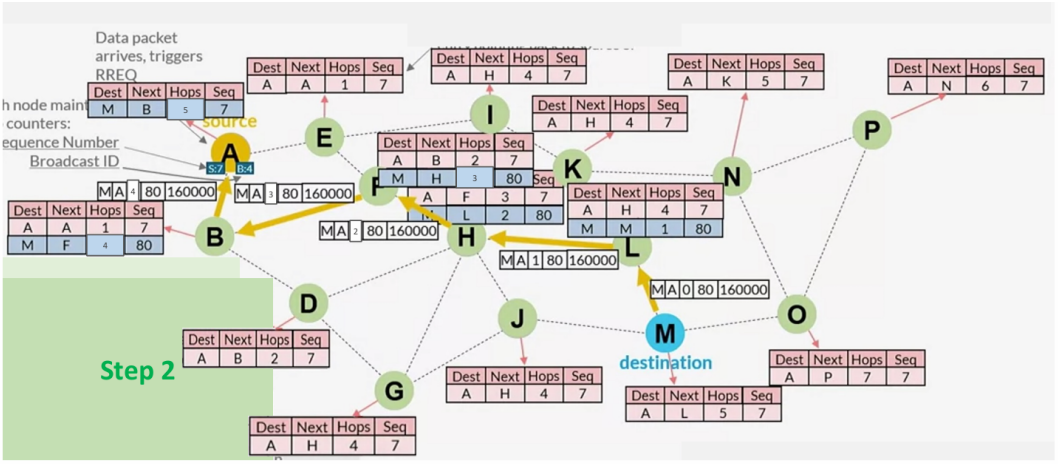
Supponiamo che il nodo A voglia raggiungere il nodo M. Per prima cosa esegue il flooding dell'informazione per raggiungere la destinazione.

Ciascun nodo intermedio invia le informazioni e costruisce la tabella con destinazione, nodo precedente, numero di hop complessivamente necessari per raggiungerla e Sequence Number.

Una volta raggiunta la destinazione il nodo M trasmette il pacchetto RREP avente come ultimo campo un timeout, allo scadere del quale le tabelle di instradamento vengono azzerate.



Il nodo L riceve il RREP e comprende che anche M può essere una sua destinazione, dunque lo aggiunge alla sua tabella di routing. Comunica queste informazioni ad H che a sua volta aggiorna la tabella di routing. In definitiva, il pacchetto raggiunge il nodo A, informandolo che esiste un percorso di 4 hop per trasmettere dati al nodo M avente come next hop il nodo B.



6.11 Sicurezza

Le principali problematiche sono:

- *Riservatezza* - un dispositivo malevolo può ottenere informazioni sensibili ascoltando i messaggi trasmessi. [Soluzione: encryption].
- *Autenticazione* - un dispositivo malevolo può modificare e rispedire alcuni dei messaggi anche se sono criptati. [Soluzione: MIC (Message Integrity Code)].

7 6LoWPAN

7.1 IPv6 over Low-poWer Personal Area Network

Questo protocollo nasce per risolvere alcune problematiche tra IPV6 ed IEEE 802.15.4, fungendo da strato cuscinetto per adatta l'IPv6 alle reti IoT low-power e *lossy*. L'organizzazione che gestisce questo protocollo è la IETF (Internet Engineering Task Force).

Le principali funzionalità del 6LoWPAN sono: compressione degli header dei livelli superiori, frammentazione dei pacchetti IPv6 in pacchetti più piccoli dalla dimensione massima di 127 byte e re-assemblaggio degli stessi.

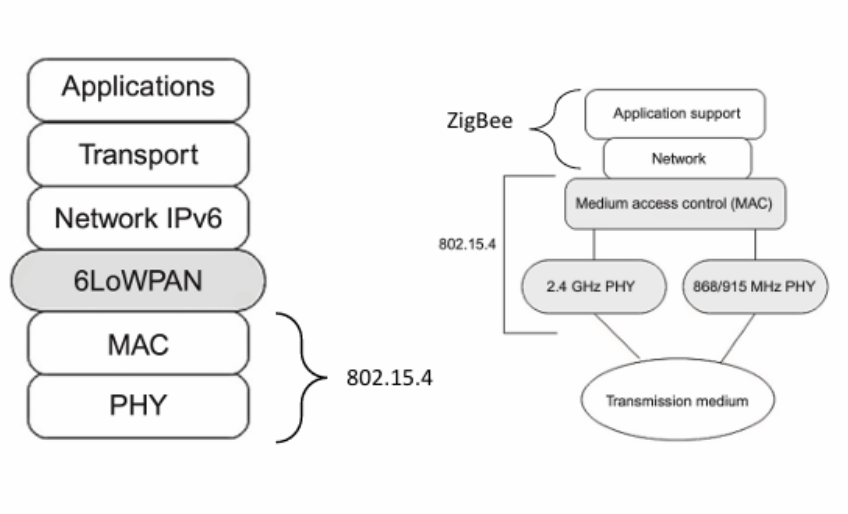


Figura 7.1: Differenze tra lo stack ibrido con 6LoWPAN e quello tradizionale ZigBee

7.2 Embedded Internet Stack

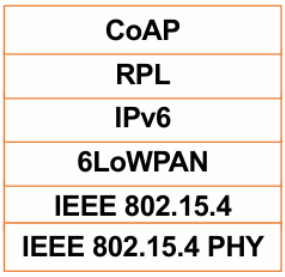


Figura 7.2: Pila protocollare ibrida

La MTU (Maximum Transmission Unit) è grande 1280 byte per l'IPv6 e solo 127 byte per l'IEEE 802.15.4, quindi abbiamo problemi con la compatibilità a causa della dimensione dei pacchetti.

L'header di un pacchetto IPv6 può essere ristretto comprimendo alcune informazioni e ricavarne altre direttamente dall'indirizzo MAC.

Nella migliore delle ipotesi il LoWPAN_IPHC può comprimere l'header IPv6 fino a 2 ottetti se 6LoWPAN deve instradare un pacchetto internamente alla sotto rete; se invece deve inoltrarlo verso l'esterno (il destinatario è diverso) il minimo è 7 ottetti.

7.2.1 Mobility

Nell'IoT è possibile sia spostare i nodi che le reti. Nell'ambito della **network mobility** parliamo di *roaming* se cambia la rete e di *handover* se cambia il punto di accesso. Nel caso d **node mobility** abbiamo *micro-mobilità* se ci spostiamo all'interno di una stessa rete e *macro-mobilità* se ci spostiamo tra domini di rete (cambiano gli indirizzi IP).

Gli spostamenti sono dovuti a diverse motivazioni: collocazione fisica, prestazione della rete ed altro.

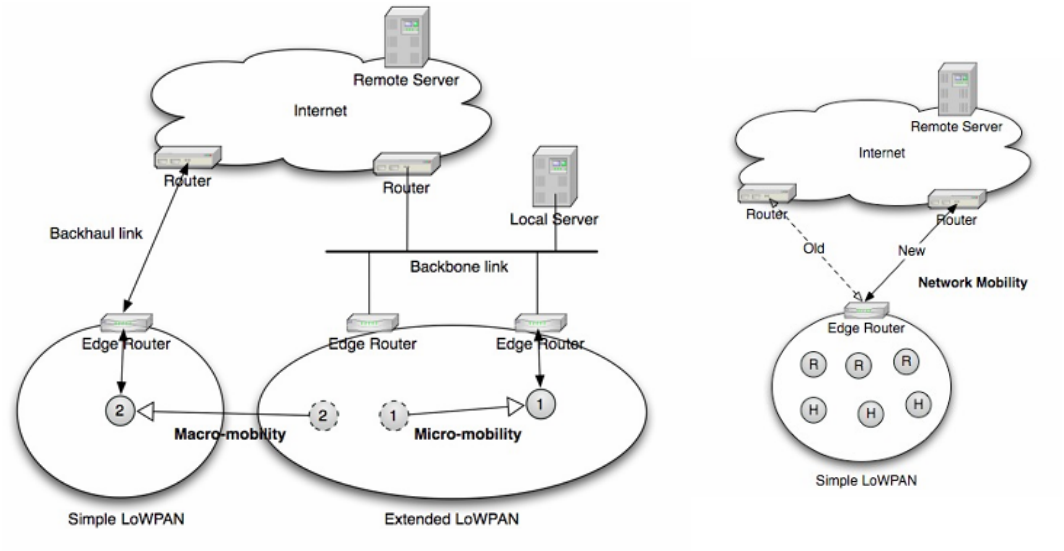
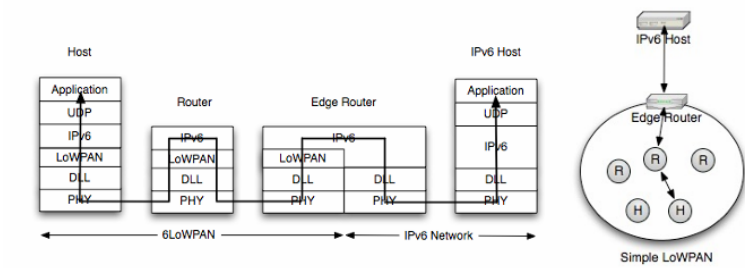


Figura 7.3: Node Mobility (sinistra) e Network Mobility (destra)

La gestione della mobilità solitamente viene eseguita ("non facendo nulla") riavviando tutti i nodi.

7.2.2 Routing

L'Edge Router (o router intermedio) elabora i pacchetti e converte lo stack sia in un senso che nell'altro,



7.3 Application Layer

I protocolli web più comuni (HTML, URI, HTTP) non sono adeguati all'IoT perché forniscono una capacità di calcolo molto ridotta (8 o 16 bit), sono anche di difficile gestione dei protocolli più complessi (IPv6, TCP, SSL/TLS) perché troppo pesanti computazionalmente. É nata quindi l'esigenza di introdurre dei protocolli applicativi con paradigmi nuovi ed adatti al mondo dell'Internet of Things.

7.3.1 CoAP (Constrained Application Protocol)

É uno dei protocolli applicativi principali sviluppati per garantire l'interoperabilità con il Web e le sue principali operazioni (GET, PUT, POST, DELETE). L'unità di base per il livello logico e di gestione è il documento, per questo è detto *document-centric*. Funziona attraverso un'architettura del tipo richiesta/risposta. Ha uno standard di sicurezza di livello alto fornita dal DTLS (Datagram Transport Layer Security) e basato su UDP. Lo svantaggio di quest'ultimo riguarda il numero e la qualità minore dei servizi offerti rispetti al TCP, in compenso è molto più snello.

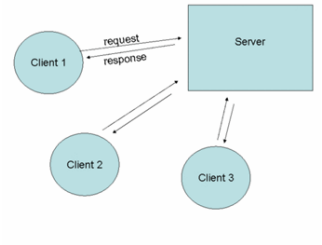


Figura 7.4: Interaction Model

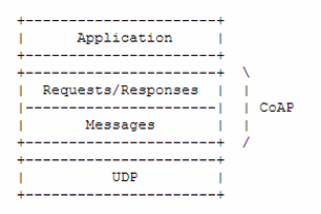


Figura 7.5: Two Layer Approach

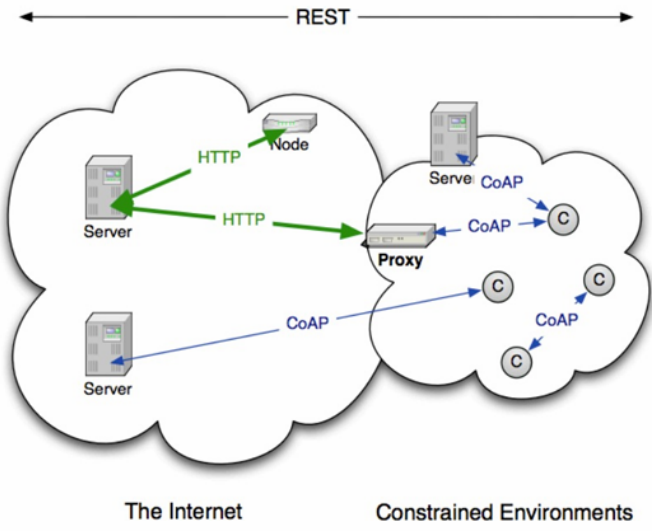


Figura 7.6: HTTP bridging

Il CoAP può essere facilmente convertito in HTTP ed è stato progettato per lavorare in ambienti limitati con scarsa larghezza di banda ed energia. La comunicazione però deve essere rapida ed ininterrotta, dunque è un'ottima scelta per la comunicazione M2M (machine-to-machine), D2D (device-to-device) e D2G (device-to-gataway).

7.3.2 MQTT (Message Queueing Telemetry Transport)

La finalità di questo protocollo è quella di supportare implementazioni operanti su sistemi embedded. Dispositivi di questo tipo presentano delle capacità di elaborazione e memoria limitate ma sono utili per lavorare sulle reti cablate. Non introduce meccanismi di sicurezza ma si appoggia sul TCP, è quindi molto più robusto ed adatto in ambito industriale. Il suo meccanismo Publish/Subscribe permette comunicazioni di tipo *punto-multipunto*.

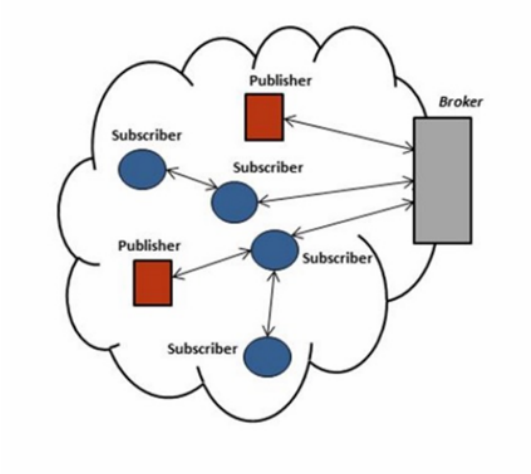


Figura 7.7: MQTT

Sono definiti tre livelli di QoS (Quality of Service):

1. *at most once* - viene trasmesso il dato ma non è garantita la sua ricezione. Non si è iteressati a difendere quel dato, se va perduto verrà rinviato;

2. *at least once* - viene garantita la ricezione del dato rispettando determinati vincoli;

3. *exactly once* - viene garantito che il dato arrivi a destinazione e che non vengano inviate delle repliche (usato ad esempio per inviare bonifici bancari).

7.3.3 MQTT-SN

É una versiona più snella, quindi idonea all'IoT, del MQTT adattata agli ambienti wireless; dove SN sta per Sensor Networks.

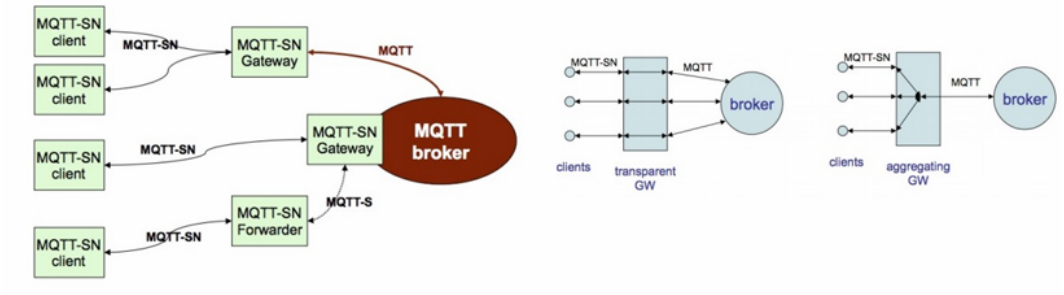


Figura 7.8: MQTT-SN

Il gateway gestisce lo stack, convertendo i pacchetti MQTT-SN in quelli MQTT per comunicare quest'ultimi attraverso dei broker. Quindi esso può permette di interconnettere una rete IoT con una non-wireless che usa MQTT.

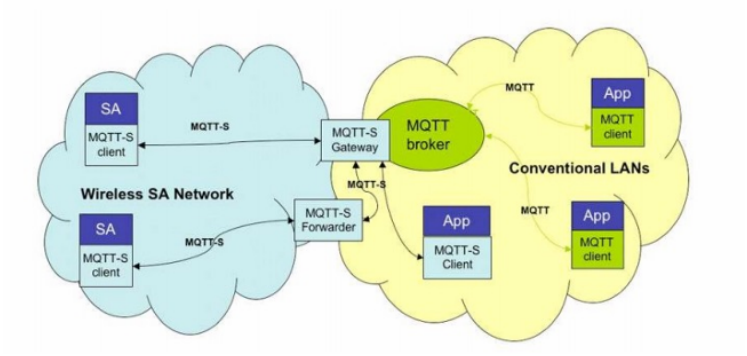
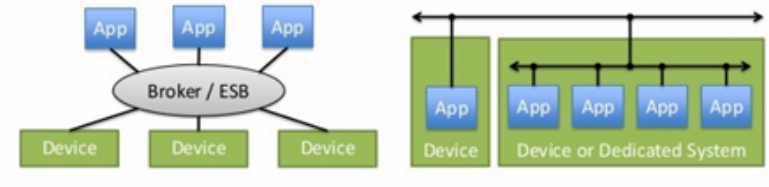


Figura 7.9: Interconnessione di reti MQTT ed MQTT-SN

A differenza del MQTT che fa affidamento su protocolli TCP/IP, l'MQTT-SN utilizza UDP. Inoltre, la dimensione massima dei pacchetti diminuisce drasticamente, allineandosi con quella ZigBee. In compenso però sono introdotte le funzionalità specifiche per le reti wireless, come la capacità di funzionare usando alimentazione a batteria e la capacità di gestire lo sleep dei dispositivi.

7.3.4 Data Distribution Service (DDS)

É una versione migliorata del MQTT, in grado di funzionare su sistemi con delle richieste esigenti in termini di qualità del servizio (*mission-critical*). Tipicamente è orientato all'ambiente cablato ed industriale, mirando a comunicazioni in tempo reale.



7.3.5 Advanced Message Queueing Protocol (AMQP)

Questo protocollo è usato nel contesto bancario (delle transazioni economiche). La sicurezza è importante per garantire un'elevata qualità del servizio.

7.3.6 eXtensible Messaging and Presence Protocol (XMPP)

Questo protocollo è usato per la messaggistica istantanea ed il gaming online. Non è progettato per essere veloce ma facile da indirizzare, sicuro e scalabile. É un protocollo di presenza perché ha delle figure intermedie (i broker) che raccolgono i messaggi e le informazioni da una delle due entità comunicanti e la forniscono all'altra entità, dando l'impressione che l'altro utente sia sempre presente.

7.4 Industrial Networks

7.4.1 Complex Control Systems

Sono dei sistemi a larga scala composti da sottosistemi che includono *sensori* ed *attuatori*. Nel passato era tutto centralizzato ma al giorno d'oggi si sta cercando di rendere il tutto gerarchico (dove abbiamo due livelli di controllo: locale e centralizzato dalla workstation) oppure una rete intelligente avente bus dedicati.

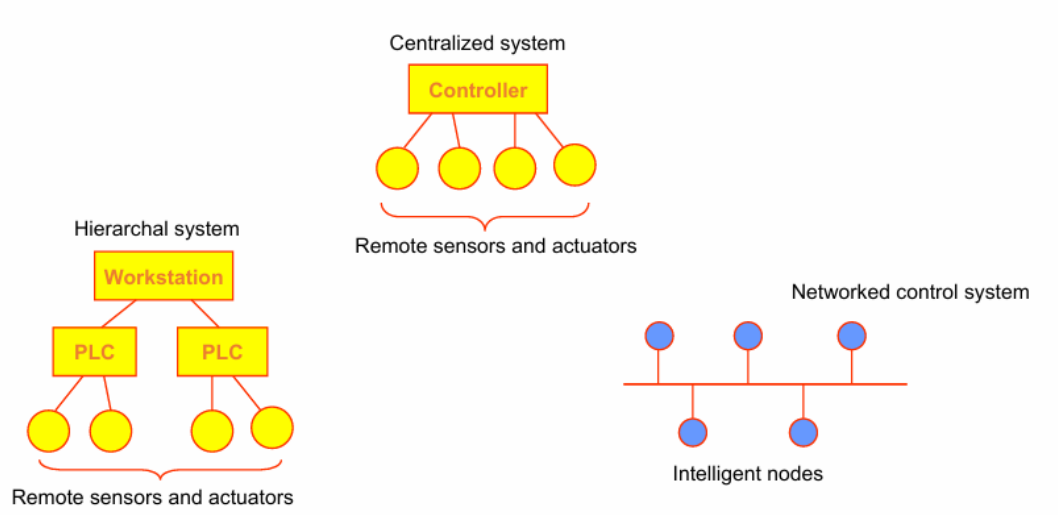


Figura 7.10: Evoluzione dei control systems

8 Short Range Technologies

8.1 RFID (Radio Frequency Identification)

Conosciuti anche come **tag**, perché spesso sono realizzati sotto forma di etichette. Possono essere di due tipi:

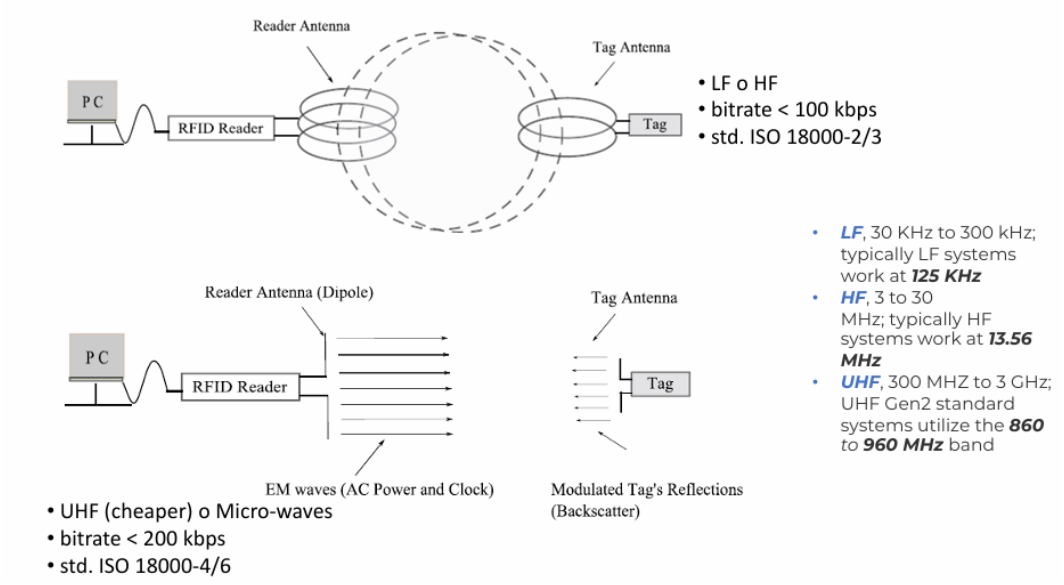
1. *passivi* - quando non presentano un trasmettitore, la trasmissione avviene, quindi, esclusivamente attraverso un meccanismo di rimbalzo. Non sono alimentati a batteria;

2. *attivi* - presentano un apparato ricetrasmettitore, quindi ha la capacità di trasmettere in maniera indipendente. Sono alimentati a batteria.

8.1.1 Protocollo di comunicazione

Si ha un reader, ovvero un dispositivo esterno senza problemi energetici che irradia una richiesta di presenza nell'ambiente. Tutti i RFID (sia attivi che passivi) rispondono alla richiesta del reader con la loro identità.

Per questo motivo sono usati nei grandi magazzini per l'inventario.



Gli RFID funzionano in maniera induttiva (raccolgono l'energia elettromagnetica attraverso un processo di induzione). L'antenna del reader trasmette e quella del tag riceve. Possono operare su LF (Low Frequency), HF (High Frequency) e UHF (Ultra High Frequency).

Il principale controllo di comunicazione è il Reader Talk First (RTF) in cui il reader parla per il primo ed i tag rispondono poco dopo essere stati interpellati (tipicamente rispondono tutti assieme). Sorge la necessità di ideare strategie per evitare le collisioni dei pacchetti in ambienti industriali, quindi protocolli di accesso casuale che funzionano in maniera precisa e meticolosa.

8.1.2 Anti-Collision Algorithms

I protocolli anti-collisione si dividono in due categorie:
quelli basati su ALOHA e quelli ad albero.

	Tree	ALOHA
How does it work?	Tags are progressively grouped is smaller and smaller sets till only one tag belongs to each sub-set and collisions are avoided. A tree of queries is built in this way that can be used till a topology change arise.	ALOHA is used to regulate packet transmissions: collisions can arise
How a new joining tag is handled?	A new tree needs to be built.	The tag can be immediately recognized
Frequency band	UHF or micro-waves	LF or HF
Delays	Very low even in presence of a high number of tags	Pretty high in presence of a high density of tags
Behaviour	Deterministic	Pseudo-random
Standard	ISO 14443-3 Type-A, ISO-18000-6B, EPCglobal (Class 0 / 1 UHF)	ISO 18000-3 (MODE 1 / 2), ISO 14443-3 Type-B, ISO-18000-6, EPCglobal (Class 1 HF)

Figura 8.1: (Tree vs ALOHA) Protocols

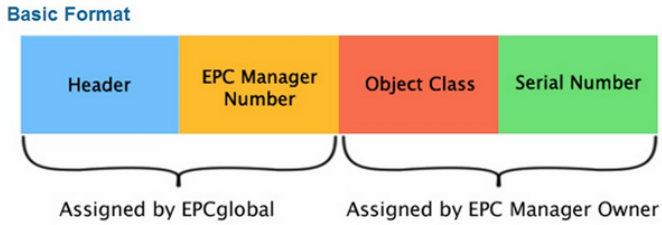
Query Tree Algorithms

Fra i più comuni abbiamo:

- **Tree Splitting** - quando si verifica una collisione tra i tag scelgono un numero casuale e si dividono in base ad esso in subset disgiunti, ripetendo la procedura fino a quando un tag risponde e non si verifica alcuna collisione;
- **Query Tree** - il reader invia una query con un solo bit e solo i tag aventi lo stesso prefisso possono rispondere, se si verifica una collisione il reader allarga la query fino a quando risponde un tag;
- **Binary Tree** - il reader invia una query con una parola binaria e solo i tag avevnti identificativo minore o uguale ad essa rispondono, se si verifica una collisione la parola viene ridotta fino a quando solo un singolo tag risponde.

8.1.3 Standard Electronic Product Code (EPC)

Il tag risponde al reader fornendo il suo EPC, ossia il codice prodotto.



Metà degll'indirizzo è assegnato ad un organo che controlla gli standard a livello globale, l'altra metà è proprietaria della categoria del bene dell'oggetto.

- Header: identifica la lunghezza, il tipo, la struttura, la versione dell'EPC.
- EPC Managaer Number: entità responsabile al mantenimento delle partizioni.
- Object Class: Identifica una classe di oggetti.
- Serial Number: identifica l'istanza.

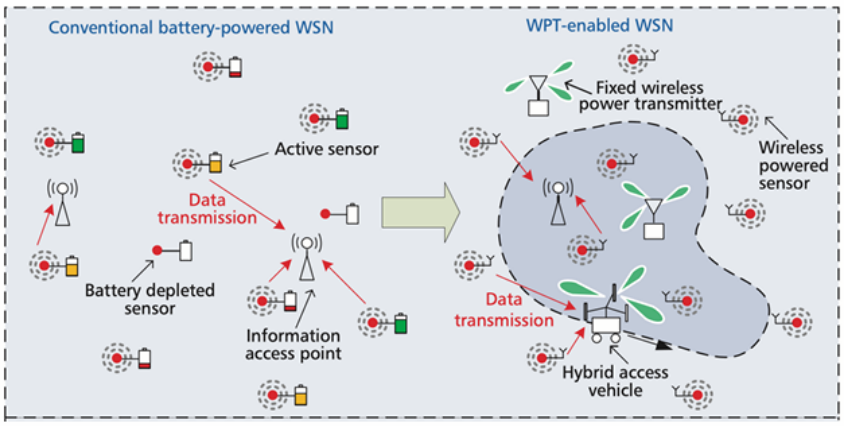
8.2 WPCNs (Wireless Powered Comunication Networks)

Reti wireless in grado di funzionare senza l'uso delle batterie. Le tecnologie utilizzate in queste reti sono:

- *EH* - le quali "raccolgono" energia e la utilizzano per alimentare i dispositivi. L'energia viene rubata dall'ambiente. Le sorgenti che forniscono energia però sono casuali e tempo varianti.
- *WPT* - realizzano architetture in grado di ricarica i dispositivi da remoto avendo il pieno controllo del trasferimento di potenza.

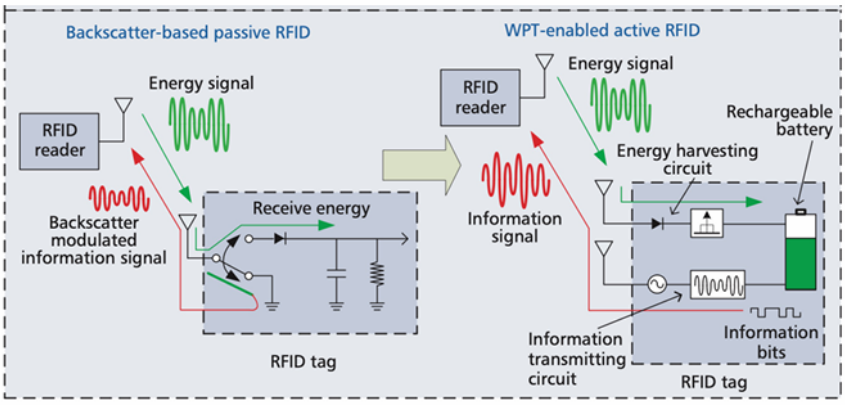
Energy-Hervesting (EH) Technologies	Wireless Power Transfer (WPT) Technologies
Wireless Devices (WDs) opportunistically harness renewable energy	Battery of WDs remotely replenished
Environment not dedicated to power the WDs	Microwave WPT technology enables WPCNs
energy srces mostly random and time varying	A WPCN has full control over its power transfer

Tabella 8.1: EH vs WPT

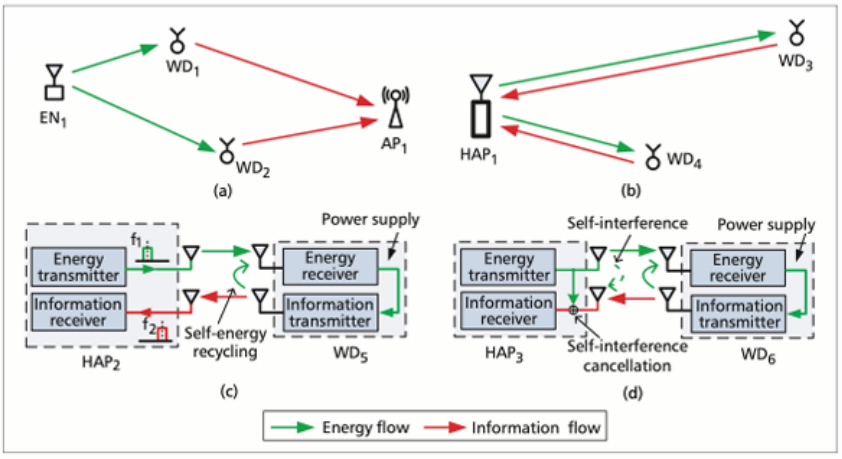


Nell'immagine precedente è rappresentata una rete convenzionale di dispositivi alimentati a batteria, WSN (Wireless Sensor Network) ed un'altra con tecnologia WPT. Si verificano dei problemi di interferenza co-canale.

I dispositivi irradiano energia ad una potenza di $-10dB_m$ per decodificare l'informazione necessitano, invece, di una potenza di $-60dB_m$.



Per risolvere questo problema si introduce il concetto di pacchetti di energia. Nell'immaggie in basso sono riportati un esempio di tag RFID passivo ed uno attivo. I segnali in verde trasportano energia, quelli in rosso informazione.

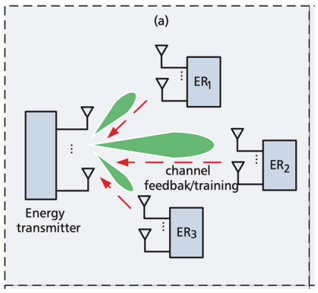


Nella figura in altro EN è il nodo responsabile della ricarica energetica dei dispositivi, quindi non ha finalità informative ma esclusivamente energetiche. I nodi WD sono i dispositivi wireless (ovvero dispositivi low-cost che inviano i loro pacchetti ai nodi AP, gli acces point). l'HAP è un acces point ibrido che può sia raccogliere le informazioni che trasferire energia.

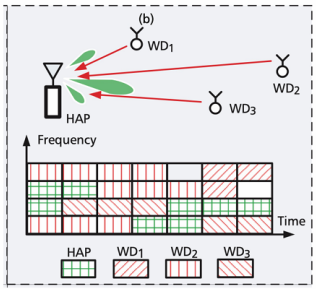
I dispositivi che inviano le informazioni da un'antenna e ricevono energia dall'altra non hanno problemi e si introduce il concetto di *self-energy recycling*. I dispositivi che utilizzano la stessa banda in diversi time-slot vengono detti *half-duplex*. Quando invece un dispositivo prova sia a trasmettere energia che segnale contemperaneamente viene detto *full-duplex*; questi introducono il SIC (Self-Interface Cancellation) con il quale eliminiamo il rumore (ottenuto quando il segnale di energia va a cancellare l'informazione ricevuta) semplicemente cancellando il segnale ricevuto.

Key Techniques:

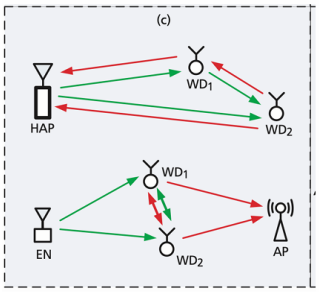
- **Energy beamforming**



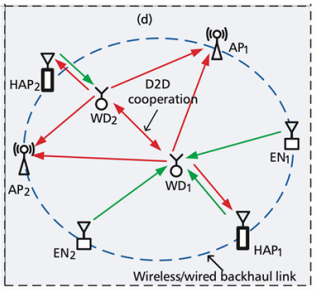
- **Joint communication/energy scheduling**



- **Wireless powered cooperative communication**



- **Multi-node cooperation**



8.2.1 Hybrid Energy Sources

I sistemi ibridi utilizzano sia l'energia degli access point dedicati che l'energia opportunistica. La raccolta di quest'ultima può essere legata, ad esempio, al sole (ma anche a dispositivi bluetooth nelle vicinanze). [Finché è presente, uso l'energia messa a disposizione nell'ambiente].

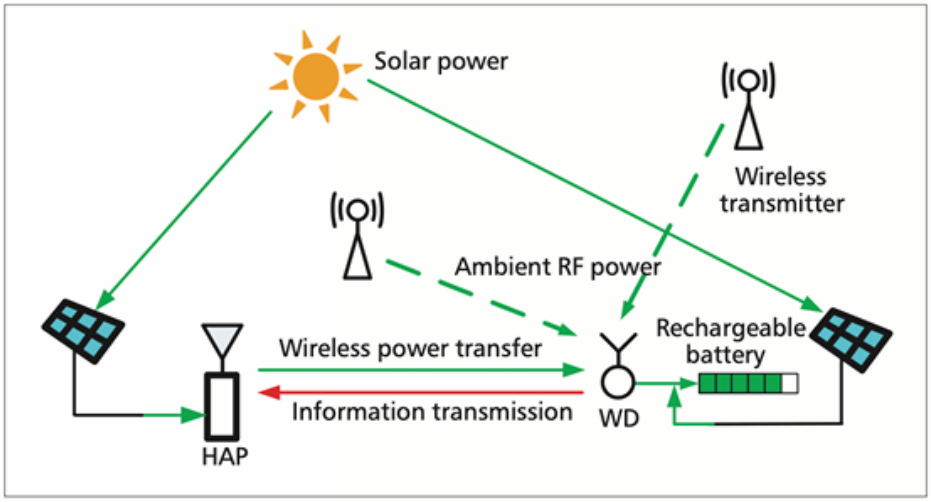


Figura 8.2: Sistema Ibrido

8.2.2 Spectrum Sharing and Coexistence

La coesistenza tra WPCNs comporta enormi problematica.

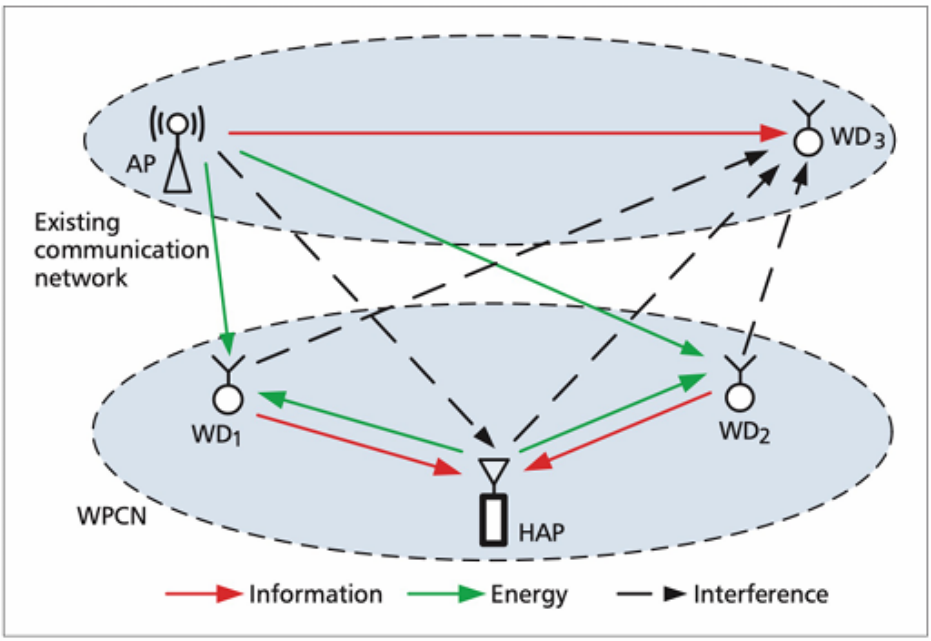


Figura 8.3: Coesistenza tra WPCNs