

# Reti Wireless E Mobili

Massimo Perego

## Contents

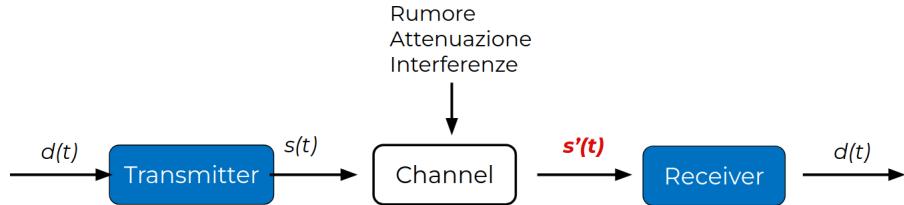
|   |           |
|---|-----------|
| <b>1 Principi di Teoria della Trasmissione</b>              | <b>4</b>  |
| 1.1 Rappresentazione dei segnali . . . . .                  | 5         |
| 1.2 Relazione tra Bandwidth e Data Rate . . . . .           | 8         |
| 1.2.1 Teorema di Nyquist sulla banda . . . . .              | 10        |
| 1.2.2 Decibel . . . . .                                     | 12        |
| 1.2.3 Rapporto segnale rumore SNR . . . . .                 | 12        |
| 1.2.4 Shannon Capacity Formula . . . . .                    | 13        |
| 1.3 Multiplexing . . . . .                                  | 15        |
| 1.3.1 Frequency Division Multiplexing FDM . . . . .         | 15        |
| 1.3.2 Time division Multiplexing TDM . . . . .              | 15        |
| <b>2 Comunicazione Wireless</b>                             | <b>16</b> |
| 2.1 Encoding, symbol e symbol rate . . . . .                | 18        |
| 2.2 Trasmissione delle onde radio . . . . .                 | 19        |
| 2.2.1 Problemi con Trasmissione radio Line of Sight (LoS) . | 20        |
| 2.3 Codifica e Trasmissione Dati . . . . .                  | 27        |
| 2.3.1 Modulazione e Codifica . . . . .                      | 27        |
| 2.3.2 Bit Error Rate Curve . . . . .                        | 30        |
| 2.3.3 Forward Error Correction . . . . .                    | 31        |
| 2.3.4 Orthogonal Frequency Division Multiplexing OFDM .     | 34        |
| 2.4 Spread Spectrum . . . . .                               | 36        |
| 2.4.1 Frequency Hopping Spread Spectrum FHSS . . . . .      | 37        |
| 2.4.2 Direct Sequence Spread Spectrum DSSS . . . . .        | 38        |
| 2.4.3 Code Division Multiple Access CDMA . . . . .          | 39        |
| <b>3 Wireless Personal Area Network WPAN</b>                | <b>42</b> |
| 3.1 ISM Band . . . . .                                      | 42        |
| 3.2 Pulse Code Modulation PCM . . . . .                     | 42        |

|          |   |            |
|----------|---|------------|
| 3.3      | 802.15.x . . . . .  | 43         |
| 3.4      | Bluetooth . . . . .   | 43         |
| 3.4.1    | Architettura dei protocolli . . . . .                         | 44         |
| 3.4.2    | Piconet & Scatternet . . . . .                                | 47         |
| 3.4.3    | Bluetooth Radio . . . . .                                     | 48         |
| 3.4.4    | Baseband . . . . .  | 51         |
| 3.4.5    | Link Manager Protocol . . . . .                               | 54         |
| 3.4.6    | Logical Link Control and Adaptation Protocol L2CAP            | 57         |
| 3.4.7    | Service Discovery Protocol SDP . . . . .                      | 58         |
| 3.5      | Bluetooth Low Energy BLE . . . . .                            | 59         |
| 3.5.1    | Architettura . . . . .  | 60         |
| 3.5.2    | Classi di potenza . . . . .                                   | 60         |
| 3.5.3    | BLE Radio (PHY) . . . . .                                     | 60         |
| 3.5.4    | BLE State Machine . . . . .                                   | 61         |
| 3.5.5    | Advertising . . . . .   | 62         |
| 3.5.6    | Generic Attribute Profile GATT . . . . .                      | 62         |
| 3.5.7    | General Access Protocol GAP . . . . .                         | 63         |
| 3.6      | ZigBee . . . . .  | 65         |
| 3.6.1    | Livello Fisico 802.15.4 (PHY) . . . . .                       | 67         |
| 3.6.2    | Livello Data Link 802.15.4 (MAC) . . . . .                    | 68         |
| 3.6.3    | Livello di rete . . . . .                                     | 74         |
| 3.6.4    | ZigBee Device Object ZDO . . . . .                            | 74         |
| 3.7      | Matter & Thread . . . . .                                     | 75         |
| <b>4</b> | <b>WiFi 802.11</b>  | <b>77</b>  |
| 4.1      | Sottolivello MAC . . . . .                                    | 80         |
| 4.1.1    | Distributed Coordination Function DCF . . . . .               | 80         |
| 4.1.2    | Problema del terminale nascosto . . . . .                     | 83         |
| 4.2      | 802.11 Con Infrastruttura . . . . .                           | 85         |
| 4.2.1    | Point Coordination Function . . . . .                         | 86         |
| 4.2.2    | Formato Frame MAC . . . . .                                   | 88         |
| 4.3      | Orthogonal Frequency Division Multiple Access OFDMA . . . . . | 90         |
| 4.3.1    | Downlink DL-OFDMA . . . . .                                   | 93         |
| 4.3.2    | Uplink UP-OFDMA . . . . .                                     | 94         |
| 4.4      | WLAN Security . . . . .                                       | 96         |
| 4.5      | WiFi Protected Setup WPS . . . . .                            | 99         |
| 4.6      | 802.11e EDCA Enhanced Distributed Channel Access . . . . .    | 100        |
| 4.7      | 802.11p & Wave - V2V . . . . .                                | 100        |
| <b>5</b> | <b>Ad Hoc Distance Vector Routing Protocol AODV</b>           | <b>102</b> |

|          |  |            |
|----------|--|------------|
| 5.1      | Tabelle di Routing . . . . .                     | 104        |
| 5.2      | RREQ . . . . .                                   | 106        |
| 5.2.1    | Expanding Ring Search . . . . .                  | 107        |
| 5.2.2    | RREQ Processamento e Inoltro . . . . .           | 108        |
| 5.3      | RREP . . . . .                                   | 109        |
| 5.3.1    | RREP Processamento e Inoltro . . . . .           | 111        |
| 5.3.2    | Hello Message . . . . .                          | 112        |
| 5.4      | Mantenimento della Connettività Locale . . . . . | 113        |
| 5.4.1    | RERR Processamento e Inoltro . . . . .           | 114        |
| 5.4.2    | Local Repair . . . . .                           | 115        |
| 5.4.3    | Reboot . . . . .                                 | 115        |
| <b>6</b> | <b>Mobile Network</b>                            | <b>116</b> |
| 6.1      | Introduzione alle reti mobili . . . . .          | 116        |
| 6.1.1    | Organizzazione Geometrica delle Celle . . . . .  | 118        |

# 1 Principi di Teoria della Trasmissione

Vogliamo **trasmettere informazioni** binarie su un **mezzo analogico**. Tipicamente abbiamo uno schema del tipo:



Si hanno dati nel tempo i quali passano da un **trasmettitore** il quale “traluce” i dati **digitali** in un segnale **analogico**, il quale può essere **trasmesso su un mezzo analogico** (e.g., aria, cavi, ecc.). Il segnale attraversa il mezzo e arriva a un ricevitore, il quale decodifica il segnale per tornare ai dati originali. Però, il canale non è perfetto o perfettamente affidabile, è possibile **introduca**:

- Il **rumore** in mezzi wireless può essere importante
- Il ricevitore deve essere in grado di recepire piccole quantità di potenza, in quanto il segnale potrebbe essere **attenuato**
- **Interferenze**, “casuali” dovute alla stessa tecnologia (propagazione del mezzo, ecc.) oppure volontarie (e.g., jamming).

Quindi il **segnale inviato** risulterà **diverso** dal **segnale ricevuto**, ma se il segnale viene strutturato correttamente ed è robusto a queste deformazioni il ricevitore sarà in grado di estrarre ugualmente dati dal segnale “modificato”. Quanto è facile ricostruire il segnale dipende da **quanto** i fenomeni di disturbo hanno **modificato** il segnale, può pure non essere ricostruibile.

Per **segnaile**

- **analogico**: si intende una variazione continua, senza interruzioni o discontinuità
- **digitale**: un livello di segnale viene mantenuto costante per un determinato intervallo, con un cambio di livello rapido (quasi istantaneo)

Vogliamo passare da una forma d'onda all'altra (trasmettitore e ricevitore fanno questo).

## 1.1 Rappresentazione dei segnali

**Dominio del tempo:** Un segnale può essere visto nel dominio del tempo come un **segnale periodico sinusoidale**

$$s(t) = A \sin(2\pi ft + \phi)$$

Dove:

- **Aampiezza** ( $A$ ): massimo livello o forza del segnale nel tempo (Volt)
- **Frequenza** ( $f$ ): Numero di cicli al secondo (Hz)
- **Fase** ( $\phi$ ): posizione relativa all'interno del periodo
- **Periodo** ( $T$ ): tempo impiegato per un ciclo ( $1/f$ )
- **Lunghezza d'onda** ( $\lambda$ ): distanza occupata da un singolo ciclo:  $\lambda = c/f$  oppure  $\lambda = Tc$  dove  $c = 3 \cdot 10^8$  m/s

La variazione di ampiezza, fase e frequenza vengono usate per codificare le informazioni (e.g., radio AM e FM, modulazione di fase non per le radio, ma alter cose).

**Dominio delle frequenze:** Possiamo considerare un'onda elettromagnetica guardandola nel tempo, ma anche nel dominio delle frequenze. **Ogni segnale** (ragionevolmente periodico) **può essere scomposto da una serie di segnali periodici** (onde seno e coseno) con ampiezza, frequenza e fase differenti: trasformata di Fourier

$$s(t) = \frac{1}{2}c + \sum_{n=1}^{\infty} a_n \cdot \sin(2\pi nft) + \sum_{n=1}^{\infty} b_n \cdot \cos(2\pi nft)$$

Dove:

- $f = 1/T$ : frequenza **fondamentale** ( $n = 1$ )
- $a_n, b_n$ : **ampiezza** delle **singole componenti**, dette armoniche
- $c$ : costante che rappresenta il **valore medio** del **segnale**

Possiamo **scomporre** un segnale in **diverse armoniche**, ognuna con il suo “contributo” rispetto al segnale originale. Questa è la serie di Fourier discreta, in un calcolatore saranno un numero finito di frequenze e la versione originale era con gli integrali, nel continuo.

Insomma, serve per passare da un dominio all'altro.

**Cosa ci interessa:** Dal punto di vista di un ricevitore, il quale riceverà tali segnali, ci interessa capire **com'è composta l'onda** a partire **da un'osservazione nel tempo** di quest'onda. Come faccio a **risalire alle componenti** a partire da un'osservazione nel tempo della forma d'onda? Le domande sono:

1. Come si fa a **determinare le ampiezze** di ciascuna componente
2. Con quale **frequenza campionare il segnale?** Il mondo digitale è discreto per definizione, in che punti della curva bisogna “leggere” per poter ricostruire l'onda in maniera precisa

**Teorema del campionamento di Shannon:** La **frequenza di campionamento** deve essere almeno il **doppio della frequenza massima del segnale** in ingresso.

Sapendo la frequenza massima (e tra ricevitore e trasmettitore ci si mette d'accordo), devo campionare ad almeno il doppio per evitare perdita di dati.

**Passaggi di dominio:** Per passare da un dominio all'altro abbiamo due algoritmi:

- **Fast Fourier Transform FFT:** da tempo a frequenze, passandogli la forma d'onda nel tempo restituisce le componenti
- **Inverse Fast Fourier Transform IFFT:** da frequenze a tempo, partendo dalle componenti restituisce la forma d'onda

Con “componenti” si intende i coefficienti  $a_n$  e  $b_n$  viste nella serie di Fourier. Questi algoritmi sono semplici, vengono implementati tramite hardware nei dispositivi, i quali devono effettuarle costantemente.

Assegnando dei bit al fingerprint di una forma d'onda (valori della trasformata), posso creare una lookup table per trovare il “significato” di una forma d'onda a partire dalla sua trasformata (osservo l'onda, faccio la trasformata, lookup per il significato).

Nel dominio delle frequenze: quando **tutte le frequenze** sono **multipli interi di una frequenza base**:

- $f = \text{frequenza fondamentale}$
- $kf = \text{armonica} (k > 1)$

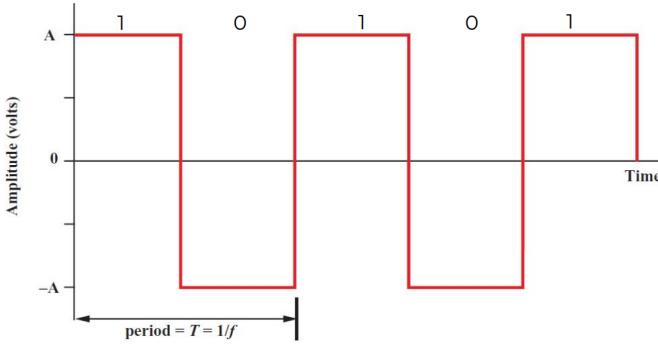
**Periodo:** Il periodo di  $s(t)$  è il periodo della frequenza fondamentale ( $T = 1/f$ ).

**Spettro:** Lo spettro del segnale è il range di frequenze che lo contiene (da dove a dove vanno le frequenze).

**Banda:** Absolute bandwidth è l'ampiezza dello spettro (“larghezza” dello spettro).

## 1.2 Relazione tra Bandwidth e Data Rate

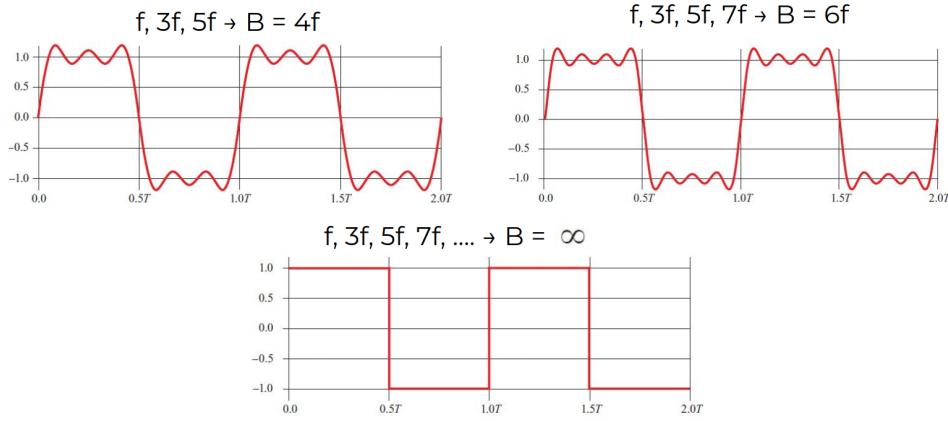
Esempio: vogliamo trasmettere un'onda quadra come **composizione finita di onde sinusoidali**. Esempio:



Trasmettiamo 2 bit per ogni periodo, ovvero un data rate di 2 bit. Possiamo avere una **sommatoria infinita** di onde **sinusoidali**:

$$s(t) = A \cdot \frac{4}{\pi} \cdot \sum_{k \text{ odd}, k=1}^{\infty} \frac{\sin(2\pi k f t)}{k}$$

Ma questo richiederebbe una banda infinita, il che è difficile, quindi possiamo **ridurre lo spettro** per ottenere un'**approssimazione**:



Nel primo esempio una banda di  $4f$  che va da  $f$  a  $5f$ , nel secondo aumentiamo la banda a  $6f$ , migliorando l'approssimazione. Con la banda che tende all'infinito otteniamo l'onda originale.

Con una certa banda, **quanti dati possiamo trasmettere?** Esempio:

| Freq. fondamentale ( $f$ ) | 1 MHz                                  | 2 MHz                                  |
|----------------------------|--|--|
| Spettro                    | 1 Mhz - 5 Mhz                          | 2 Mhz - 10 Mhz                         |
| Periodo (T)                | $1 \mu s$                              | $0.5 \mu s$                            |
| Durata di 1 bit            | $0.5 \mu s$                            | $0.25 \mu s$                           |
| Bandwidth (B)              | $4 \text{ Mhz} (5f - f)$               | $8 \text{ Mhz} (2(5f - f))$            |
| Data rate (bps)            | $2 \text{ Mbps} (2 \text{ bit}/\mu s)$ | $4 \text{ Mbps} (4 \text{ bit}/\mu s)$ |

Con il doppio della banda abbiamo raddoppiato il data rate.

**Capacità del canale:** Quanti bit possiamo trasmettere sul canale senza perdere informazioni?

- **Channel capacity:** massimo bit rate alla quale è possibile trasmettere dati su un canale di comunicazione in determinate condizioni
- **Noise:** segnale NON voluto che si combina al segnale trasmesso, distorcendolo
- **Error rate:** tasso di errore (bit error rate), quante volte viene modificato involontariamente il segnale

Come possiamo trasmettere la **stessa quantità di informazioni senza usare più banda?** La soluzione è **ridurre il numero di armoniche**, semplificando la forma d'onda e peggiorando l'approssimazione. Questo si può fare finché l'onda non è un'approssimazione “troppo approssimata” (deve essere distinguibile).

**Considerazioni:** Vogliamo trasmettere una banda infinita con banda finita, ma una banda minore porta a distorsione maggiore. Una soluzione potrebbe essere scegliere la banda finita più ampia; sarebbe fattibile ma ci sono costi economici (i.e., la banda non è gratis) e il dispositivo deve essere in grado di gestirla. Inoltre può creare rumore aggiuntivo.

### 1.2.1 Teorema di Nyquist sulla banda

Dato un canale noise-free (ideale) la bandwidth limita il data-rate. Il **limite della quantità di informazioni** (misurata in bit/secondo e multipli) è limitato da due volte la banda

$$C = 2B$$

per **segnali binari** (2 livelli di voltaggio). Per **segnali multilivello**, codificare i dati su più livelli di segnale

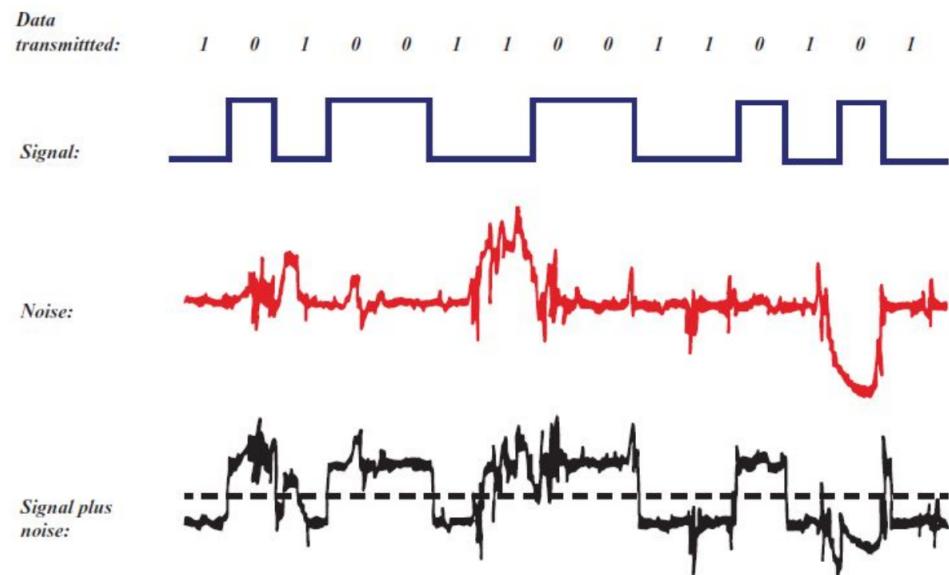
$$C = 2B \log_2 M$$

Dove  $M$  è il numero di livelli.

La **capacità del canale aumenta con il numero di possibili livelli** in cui codificare il segnale, modificando uno dei parametri (ampiezza, frequenza, fase). Il numero di livelli di segnale aumenta il numero di bit che si possono trasmettere con ogni trasmissione: con 2 valori trasmettiamo 1 bit, con 8 possibili valori possiamo trasmettere 3 bit alla volta.

Questo possibile solo se non c'è rumore, il massimo possibile. Tenendo il considerazione il rumore la capacità si abbassa.

Esempio di effetto del rumore sul segnale trasmesso:



Considerando il profilo temporaneo del rumore, il ricevitore riceve un segnale diverso, con modifiche su cui non abbiamo controllo, indipendenti da ricevitore e trasmettitore.

**Tipi di rumore:**

- Thermal noise: rumore di base dovuto all'agitazione delle molecole, un rumore bianco costante su tutta la banda
- Intermodulation noise: determinato dal fatto che ci sono dei problemi tra le diverse modulazioni, ci si "accavalla" nel trasmettere le informazioni
- Cross talk: cavi vicini che alterano il segnale l'uno dell'altro, tramite radiazioni elettromagnetico
- Impulse noise: impulso elettromagnetico intenso di durata limitata che attraverso il mezzo e distrugge temporaneamente il segnale

### 1.2.2 Decibel

Il decibel è un'**unità di misura del rapporto di potenze**, in scala **logaritmica**

$$\left(\frac{P_1}{P_2}\right)_{dB} = 10 \cdot \log_{10} \left(\frac{P_1}{P_2}\right)$$

$\pm 3$  corrispondono al doppio/metà.

**Decibel-Milliwatt:** Un decibel ma con la **potenza al denominatore fissata** ad  $1mW$

$$P_{dBm} = \frac{P}{1mW}$$

Generalmente una Wireless-LAN (WiFi 802.11) usa  $100mW$ , quindi  $20dBm$ , mentre, una trasmissione cellulare usa  $500mW$ , quindi  $27dBm$  (ricorda di mettere dopo il  $10\log_{10}$ ). In generale, una lettera dopo  $dB$  indica una potenza fissata al denominatore.

### 1.2.3 Rapporto segnale rumore SNR

Dobbiamo essere un grado di **quantificare il rumore** su un canale, quindi l'impatto sul rumore della trasmissione.

$$(SNR)_{dB} = 10 \log_{10} \frac{\text{signal power}}{\text{noise power}}$$

Misurato in decibel (rapporto tra due potenze), tanto più il rapporto è alto, tanto più il segnale si distingue dal rumore.

#### 1.2.4 Shannon Capacity Formula

Nyquist descrive la capacità del canale nel caso di assenza di rumore, Shannon dice che la **capacità dipende** non solo dalla capacità di codificare livelli ma **anche dal SNR**

$$C = B \log_2(1 + SNR)$$

Quindi la capacità è direttamente proporzionale al SNR. Questo valore è puramente teorico e **considera solo il thermal noise**, ma fornisce una **massima teorica per trasmettere informazioni senza errori** su un canale. La teoria della trasmissione di Shannon ha come assioma una trasmissione senza errori.

In una determinata condizione di rumore (SNR) possiamo aumentare il data rate in due modi:

1. Aumentando la bandwidth ( $B$ ), ma il rumore termico aumenta con la larghezza della banda
2. Aumentando la potenza del segnale trasmesso (quindi aumentando il valore del SNR), ma un aumento della potenza porta ad aumentare intermodulation e crosstalk noise (aumenta il campo elettromagnetico)

Posso aumentare SNR o  $B$  ed in entrambi i casi l'aumento viene "bilanciato" da un aumento del rumore.

**Esempio:** Supponiamo di avere a disposizione uno spettro tra  $3MHz$  e  $4MHz$  e  $SNR_{dB} = 24dB$ :

$$\begin{aligned} B &= 4MHz - 3MHz = 1MHz \\ SNR_{dB} &= 24dB = 10 \log_{10}(SNR) \\ \implies SNR &= 251 \end{aligned}$$

Il valore di  $SNR$  significa che il segnale è circa 251 volte superiore al rumore.  
Ora possiamo calcolare la capacità secondo Shannon:

$$C = 10^6 \cdot \log_2(1 + 251) \approx 10^6 \cdot 8 = 8Mbps$$

E da questa possiamo trovare quanti livelli di segnale servono per ottenere  $8Mbps$ , tramite la formula della capacità secondo Nyquist:

$$\begin{aligned} C &= 2B \log_2(M) \implies 8 \cdot 10^6 &= 2 \cdot 10^6 \cdot \log_2(M) \\ &\implies 4 &= \log_2(M) \\ &\implies M &= 16 \end{aligned}$$

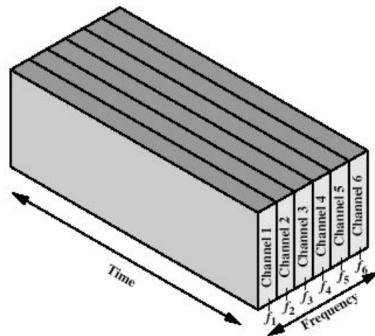
Di più di 16 sarebbe inutile perché trasmettendo un numero maggiore di livelli non possiamo superare la capacità teorica detta da Shannon. La capacità dipende dalla condizione del canale e dal numero di livelli.

## 1.3 Multiplexing

L'idea è quella di un collegamento che può trasportare più canali. Solitamente la capacità di un mezzo è superiore della capacità richiesta da una singola trasmissione quindi cerchiamo un modo di trasportare più segnali sullo stesso link.

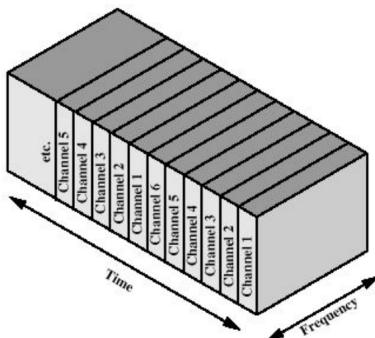
### 1.3.1 Frequency Division Multiplexing FDM

Se ho una banda "larga", posso dividerla in "slot" per ottenere sotto-bande con comunicazioni diverse, sfruttando il fatto che il segnale da trasmettere richiede una banda minore di quella disponibile. Creiamo  $n$  canali paralleli all'interno della banda.



### 1.3.2 Time division Multiplexing TDM

Se la banda è "piccola", ma il data rate è molto superiore a quello richiesto da una singola trasmissione, crea  $n$  slot di tempo ciclici.



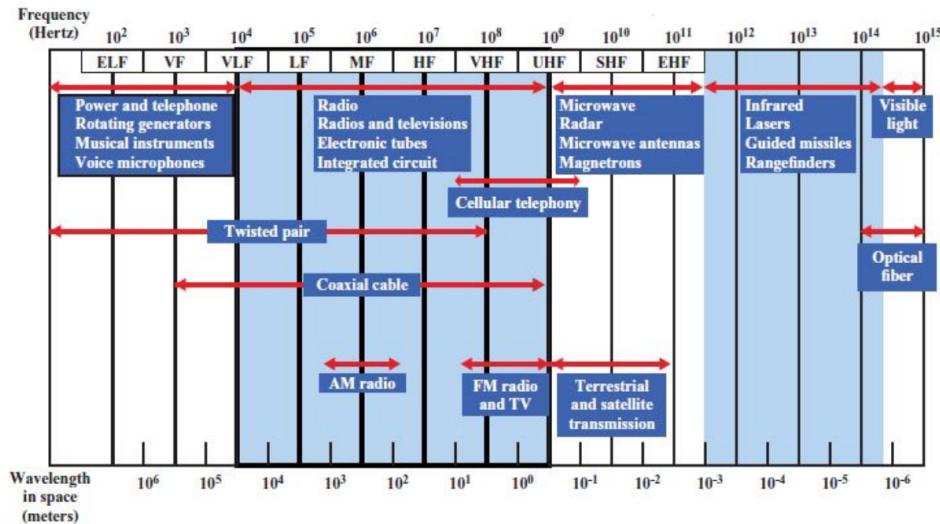
## 2 Comunicazione Wireless

Tutte le comunicazioni nel mondo wired avvengono tipicamente in banda base, i.e., data una banda  $B$  trasmetto direttamente usando lo spettro di frequenze  $[0, B]$ .

Problemi:

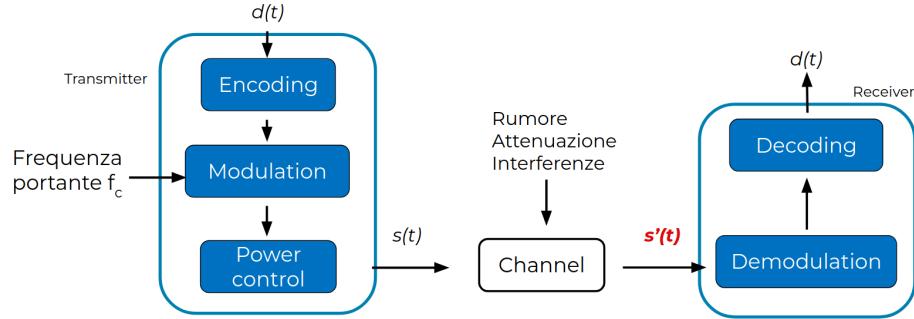
1. Se tutti i dispositivi trasmettessero in banda base, tutte le comunicazioni radio farebbero interferenza l'una con l'altra
2. Più è bassa la frequenza più grande deve essere l'antenna ( $\lambda/2$  per antenna dipole, es:  $1MHz \sim 142$  metri,  $2100MHz \sim 7$  cm)
3. Ogni range di radio frequenze possiede diverse proprietà i propagazione e attenuazione

**Spettro elettromagnetico per le telecomunicazioni:**



Molto approssimativo, c'è uno standard per ogni tipo di telecomunicazione (allocazione delle frequenze per gli USA).

**Trasmissione in banda traslata:** Da  $[0, B]$  spostiamo la banda all'interno di un altro range, ovvero  $[f_c - B/2, f_c + B/2]$ , la stessa banda ma traslata attorno ad una frequenza portante  $f_c$ .



Il compito del trasmettitore diventa codificare, **modulare attorno alla frequenza portante  $f_c$**  ed il trasmettitore deve de-modulare, pulire e de-codificare il segnale trasmesso. Bandwidth e data rate rimangono gli stessi, cambia solo “dov’è” la banda.

Domande:

1. Che spettro utilizzare? (che frequenza portante scegliere)
2. Come codifico i dati? (da digitale ad analogico)
3. Come modulo il mio segnale in banda base sulla frequenza portante?

## 2.1 Encoding, symbol e symbol rate

**Simbolo:** Una forma d'onda, uno stato o una condizione significativa del canale di comunicazione che persiste per un intervallo di tempo fissato. Esempio: voltaggio a 3V per 1 secondo.

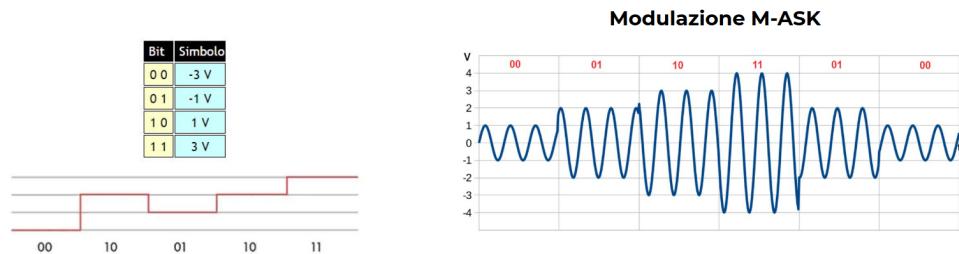
**Symbol rate:** Quantità di simboli trasmessi al secondo (misurato in baud). Dipende dal processore fisico.

In generale, un **simbolo può contenere più bit** (codifica e modulazione), quindi **symbol rate  $\neq$  bit rate**.

Generalmente, più la distanza da coprire si allunga più la durata si abbassa.

Una data bandwidth può supportare diversi data rate, a seconda dell'abilità del ricevente di distinguere 0 e 1 in presenza di rumore.

Possiamo codificare tramite una modulazione in ampiezza: data una entry di bit scegliamo il voltaggio corrispondente



Moduliamo la portante (modifichiamo l'ampiezza) in base ai bit da codificare. Usato nelle radio AM. Si può vedere che symbol rate e data rate sono diversi, ogni simbolo (livello di ampiezza) rappresenta due bit.

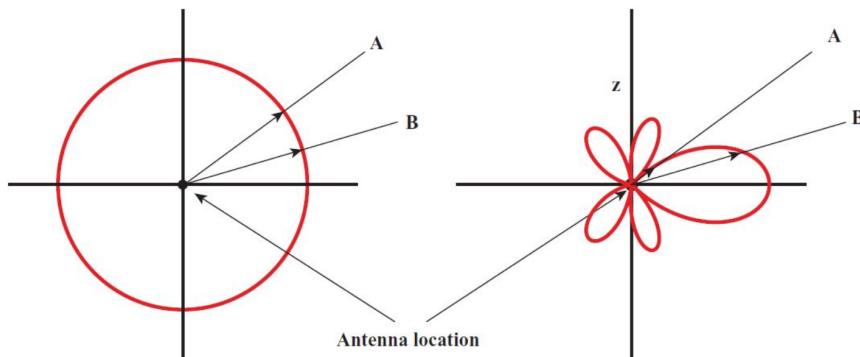
## 2.2 Trasmissione delle onde radio

**Propagazione onde radio:** Ci sono più possibili tipi di propagazione delle onde radio, in base alla frequenza utilizzata

- Ground Wave Propagation (sotto i  $2MHz$ ): il segnale segue la curvatura della terra
- Sky Wave Propagation (da 2 a  $30MHz$ ): il segnale rimbalza nella ionosfera
- **Line of Sight LoS** (sopra  $30MHz$ ): Deve esserci una linea diretta (anche non a vista in realtà, ma diretta) tra trasmettitore e ricevitore

### Tipi di antenne:

- Omnidirezionale (a sinistra), la potenza emessa è uguale in tutte le direzioni (ideale)
- Direzionale (a destra): si vuole propagare segnale in una sola direzione, non ideale, ma solitamente la propagazione non è limitata strettamente ad una direzione



### 2.2.1 Problemi con Trasmissione radio Line of Sight (LoS)

Ci sono problemi legati alla trasmissione LoS:

- Free space loss & path loss: attenuazione del segnale dovuta alla distanza e all'ambiente in cui il segnale si propaga; cosa perdo da antenna trasmettitrice (TX) a ricevitrice (RX)
- Rumore: disturbo che può distorcere il segnale
- Multipath: il segnale tra TX e RX può subire riflessioni, diffrazioni e scattering causando la ricezione di più onde elettromagnetiche dello stesso segnale in tempi diversi; per qualche motivo, un riflesso del segnale giunge al ricevitore, in momenti diversi dall'originale, possono diventare interferenze per segnali successivi
- Effetto Doppler: Il segnale cambia a causa del movimento di RT, TX e ostacoli; il segnale ricevuto potrebbe essere su una frequenza "vicina" a quella originale, disturbando la fingerprint di un segnale

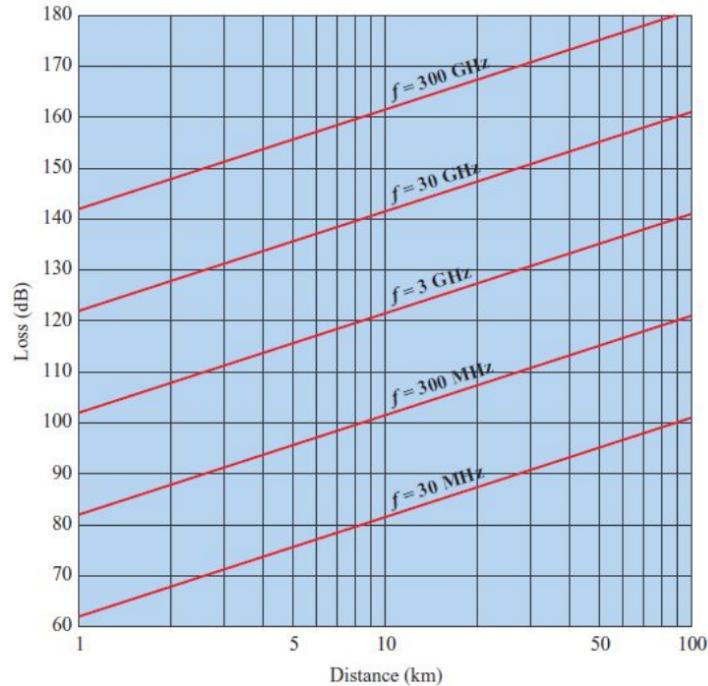
**Path Loss:** Attenuazione del segnale radio in funzione della distanza tra trasmettitore e ricevitore.

$$\frac{P_t}{P_r} = \left( \frac{4\pi}{\lambda} \right)^2 d^n = \left( \frac{4\pi f}{c} \right)^2 d^n$$

Misurata in  $dB$ , rapporto tra potenza del segnale trasmesso (fisso) rispetto alla potenza del segnale ricevuto (diventa sempre più piccolo, in base alla distanza).

Direttamente proporzionale (al quadrato della) alla frequenza, direttamente proporzionale ad una potenza della distanza, l'esponente  $n$  dipende dall'ambiente (più l'ambiente è ostruito più la distanza sarà influente).

**Free space loss ( $n = 2$ ):** A parità di distanza, maggiore è la frequenza maggiore è il path loss. La potenza di trasmissione massima è regolamentata (standard). A parità di potenza, maggiore è la frequenza minore è il raggio di copertura. Con “free space loss” si intende la perdita ideale in caso di spazio completamente libero, quindi con  $d^n = d^2$ .



Con  $300GHz$  la perdita ad  $1\text{km}$  è oltre i  $140db$ , mentre con  $30MHz$  è poco sopra i  $60dB$ . La distanza di trasmissione è limitata dalle frequenze utilizzate da una certa tecnologia.

**Antenna gain:** La diffusione di un’antenna ideale è isotropica (in tutte le direzioni), ma questo può essere inefficiente, capendo la posizione dei ricevitori si può migliorare la potenza ricevuta. L’idea di un’antenna direzionale è convogliare l’energia verso un determinato punto, un “beam” verso il ricevitore. Il segnale si propaga in una ellisse verso la direzione voluta dall’antenna, al contrario del cerchio creato da un’antenna isotropica.

Il **gain** di un’antenna viene definito come il rapporto tra l’intensità della radiazione elettromagnetica in una data direzione e l’intensità che si avrebbe se si usasse un’antenna isotropica, misurato in  $dB_i$ . La direzionalità viene misurata in  $dB_i$ , dove  $i$  sta per “isotropic”, ed è il rapporto tra la potenza che si avrebbe in quel punto con una antenna isotropica al denominatore e l’antenna attuale al numeratore. Concentrando verso il ricevitore “perdo” verso le altre direzioni.

Questo ha un **effetto sul path loss**; nel caso ideale avremmo allineamento perfetto tra TX e RX e di conseguenza un gain che riduce la perdita di potenza nella trasmissione.

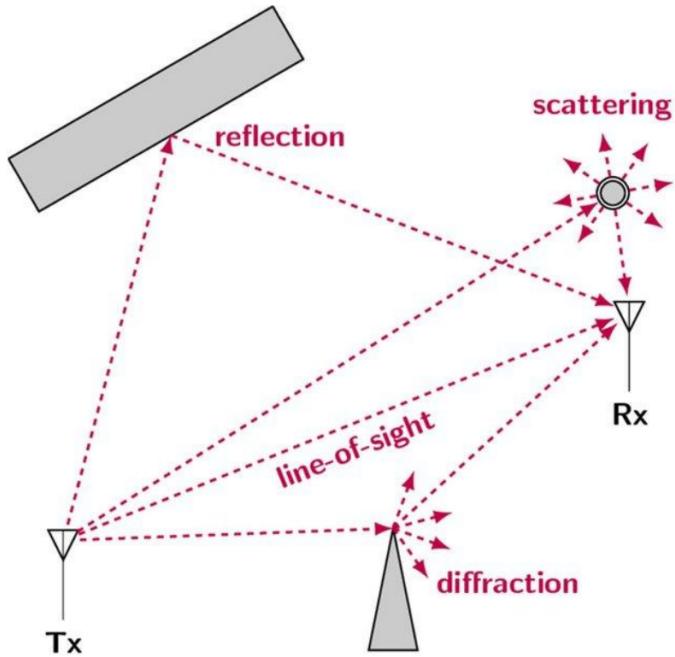
Alla perdita in free space, si sottraggono valori dipendenti dal gain di trasmittente e ricevente, abbassando il rapporto e riducendo la perdita. Chiamando  $G_{tx}$  e  $G_{rx}$  gain di trasmittitore e ricevitore, rispettivamente:

$$\frac{P_t}{P_r} = \frac{(4\pi f)^2}{G_{tx}G_{rx}c^2} d^n$$

Da ricordare che è necessaria un allineamento, se il ricevitore è fuori dal “beam” allora la potenza ricevuta sarà minore di quella di un’antenna isotropica, a parità di distanza ( $dB_i$  negativi, il gain peggiora).

**Multipath:** Anche se direzionale, il segnale avrà una “fascia” di direzioni in cui viene inviato, quindi può interagire con l’ambiente:

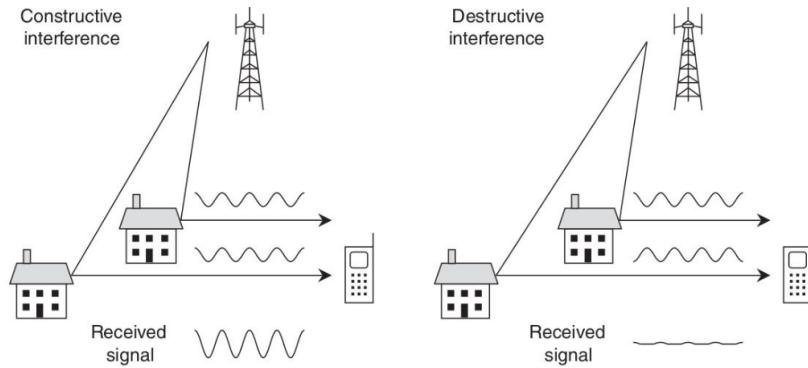
- **riflessione:** il segnale “rimbalza”, come su uno specchio,
- **scattering:** quando la lunghezza d’onda  $\lambda$  è simile a quella dell’oggetto, un raggio colpisce l’elemento e viene separato in tanti altri, sparsi in tutte le direzioni
- **diffrazione:** se la lunghezza d’onda è  $\lambda \ll$  della dimensione di un oggetto e lo colpisce sui bordi, diffondendo il segnale in più direzioni dall’altro lato dell’oggetto



I percorsi che non sono LoS saranno più lunghi e, di conseguenza, più lenti.

**Fading:** Quando più segnali vengono ricevuti in tempi diversi ci sono due tipi di interferenze:

- **costruttiva:** “fa bene”, aumenta l’ampiezza ed il segnale
- **distruttiva:** il segnale ricevuto viene modificato in modo imprevedibile (fading/evanescenza)



Bisogna tenere conto delle proprietà fisiche del mezzo, quindi da come il segnale si modifica durante la propagazione. Il **Coherence time** permette di avere una stima per sapere ogni quanto campionare la condizione di un canale, un lasso di tempo in cui essere sicuri che il canale non subirà cambiamenti significativi; scala temporale in cui le caratteristiche del segnale sono “costanti”

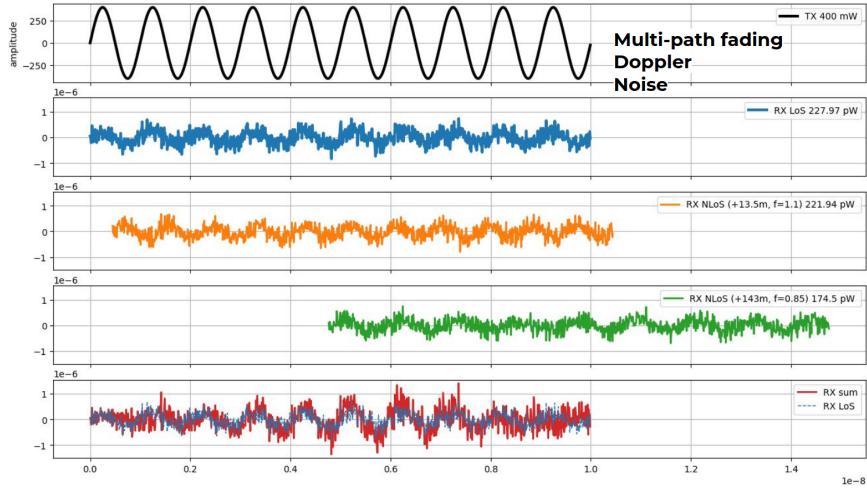
$$T_c = \frac{1}{f_D}$$

Dove  $f_D$  è la **frequenza di Doppler**: dipende dalla velocità di movimento tra trasmettitore e ricevitore e dalla frequenza (e velocità della luce); più alta è la frequenza o più velocemente ci muoviamo più il campionamento dovrà essere fitto

$$f_D = \frac{v}{c} f_c$$

Una tecnologia pensata per determinati spettri ed una determinata velocità relativa possiamo determinare una stima di  $T_c$  e costruire apparati che funzionano di conseguenza.

Esempio:

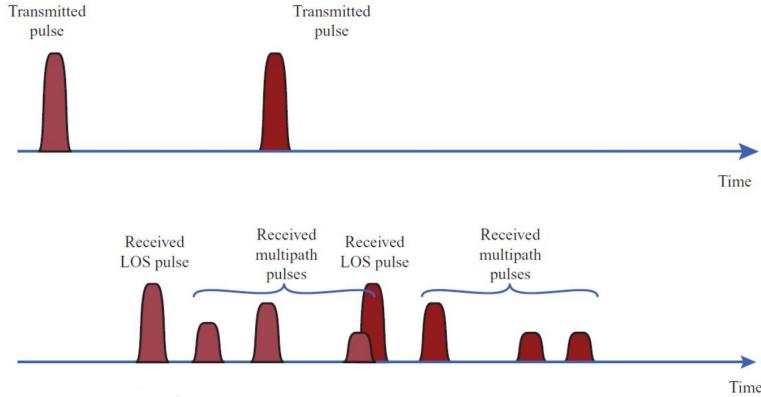


I primi 4 grafici sono:

- segnale trasmesso
- segnale ricevuto LoS, in  $pW$ , con effetto Doppler e noise
- altri 2 segnali NLoS (Non Line of Sight), percorrono più spazio, vengono ricevute dopo nel tempo e con una potenza ridotta, oltre che subire effetto Doppler e noise

L'ultima è ciò che viene effettivamente ricevuto, ovvero la combinazione di tutte le onde ricevute. E questo solo con il multipath, bisogna aggiungere anche effetto doppler e noise.

**Inter-Symbol Interference:** La lunghezza dei vari simboli deve tenere conto degli effetti del multipath.

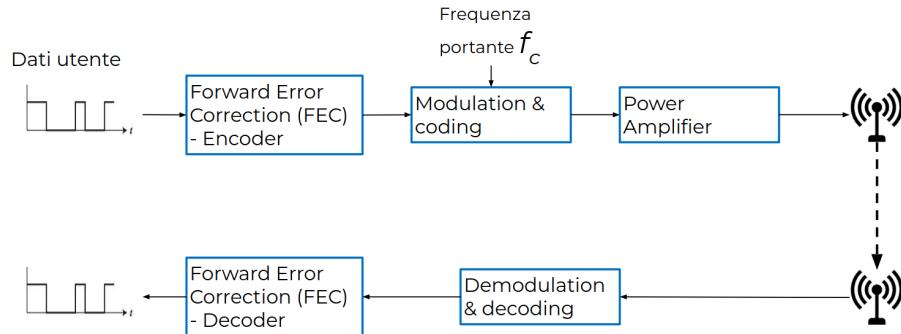


Dopo il primo impulso LoS, vengono ricevuti altri simboli per effetto del multipath e ci possono essere altri segnali che interferiscono con l'impulso LoS successivo. Si sovrappongono segnali LoS e multipath di quello precedente, causando interferenza distruttiva, interferenza inter-simbolo ISI. La distanza tra un segnale e l'altro è troppo breve.

Maggiore è la distanza tra TX e RX, più alta è la probabilità di questi fenomeni. Trasmettendo meno simboli ho un data rate minore, ma incrementandoli aumenta la probabilità di avere ISI.

## 2.3 Codifica e Trasmissione Dati

La struttura di una trasmissione radio è



Le fasi sono:

- Forward Error Correction FEC
- Modulazione e encoding
- Si amplifica la potenza in modo da renderla sufficiente (secondo il path loss previsto), deve essere in grado di raggiungere il RX
- il segnale “brutto” ricevuto va demodulato e decodificato
- FEC sul decoder

### 2.3.1 Modulazione e Codifica

Per la codifica possiamo agire sui 3 parametri di una sinusoidale:

- **Amplitude Shift Keying ASK:** diversi livelli di ampiezza per diversi bit
- **Frequency Shift Keying FSK:** diverse frequenze per diversi bit
- **Phase Shift Keying PSK:** diverse fasi per diversi bit

Ci possono anche essere combinazioni. L'obiettivo della parte di encoding e modulation è produrre una forma d'onda con un determinato significato.

**Differential Phase-Shift Keying (DPSK):** Risulta più semplice accorgersi della differenza rispetto che misurare un certo livello: la codifica dipende dallo stato precedente:

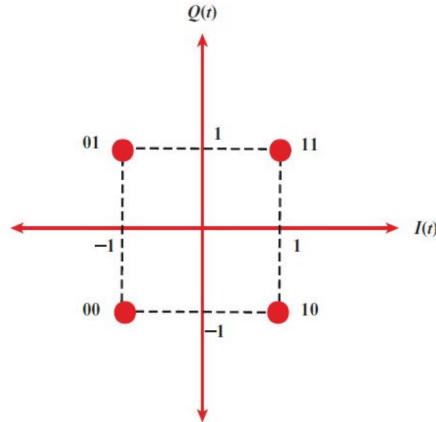
- 0 → nessun cambio di fase
- 1 → applico uno shift

Richiede un allineamento meno preciso tra TX e RX, identificare le differenze è più semplice. Codifica e decodifica non sono più fisse ma dipendono dalla variazione tra un bit ed il successivo (o mancanza di essa).

**Quadrature Phase-Shift Keying QPSK:** Viene utilizzata la fase per determinare i bit. Ci sono 4 fasi differenti ( $90^\circ$  tra una e l'altra), quindi 2 bit per simbolo

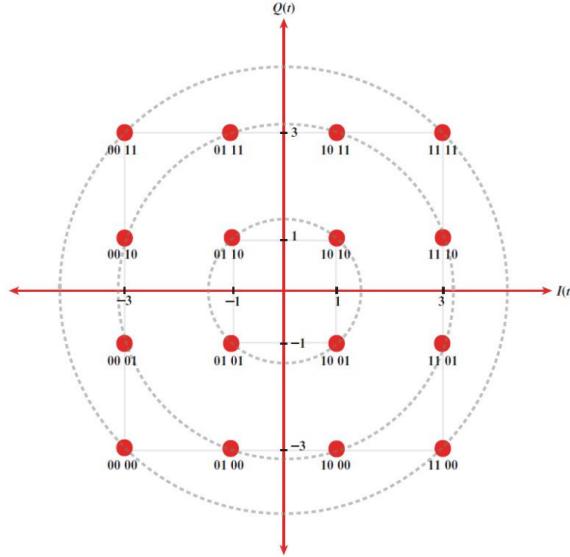
$$s(t) = \begin{cases} A \cos(2\pi f_{ct} t + \frac{\pi}{4}) & 11 \\ A \cos(2\pi f_{ct} t + \frac{3\pi}{4}) & 01 \\ A \cos(2\pi f_{ct} t - \frac{3\pi}{4}) & 00 \\ A \cos(2\pi f_{ct} t - \frac{\pi}{4}) & 10 \end{cases}$$

Si usa la codifica Gray, punti adiacenti differiscono di un solo bit. Diagramma della costellazione:

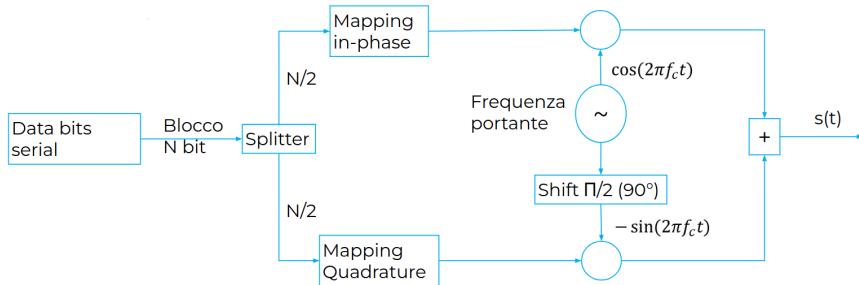


Si può vedere che ci sono 4 “posizioni”, ognuna delle quali codifica 2 bit. Si chiama “quadra” per la variazione di  $90^\circ$  tra uno e l'altro. Qua abbiamo 2 bit per symbol.

**Quadrature Amplitude Modulation QAM:** Oltre alla fase abbiamo anche una modifica dell'ampiezza. Utilizziamo due parametri per avere più codifiche differenti, portando ad una costellazione più densa. Combina variazioni di ampiezza e fase.



Schema QAM:



Si spezzano i bit, si mappa prima in fase e poi in ampiezza, si modula sulla portante e le sinusoidi risultanti si uniscono per avere il segnale modulato da trasmettere.

Wi-Fi 5 usa 256-QAM mentre Wi-Fi 6 usa 1024-QAM. La potenza rimane costante, ci muoviamo all'interno dello stesso spazio, quello che cambia è quanto "densi" sono i simboli all'interno dello spazio.

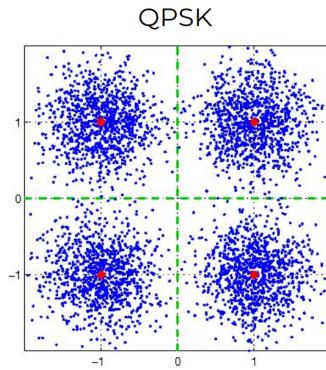
### 2.3.2 Bit Error Rate Curve

Rappresenta la **probabilità che un bit venga alterato**, in funzione del rapporto tra la densità di energia del segnale per bit ed il livello di rumore

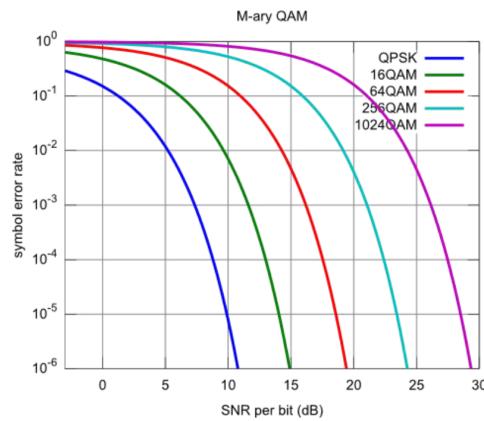
$$BER = func = \left( \frac{E_b}{N_0} \right)$$

Qual'è la probabilità di sbagliare un bit, dato un certo rapporto tra densità di energia del segnale per bit e rumore. Simile a SNR come idea.

Dei valori ricevuti da un trasmettitore potrebbero essere:



Il RX riceve una “nuvola” di punti e cerca il simbolo più vicino ma se l’alterazione subita è significativa potrebbe essere un’interpretazione sbagliata quindi un errore. Man mano che il segnale migliora rispetto al rumore, la probabilità di avere un errore diminuisce. Tanto più densa è la costellazione tanto è più probabile sbagliare simbolo:



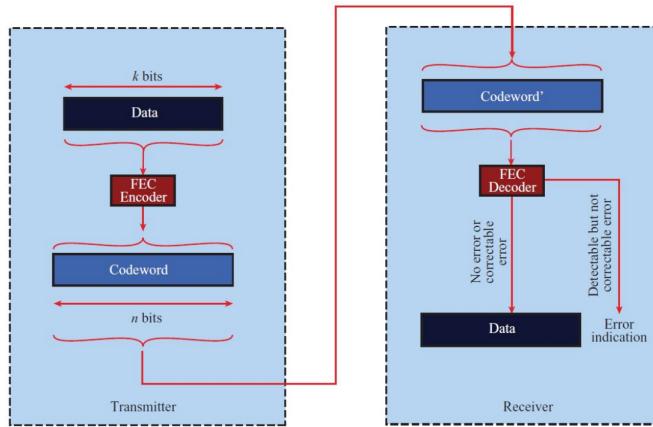
Si può vedere come QPSK sbagli molto meno rispetto a 1024-QAM.

**Adaptive Modulation and Coding AMC:** Misuriamo la qualità del canale e cerchiamo di capire quale sia la codifica più efficiente, troppi errori renderebbero la comunicazione effettivamente inutile. Se il SNR è 10 è inutile trasmettere con 1024-QAM in quanto quasi tutti i simboli saranno sbagliati (immagine precedente), è meglio inviare con un data rate minore che permette meno errori.

La **codifica viene modificata in base alla qualità del canale**, ogni tot di tempo misuro la qualità del canale ed adatto il data rate di conseguenza.

### 2.3.3 Forward Error Correction

Servono delle tecniche di correzione dell'errore. Dato che nel wireless la probabilità di errore è elevata, viene aggiunta una ridondanza ai dati inviati: ogni sequenza di bit diventa una codeword, abbiamo  $k$  bit trasmessi per  $n$  bit da inviare (sempre  $k > n$ ). In questo modo, con una tabella di codeword adatta, è possibile rendere la trasmissione resiliente agli errori.



Ogni  $n$  bit da trasmettere diventano  $k$ , secondo una tabella prefissata, così anche se qualcuno arriva sbagliato si può ricostruire il valore originale (esempio:  $00 \rightarrow 00000$ ,  $10 \rightarrow 11001$ ).

Rimangono valide le tecniche di error detection, in cui a livello di RX riusciamo a capire se c'è stato un errore tramite dei bit di controllo aggiunti ai dati (e.g., CNC).

**AMC:** A seconda delle condizioni del canale wireless il trasmittente sceglie lo schema di modulazione e codifica opportuno. Bisogna scegliere: codifica, coding rate (redundancy). Usato in reti mobili 4G, 5G e WiFi (802.11n+).

Generalmente, queste informazioni vengono scelte a partire da una tabella: se ho un SNR di questo tipo, che modulazione e coding rate (rapporto bit totali e di informazione effettiva) posso usare?

| SNR<br>(dB) | MCS<br>index <sup>[i]</sup> | Modulation<br>type | Coding<br>rate | Modulation and coding schemes      |           |                 |           |                 |           |                  |           |
|-------------|-----------------------------|--------------------|----------------|------------------------------------|-----------|-----------------|-----------|-----------------|-----------|------------------|-----------|
|             |                             |                    |                | Data rate (Mbit/s) <sup>[ii]</sup> |           |                 |           |                 |           |                  |           |
|             |                             |                    |                | 20 MHz channels                    |           | 40 MHz channels |           | 80 MHz channels |           | 160 MHz channels |           |
| 5 dB        | 0                           | BPSK               | 1/2            | 1600 ns GI <sup>[iii]</sup>        | 800 ns GI | 1600 ns GI      | 800 ns GI | 1600 ns GI      | 800 ns GI | 1600 ns GI       | 800 ns GI |
|             |                             |                    |                | 8                                  | 8.6       | 16              | 17.2      | 34              | 36.0      | 68               | 72        |
| 10 dB       | 1                           | QPSK               | 1/2            | 16                                 | 17.2      | 33              | 34.4      | 68              | 72.1      | 136              | 144       |
|             |                             |                    |                | 24                                 | 25.8      | 49              | 51.6      | 102             | 108.1     | 204              | 216       |
| 15 dB       | 2                           | QPSK               | 3/4            | 33                                 | 34.4      | 65              | 68.8      | 136             | 144.1     | 272              | 282       |
|             |                             |                    |                | 49                                 | 51.6      | 98              | 103.2     | 204             | 216.2     | 408              | 432       |
| 20 dB       | 3                           | 16-QAM             | 1/2            | 65                                 | 68.8      | 130             | 137.6     | 272             | 288.2     | 544              | 576       |
|             |                             |                    |                | 73                                 | 77.4      | 146             | 154.9     | 306             | 324.4     | 613              | 649       |
| 25 dB       | 4                           | 16-QAM             | 3/4            | 81                                 | 86.0      | 163             | 172.1     | 340             | 360.3     | 681              | 721       |
|             |                             |                    |                | 98                                 | 103.2     | 195             | 206.5     | 408             | 432.4     | 817              | 865       |
| 30 dB       | 5                           | 64-QAM             | 5/6            | 108                                | 114.7     | 217             | 229.4     | 453             | 480.4     | 907              | 961       |
|             |                             |                    |                | 122                                | 129.0     | 244             | 258.1     | 510             | 540.4     | 1021             | 1081      |
|             | 6                           | 64-QAM             | 3/4            | 135                                | 143.4     | 271             | 286.8     | 567             | 600.5     | 1134             | 1201      |
|             |                             |                    |                |                                    |           |                 |           |                 |           |                  |           |

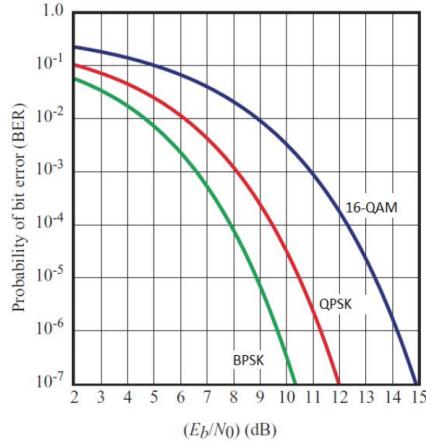
**Esempio:** Dati un  $SNR$  sul canale, un  $BER$  che si vuole garantire ed un symbol rate

$$SNR = 8dB, BER = 10^{-2}, SR = 1000sym/s$$

Che modulation and coding scheme posso usare sul mio canale per avere l'error rate desiderato, sapendo che il coding rate per ogni codifica è

| SNR      | Coding rate |      |        |
|----------|-------------|------|--------|
|          | BPSK        | QPSK | 16-QAM |
| < 6dB    | 0.6         | 0.4  | 0.2    |
| 6 – 10dB | 0.8         | 0.6  | 0.5    |
| > 10dB   | 0.9         | 0.8  | 0.7    |

Ed il BER per ogni codifica è



Dall'immagine si può vedere che con  $SNR = 8dB$ , sia BPSK che QPSK permettono un  $BER$  adeguato. Quale dei due però permette il data rate migliore? Possiamo calcolare il data rate come

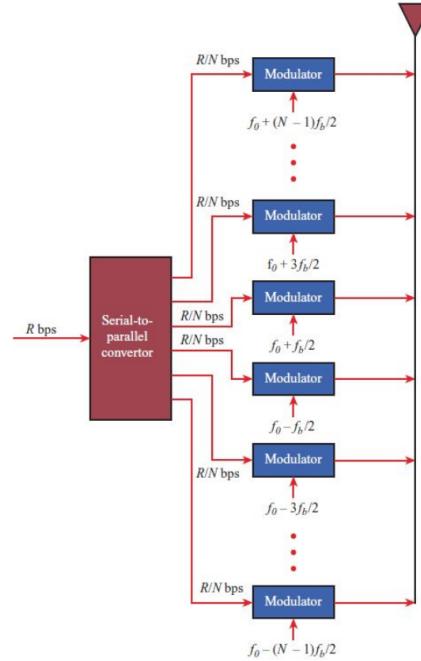
$$sym/s \cdot bit/sym \cdot coding\_rate$$

Quindi

- BPSK: con coding rate di 0.8 ed 1bps:  $1000 \cdot 1 \cdot 0.8 = 800bit/s$
- QPSK: con coding rate di 0.6 e 2bps:  $1000 \cdot 2 \cdot 0.6 = 1200bit/s$

### 2.3.4 Orthogonal Frequency Division Multiplexing OFDM

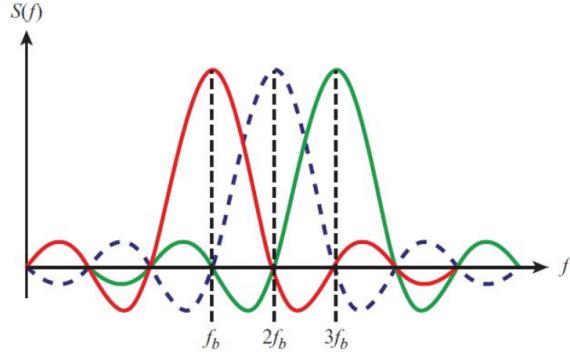
Vogliamo offrire canali differenti usando una divisione in frequenza diversa da FDM. L'obiettivo è quello di garantire lo stesso data rate che avremmo in una trasmissione in TDM ma usando una divisione in frequenza, senza usare troppa banda. Schema:



Lo si implementa tramite uno split da seriale a parallelo ed ogni componente viene inviata ad un modulatore, ognuno dei quali modula rispetto a multipli di una certa frequenza base  $f_b$  e alla portante  $f_0$ ; quindi abbiamo una portante  $f_0$  e tante sotto-portanti multiple di  $f_b$ . Stessa codifica e modulazione. Alla fine si trasmette la combinazione di tutte le onde create.

Nel caso di FDM classico viene lasciata una guardia per evitare interferenze, si allontanano un po' le frequenze in modo tale che le varie armoniche di una frequenza non possa interagire con le adiacenti.

**Ortogonalità:** Nell'OFDM le **subcarrier sono ortogonali tra loro** e non interferiscono, i.e., la distanza tra subcarrier è studiata in modo da evitare interferenze. Nel momento di picco di ogni sotto-portante, il contributo delle altre frequenze è nullo.



La scelta di  $f_b$  dipende dalla durata dei simboli  $T$ :

$$f_b = \frac{1}{T}$$

Tutti i segnali multipli di  $f_b$  sono ortogonali tra loro. Si tratta dell'inverso della durata del simbolo.

Considerazioni:

- più robusto riguardo ad interferenze che riguardano solo alcuni subcarrier
- più robusto rispetto ai problemi di multipath perché la distanza tra un simbolo e l'altro è maggiore (ISI ridotta)

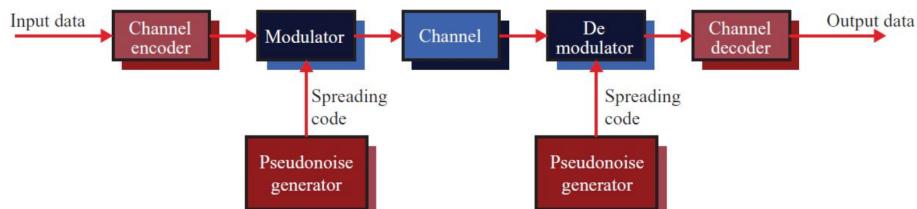
N.B.: in generale multiple access  $\neq$  multiplexing, far passare più segnali contemporaneamente è diverso da fare comunicare più utenti contemporaneamente.

**TL;DR:** Si tratta di un FDM in cui le frequenze sono ortogonali tra loro in modo da non avere interferenze. Le interferenze in FDM erano prevenute tramite l'uso di una guardia, ovvero spazio di banda libero, OFDM permette di usare meno banda.

## 2.4 Spread Spectrum

Letteralmente “spettro espanso”, consiste nel trasmettere il segnale di informazione su uno **spettro di frequenze più ampio** di quella del segnale. Una banda maggiore del necessario.

La struttura, concettualmente, è



Dopo l'encoding, c'è una fase di **spreading dei dati**, generalmente basato su valori pseudo-casuali (da invertire dopo la ricezione).

Motivazioni:

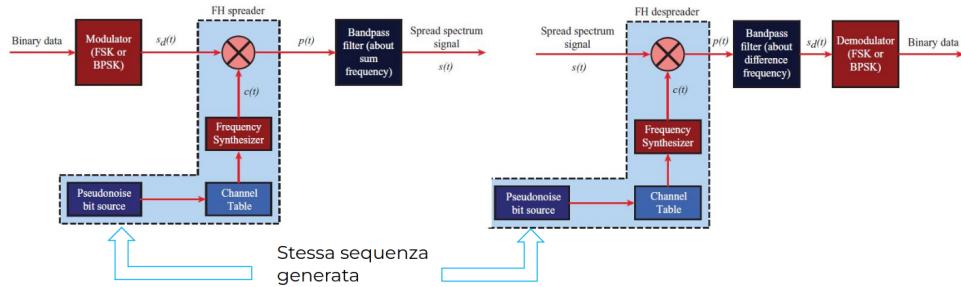
- rendere il segnale più robusto, “spalmandolo” su una banda più ampia lo rende più resiliente a diversi tipi di rumore e interferenze
- nascondere e cifrare il segnale (nato in ambito militare); solo TX e RX sono a conoscenza del codice di spreading (non una tecnica sufficiente, ma a livello radio questo si può fare)
- molti utenti possono usare indipendentemente la stessa banda contemporaneamente senza interferenza. Usata da CDMA

### 2.4.1 Frequency Hopping Spread Spectrum FHSS

In FHSS il codice di spreading determina quale frequenza usare per trasmettere il segnale. Ad ogni intervallo di tempo prestabilito, la frequenza viene cambiata pseudo-casualmente (frequency hopping). Usato da Bluetooth (802.15.1), “saltando” ogni  $625\mu s$ .

In altre parole: ho una certa ampiezza di banda e cambio in modo pseudo-casuale la frequenza usata ad ogni intervallo di tempo. Ho  $n$  frequenze e ne uso una per volta, cambiata ogni tanto.

Schema di trasmissione e ricezione:



La channel table permette di scegliere la frequenza da utilizzare, in base al valore casuale, ed il sintetizzatore utilizzerà quella frequenza. Il tempo di hopping generalmente è predefinito, serve un seed per la sequenza di valori pseudo-casuali usati per determinare la frequenza usata.

Il frequency hopping è più resistente a rumore e jamming, in quanto compromettere una frequenza non compromette l'intera trasmissione.

Un altro ricevitore che si sincronizza con il trasmettitore può solo leggere alcuni pezzi dei messaggi perché non conosce la sequenza di hopping.

### 2.4.2 Direct Sequence Spread Spectrum DS-SS

Per una sequenza di  $D$  bit, ogni bit della sequenza viene rappresentato da un insieme di bit usando un codice di spread (pseudo-casuale). Ogni bit diventa  $n$  bit ottenuti da una sequenza casuale.

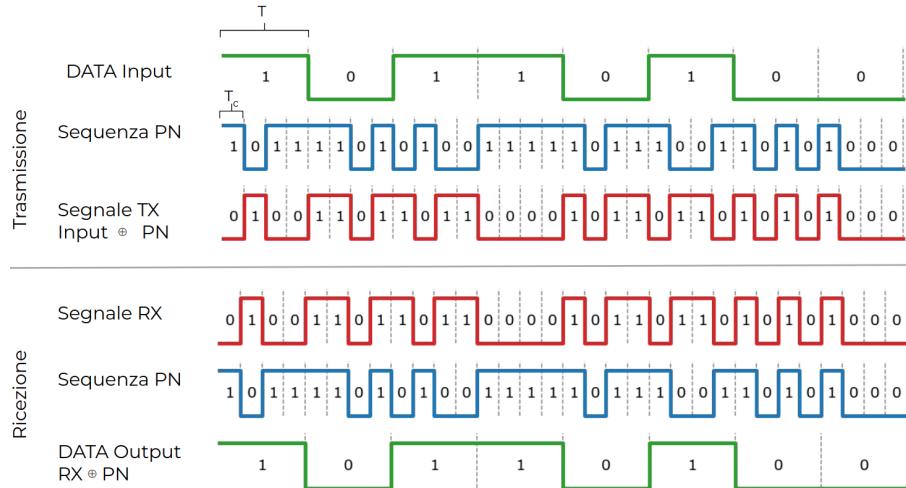
I bit della sequenza di spread “durano meno” (durano  $1/n$  del tempo dei bit di informazione) e vengono chiamati “chip”. Per ogni bit originale abbiamo  $n$  chip.

Per mantenere lo stesso data rate abbiamo bisogno di  $n$  volte la banda ( $n$  fattore di spreading).

Generalmente, si usa uno **xor** (in quanto invertibile) degli  $n$  bit casuali con il bit da inviare e si invia il risultato. Esempio:

| Bit informazione | 1 |   |   |   | 0 |   |   |   |
|------------------|---|---|---|---|---|---|---|---|
| Sequenza di chip | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| Trasmesso (XOR)  | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

Usando BPSK diventa:



In verde il **segnale originale**, in blu la **sequenza pseudo-casuale**, in rosso ciò che viene **effettivamente inviato**.

### 2.4.3 Code Division Multiple Access CDMA

Differenza tra multiplexing e multiple access:

- multiplexing significa avere più canali di comunicazione
- multiple access: come faccio a far comunicare più utenti

L'idea è utilizzare un canale con una certa banda a disposizione, e tutti i dispositivi possono usare la stessa banda, evitando interferenze.

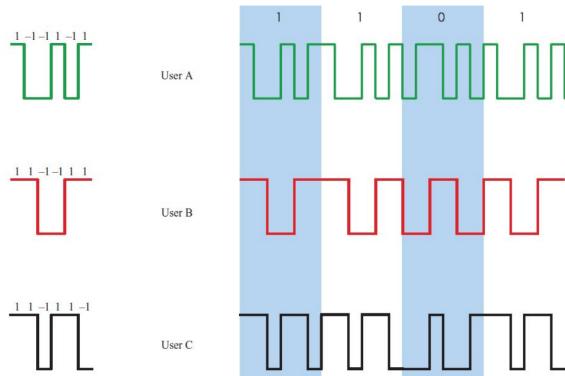
Ogni bit della sequenza da trasmettere viene codificato con un codice largo  $k \geq 1$  chip usando un pattern prefissato detto **codice**. Un chip è una sequenza di 1 e -1 (al posto dello 0). Ogni utente ha un codice diverso e di conseguenza produrrà chip diversi. Tutti “parlano assieme” e posso comunque distinguere le informazioni provenienti da un utente specifico.

**Sequenze Walsh:** Creano un insieme di codici ortogonali (sono in numero limitato).

**Sequenze PN, Gold, Kasami:** Non ortogonali ma in numero molto maggiore.

Non si ha nessuna divisione della banda, ma uno spettro maggiore che quello per un singolo bit. Ognuno comincia a trasmettere con il proprio codice assegnato senza preoccuparsi di interferenze, starà al ricevitore distinguere i vari segnali. Tutte le forme d'onda vengono trasmesse assieme, senza il problema dell'interferenza.

Esempio di trasmissione da parte di 3 utenti:



Il RX è a conoscenza del codice, numero di chip per bit e la regola (1 viene codificato tramite il codice stesso, 0 tramite l'inverso). Il RX calcola la funzione

$$s_u(d) = \sum_{i=1}^k d_i \cdot c_i$$

Si tratta di una moltiplicazione tra il valore ricevuto ed il relativo bit all'interno del codice: se il risultato della sommatoria è positivo è stato trasmesso un 1, 0 altrimenti.

Esempio: con 3 utenti

|        |   |    |    |    |    |    |  |
|--------|---|----|----|----|----|----|--|
| User A | 1 | -1 | -1 | 1  | -1 | 1  |  |
| User B | 1 | 1  | -1 | -1 | 1  | 1  |  |
| User C | 1 | 1  | -1 | 1  | 1  | -1 |  |

Con un solo segnale, se l'utente A invia un 1:

|                           |   |    |    |   |    |   |    |
|---------------------------|---|----|----|---|----|---|----|
| Transmit (data bit = 1) A | 1 | -1 | -1 | 1 | -1 | 1 |    |
| Receiver codeword A       | 1 | -1 | -1 | 1 | -1 | 1 |    |
| Multiplication            | 1 | 1  | 1  | 1 | 1  | 1 | =6 |

Per uno 0, il codice trasmesso sarebbe invertito ed il risultato della sommatoria diventa  $-6$ . In questo caso l'utente A ha trasmesso ed è stato usato il codice di A per decodificare. Se provassi ad usare il codice di A per decodificare un messaggio di B:

|                           |   |    |    |    |    |   |    |
|---------------------------|---|----|----|----|----|---|----|
| Transmit (data bit = 1) B | 1 | 1  | -1 | -1 | 1  | 1 |    |
| Receiver codeword A       | 1 | -1 | -1 | 1  | -1 | 1 |    |
| Multiplication            | 1 | -1 | 1  | -1 | -1 | 1 | =0 |

Il risultato è 0, quindi non è stato trasmesso da A ma da un segnale ortogonale rispetto ad A. Non è detto che i segnali debbano essere perfettamente ortogonali, possono anche esserci interferenze:

|                           |   |   |    |    |   |    |    |
|---------------------------|---|---|----|----|---|----|----|
| Transmit (data bit = 1) C | 1 | 1 | -1 | 1  | 1 | -1 |    |
| Receiver codeword A       | 1 | 1 | -1 | -1 | 1 | 1  |    |
| Multiplication            | 1 | 1 | 1  | -1 | 1 | -1 | =2 |

Non sono perfettamente ortogonali quindi risulta in una interferenza, il risultato è però troppo basso per essere interpretato come un valore corretto.

Ma l'idea è che tutti trasmettano assieme, quindi i segnali possono combinarsi:

|                     |   |   |    |    |   |    |    |
|---------------------|---|---|----|----|---|----|----|
| B (data bit = 1)    | 1 | 1 | -1 | -1 | 1 | 1  |    |
| C (data bit = 1)    | 1 | 1 | -1 | 1  | 1 | -1 |    |
| Combined signal     | 2 | 2 | -2 | 0  | 2 | 0  |    |
| Receiver codeword B | 1 | 1 | -1 | -1 | 1 | 1  |    |
| Multiplication      | 2 | 2 | 2  | 0  | 2 | 0  | =8 |

Cercando di estrarre cosa ha mandato B (ovvero usando il codice B) risulta un segnale abbastanza alto da poter essere interpretato come 1.

Più è alto il numero di utenti più lo spreading factor deve essere alto (lunghezza del codice), riducendo il data rate del dispositivo. Può essere modulato in funzione del numero di utenti, quando ci sono tanti utenti connessi viene usato uno spreading factor maggiore.

**Near-Far Problem:** Questo sistema funziona bene se i vari segnali ricevuti hanno più o meno la stessa potenza, ma utenti più lontani avranno una potenza minore (se tutti trasmettono con la stessa potenza).

La soluzione è utilizzare potenze diverse in base alla distanza (sì, consuma più batteria).

## 3 Wireless Personal Area Network WPAN

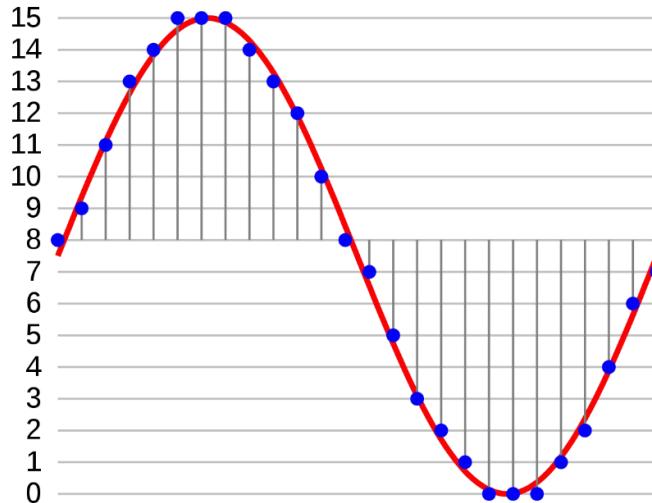
### 3.1 ISM Band

Ci sono **porzioni di banda unlicensed**, ovvero che non necessitano di licenza per essere utilizzate. Sono riservate per usi industriali, scientifici e medici (ISM).

Bluetooth, Wi-Fi, apparecchi medici e tutti i dispositivi wireless comunicano sulle stesse frequenze. Essendo condiviso, le varie tecnologie devono tollerare interferenze.

### 3.2 Pulse Code Modulation PCM

Si tratta di una codifica lossless per onde. Partendo da un segnale continuo, l'ampiezza della curva viene quantizzata in livelli (intervalli) e viene effettuato un campionamento. La frequenza di campionamento è il doppio della frequenza massima da campionare (Teorema del campionamento di Shannon).



Per la voce telefonica: PCM 8bit  $8000Hz$ , quindi  $64kbps$  (frequenza della voce  $300 - 3400Hz$ ). La musica viene campionata a 24bit  $41/48kHz$ .

### 3.3 802.15.x

Lo standard 802.15 comprende un insieme di tecnologie per la **comunicazione a corto raggio** (<https://www.ieee802.org/15/>).

Esempi di tecnologie:

- 802.15.1 Bluetooth
- 802.15.2 High-rate WPAN
- 802.15.4 Low-rate WPAN (es. Zigbee)
- 802.15.5 Mesh Networking (combinazione di High-rate e Low-Rate)
- 802.15.6 Body Area Network (BAN) pensato ad esempio per applicazioni in ambito medico
- 802.15.7 Visible Light Communication (VLC) (es. Vehicle-to-vehicle communication & Li-Fi)

### 3.4 Bluetooth

Si tratta dello standard 812.15.1. Si compone di reti chiamate piconet ed all'interno della rete si ha un dispositivo **master** e **uno o più slave** sotto il controllo del master, che controlla la piconet.

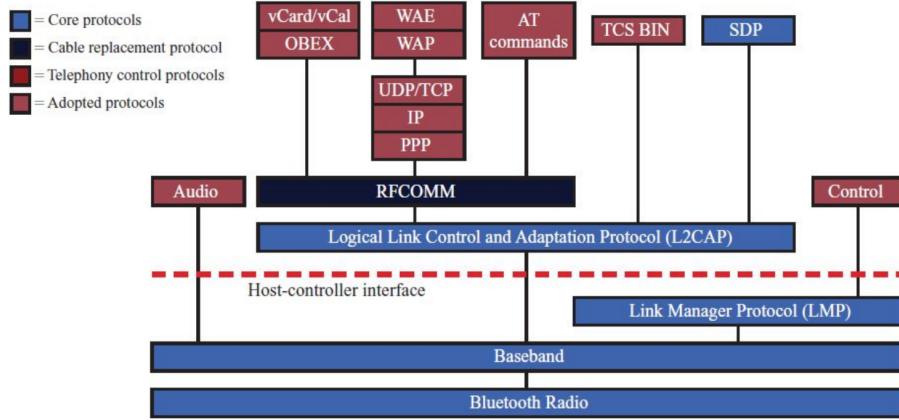
**Caratteristiche:**

- Short range (10-50m nei casi d'uso tipici a seconda della classe di potenza del dispositivo, **Bluetooth range estimator**)
- Usa la banda ISM **2.4GHz**
- Data Rate: **2.1Mbps – 24Mbps**

Utilizzi:

- punto di accesso per dati e voce
- sostituzione di cavi (periferiche wireless)
- comunicazione ad hoc con altri dispositivi Bluetooth

### 3.4.1 Architettura dei protocolli



In blu sono i “core protocols”, presenti in tutti i dispositivi Bluetooth. Gli altri (rossi e blu scuro) sono implementati solo se il dispositivo ne necessita, in base all’uso.

**Bluetooth Radio:** La parte di livello fisico, specifica l’interfaccia radio:

- radio frequenze, quali frequenze utilizzare
- gestisce il frequency hopping
- decide lo schema di modulazioni in base al canale
- determina la potenza di trasmissione

**Baseband:** Il livello di Baseband si occupa di

- stabilire la connessione con la piconet
- gestire l’indirizzamento
- formattazione dei pacchetti (frame)
- gestire le tempistiche di comunicazione (Time Division Duplex TDD e Time Division Multiple Access TDMA)
- gestisce la potenza di trasmissione (passa le indicazioni a livello radio)

**Link Manager Protocol LMP:** Fa da “manager” del collegamento. Si occupa di:

- configurare i collegamenti tra dispositivi
- gestione di collegamenti attivi
- funzionalità di sicurezza e cifratura

Si tratta di un protocollo di controllo, non passano dati ma gestisce il collegamento per i livelli sottostanti.

**Logical Link Control and Adaptation Protocol (L2CAP):** I livelli precedenti erano implementati sul chip Bluetooth, da qui in su vengono implementati a livello software. Si occupa di:

- adatta i protocolli di livello superiore al livello baseband
- astrarre tutto ciò che c’è sotto per i servizi a livelli superiori, *Connectionless* e *Connection-oriented*

**Service Discovery Protocol SDP:** Gestisce le informazioni del dispositivo. Permette di comunicare

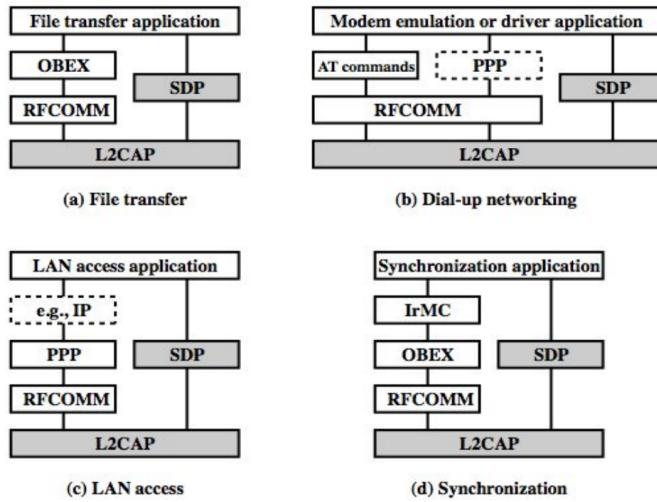
- servizi disponibili sul dispositivo
- caratteristiche sul dispositivo

Stile architettura client-server: prevede interrogazioni richiesta-risposta per stabilire connessioni tra dispositivi Bluetooth.

**Radio Frequency Communication RFCOMM:** Astra il livello di Bluetooth, permette di “astrarre un cavo”. Simula una comunicazione seriale e permette la trasmissione di dati tra dispositivi Bluetooth.

**Livelli superiori:** I livelli in rosso sono protocolli “già esistenti”, ciascun dispositivo può avere parte di questi protocolli in base all’uso, i livelli inferiori permettono la comunicazione a livello fisico.

**Profilo:** Per avere determinate funzionalità un dispositivo deve seguire dei “profili”. Esempi di profili:

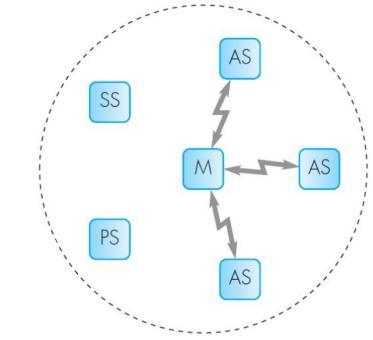


Questi sono standard, un dispositivo deve aderire ad uno o più profili (annunciati dal SDP) per avere il relativo utilizzo.

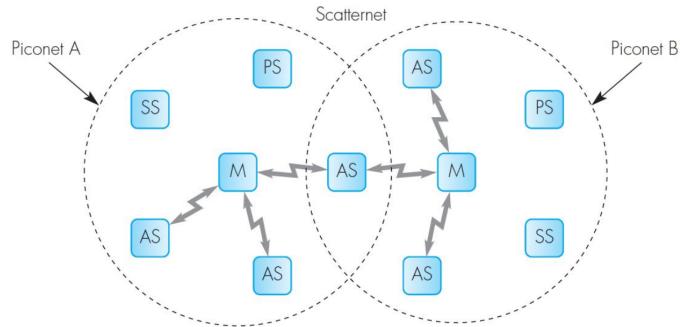
### 3.4.2 Piconet & Scatternet

**Piconet:** Una piconet è composta da un **master** e

- **Active Slave (AS):** membro attivo delle piconet, con un indirizzo Active Member Address (AMA) su 3 bit assegnato dal master (0 è il master, quindi 7 dispositivi al massimo)
- **Parked Slave (PS):** membro della piconet ma temporaneamente disattivato, non possono comunicare attivamente ma solo ogni tanto tramite il Parked Member Address (PMA), 8 bit (0 è il master); un parked può tornare attivo solo se “c’è spazio”
- **Standby Slave (SS):** non sconosciuti ma scollegati; non hanno indirizzi e possono quindi essere infiniti



**Scatternet:** La piconet ha al centro sempre un master, ma un dispositivo può appartenere a più piconet, portando ad una scatternet



In qualsiasi caso i master lavorano in maniera completamente **separata**. Un AS attivo in due piconet diverse avrà indirizzamento diverso sulle due reti.

### 3.4.3 Bluetooth Radio

Specifiche radio Bluetooth 2.1:

|                           | Basic Rate (BR)                              | Enhanced Data Rate (EDR) |
|---------------------------|--|--------------------------|
| <b>Topology</b>           | Up to 7 simultaneous links in a logical star |                          |
| <b>Modulation</b>         | GFSK   | $\pi/4$ -DQPSK and 8DPSK |
| <b>Peak data rate</b>     | 1Mbps  | 2Mbps and 3Mbps          |
| <b>RF bandwidth</b>       | 220kHz( $-3dB$ ), 1MHz( $-20dB$ )            |                          |
| <b>RF band</b>            |  | 2.4GHz, ISM band         |
| <b>RF carriers</b>        |  | 23/79                    |
| <b>Carrier spacing</b>    |  | 1MHz                     |
| <b>Transmit power</b>     |  | 0.1W                     |
| <b>Piconet access</b>     |  | FH-TDD-TDMA              |
| <b>Frequency hop rate</b> |  | 1600 hops/s              |
| <b>Scatternet access</b>  |  | FH-CDMA                  |

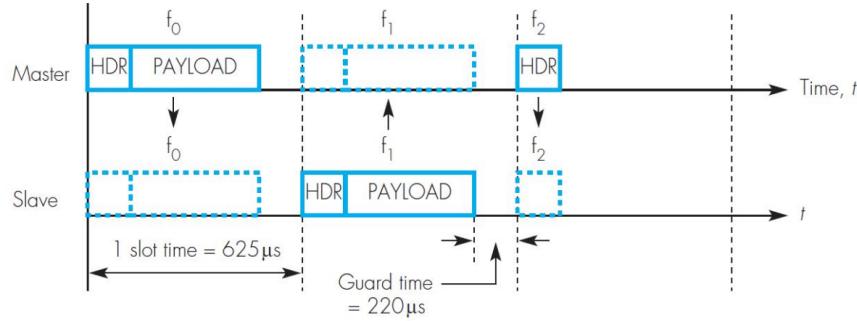
Per gestire le scatternet si usa FH-CDMA, anche in caso due scatternet comunichino sulla stessa frequenza si usa CDMA per poter distinguere i segnali.

**Classi di potenza:** I dispositivi Bluetooth si differenziano in base alla classe di potenza

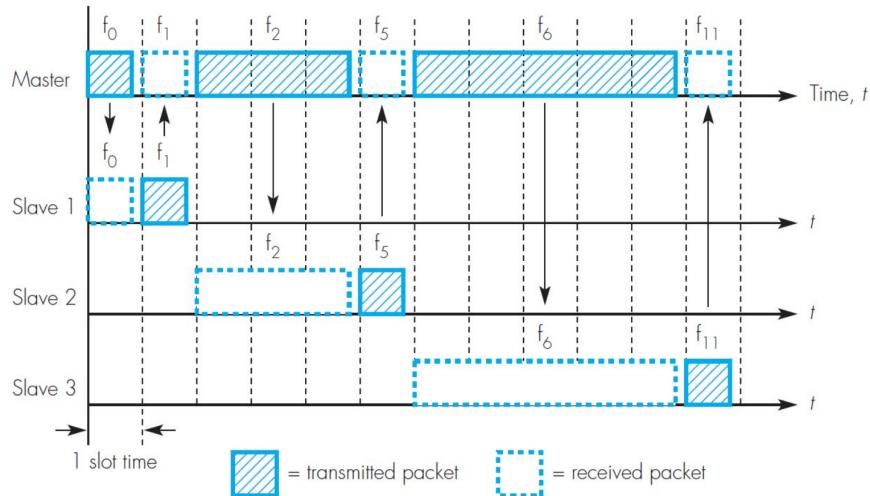
- Power class 1: 100mW 100 metri (senza ostacoli)
- Power class 2: 2.5mW 10 metri (più tipico per BT)
- Power class 3: 1mW 1-2 metri

**Comunicazione nelle piconet:** All'interno di una piconet vengono usate

- Frequency Hopping FH: la frequenza è decisa dal master e condivisa agli slave nella piconet
- Time Division Duplex TDD: la comunicazione tra master e slave è gestita in tempo e alternata, prima comunica il master con lo slave, poi si invertono, con slot di  $625\mu s$  (inclusa una guardia di  $220\mu s$ )
- Time Division Multiple Access TDMA: per gestire più dispositivi nello stesso momento



Esempio a più dispositivi:



Nelle frequenze pari master a slave, nelle frequenze dispari viceversa. Più slave vengono coordinati tramite TDMA, ovvero il master decide chi parla in quale slot di tempo. La **dimensione dei messaggi** può essere di 1, 3

o 5 slot di tempo consecutivi (dispari per mantenere l’alternanza in TDD). Nell’header di ogni pacchetto c’è la dimensione.

La frequenza usata dal frequency hopping è data dal numero di timeslot passati, non dal numero di messaggi, quindi al time slot 5 viene usata la quinta frequenza  $f_5$ , anche se sono stati inviati solo 3 messaggi. Per una singola trasmissione (messaggio) viene mantenuta la stessa frequenza (non cambia a metà). All’interno della piconet tutti i clock devono essere sincronizzati.

**Scatternet:** Nel caso di una scatternet, ogni piconet che la compone ha tutto diverso: diverse frequenze, diverso clock e di conseguenza completa autonomia. Un AS connesso a più piconet deve essere in grado di gestire le due connessioni indipendentemente.

Su 79 canali, cambiati ogni  $625\mu s$ , può capitare una sovrapposizione (quindi interferenza), possibili soluzioni sono:

- FH su un numero ridotto di canali, e.g., ogni piconet su 20 canali e si spera non ci sia sovrapposizione in quei 20
- Si usa CDMA per evitare interferenze; il master comunica un codice ortogonale ai dispositivi all’interno della piconet (soluzione usata)

### 3.4.4 Baseband

**Tipologie di servizio:** Offre due possibili canali logici:

- **Synchronous Connection-Oriented Link (SCO)** point-to-point:

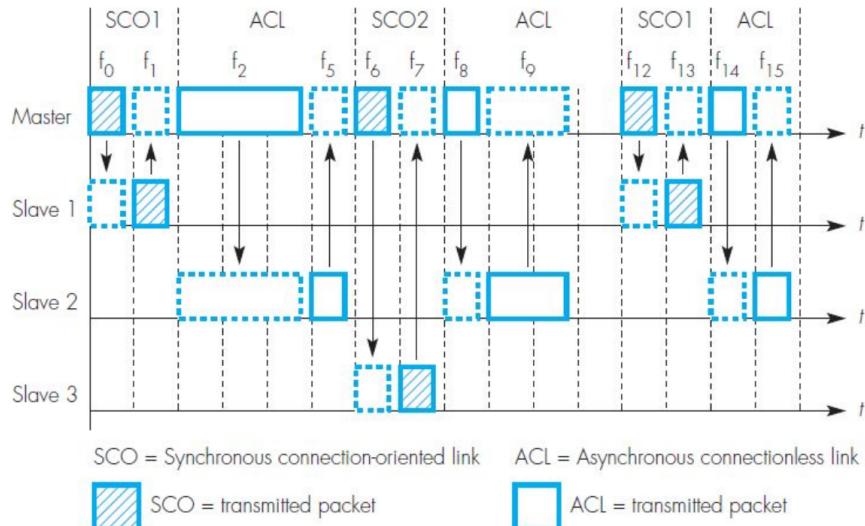
- canale audio/voce di  $64\text{ kbps}$  bidirezionale
- il master riserva una coppia di slot adiacenti ad intervalli regolari
- previsti fino al massimo di 3 canali SCO attivi contemporaneamente
- traffico real time

Garantiscono un bit rate fisso, un canale riservato, usato per casistiche sensibili al delay (real time).

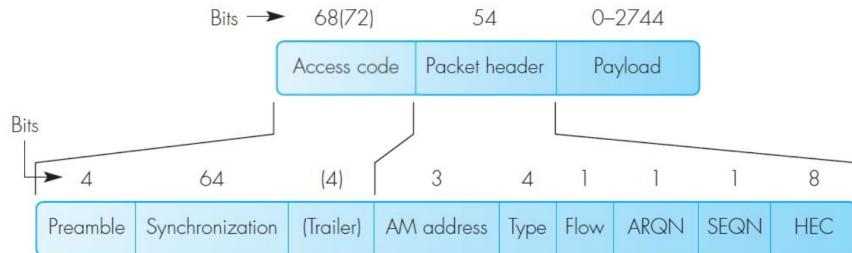
- **Asynchronous Connectionless Link (ACL)** point-to-multipoint:

- canali ACL occupano gli slot rimanenti
- traffico dati con ciascuno degli slave
- un solo ACL contemporaneo fra master e uno slave
- traffico best effort (nessuna garanzia di delay)

Bit rate non costante, permette una qualità maggiore ma senza nessuna garanzia.



### Formato pacchetti:



Access code: utilizzato per sincronizzazione e identificazione. Può essere di 3 tipi:

- Channel Access Code (CAC): identifica la piconet (derivato dai 48 bit dell'indirizzo hardware del master)
- Device Address Code (DAC): derivato dall'indirizzo hardware dello slave ed è usato dal master per chiamare il dispositivo (paging)
- Inquiry Address Code (IAC): usato per trovare l'indirizzo di un dispositivo vicino (durante la fase di inquiry)

Packet Header: ha diversi campi:

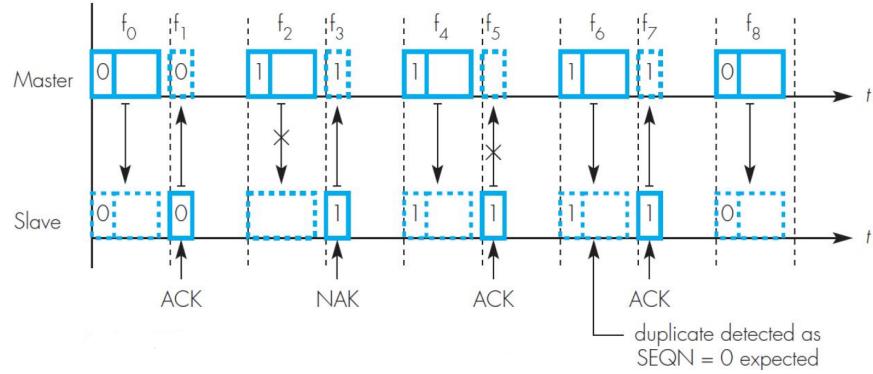
| Campo Header |   |
|--------------|---|
| AMA          | Indirizzo del membro attivo della piconet   |
| Type         | Identifica la tipologia del pacchetto e il formato del pacchetto SCO/ACL, numero di slot ecc... |
| Flow         | Per gli ACL: stop (1), resume (0)   |
| ARQN         | 1 ACK, 0 NACK   |
| SEQN         | Sequence number modulo 2  |
| HEC          | Controllo errori del campo header (1/3 FEC)   |

ARNQ e SEQN sono 2 bit necessari e sufficienti per il controllo degli errori (bastano grazie alla rigida struttura di comunicazione delineata al livello precedente).

Payload: contenuto effettivo del pacchetto può essere:

- SCO: 30 byte, (FEC 0, 2/3, 1/3), che porta a massimo  $64\text{ kbps}$ , dato che  $(30 \cdot 8) \cdot (1600/6)$ , ovvero  $30 \cdot 8$  bit,
- ACL: variabile da 0 a 343 byte

### Controllo degli errori:



All'interno del quadrato blu si vede il SEQN, mentre ACK/NACK sono indicati al di fuori.

Sequenza tipica:

- Il master invia un pacchetto con un SEQN  $s$
- lo slave risponde con un ACK ed un SEQN  $s$  (lo stesso)

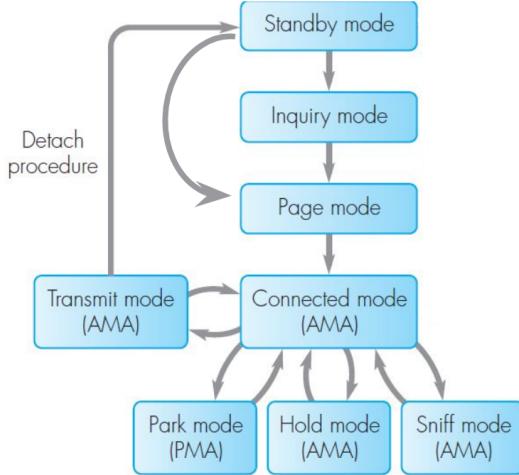
Possibili problemi:

- Lo slave non riceve il pacchetto: invierà un NACK per indicare la mancata ricezione, nello slot successivo deve parlare lo stesso dato che “è il suo turno”, sa che ci sarà un SEQN pari a  $\neg s$  ma non ha ricevuto nulla e lo indica con il NACK. “Non ho ricevuto e mi aspettavo questo”
- Se il master non riceve risposta dallo slave (ACK perso), il pacchetto viene re-inviato, con la possibilità di inviare un duplicato, ma meglio che perdere pacchetti
- Lo slave riceve un duplicato e se ne accorge in quanto il SEQN è uguale a quello precedente (il master non ha ricevuto l'ACK, non è cambiato il SEQN, quindi è una ritrasmissione del pacchetto precedente)

La rigidità dell'alternanza tra master e slave permette di rendere minimale lo spazio necessario per effettuare il controllo degli errori.

### 3.4.5 Link Manager Protocol

Transizioni di stato:



**Standby Mode:** A dispositivo “appena acceso” entra in standby mode, non membro di alcuna piconet, minimo consumo.

**Inquiry mode:** Periodicamente il master invia 32 messaggi consecutivi usando 32 canali (wake-up channels, la sequenza è standard). I messaggi contengono un **IAC packet**. Gli slave periodicamente ascoltano i 32 canali per vedere se qualcuno ha mandato un IAC packet. Tutto questo in modo non coordinato.

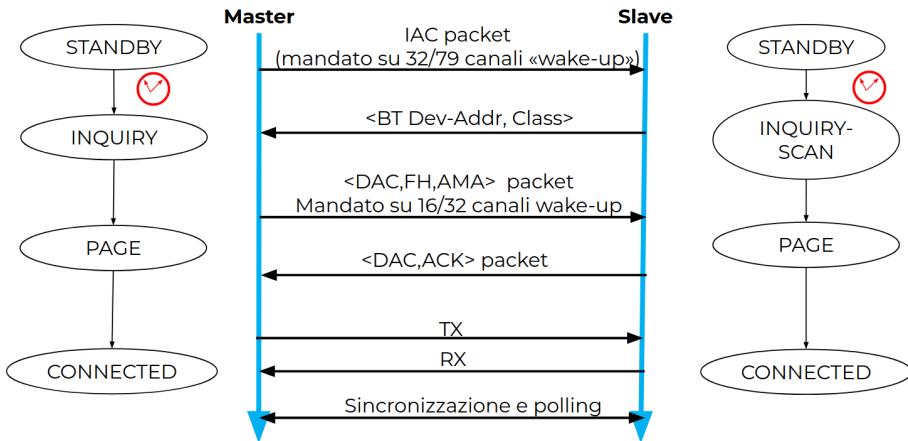
Ogni dispositivo ascolta per  $11,25ms$ , all’inizio di ogni intervallo di  $1,28$  o  $2,56s$ . Una volta “beccata” una trasmissione del master si aspetta un numero random di slot di tempo (random backoff) prima di trasmettere la risposta (per evitare che più slave che hanno ascoltato quella trasmissione rispondano nello stesso momento). Possiamo farlo dato che adesso l’unica cosa che sappiamo è il clock del master, ricevere un qualsiasi segnale del master permette di sincronizzarsi con la piconet. La risposta contiene il Bluetooth Device Address dello slave e la classe, per indicare il tipo di dispositivo.

Lo slave ascolta su una delle 32 frequenze di wake-up, mentre il master invia

sempre su tutte e 32. In questo modo, ascoltando per un certo periodo di tempo, se, anche solo *vagamente* allineato, lo slave vedrà la comunicazione del master (insomma, prima o poi la becca). Poi c'è un random backoff di un certo numero di slot di tempo prima di rispondere al master per richiedere il FH, DAC e AMA, inviato su 16 dei 32 canali di wake up (simile a prima).

Per finalizzare la connessione mancano le frequenze di hopping: si usano 16 delle 32 frequenze standard, per mandare: Device Access Code DAC, la sequenza di FH e l'Active Member Address AMA dello slave.

Lo slave risponde con DAC e ACK, poi può cominciare la comunicazione.



**Fase di paging:** Dopo la fase di inquiry, il master invia messaggi di paging per richiamare il dispositivo target, fornire tutti i dati di sincronizzazione necessari e stabilire definitivamente la connessione.

**Altri stati:** Un dispositivo “attivo” può essere:

- **Connected:** connessi ma non stiamo trasmettendo, sta solo ascoltando il master.
- **Transmit:** quando deve trasmettere va in transmit mode.

Entrambe queste modalità fanno uso del AMA.

Entrambe queste modalità consumano batteria (transmit un po' di più, ma comunque consumano), di conseguenza ci sono **3 stati di power saving** (in ordine discendente di consumo):

- **Sniff Mode:** non ascolta tutti gli slot (mantiene AMA)
- **Hold Mode:** ascolta solo canali SCO (mantiene AMA)
- **Park Mode:** rimane membro della piconet ma rilascia l'AMA e gli viene assegnato un PMA. Periodicamente ascolta i messaggi che il master invia in broadcast a tutti i membri parked. Rimane sincronizzato con clock e FH del master. Può tenere spenta la radio mentre non ascolta, per risparmiare batteria (ma ogni tanto va accesa per risincronizzare il clock, bisogna mantenere quello del master)

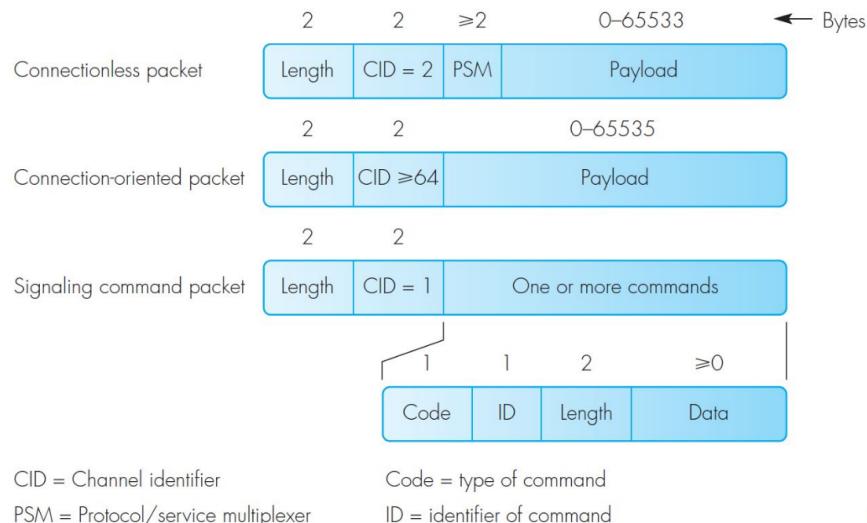
### 3.4.6 Logical Link Control and Adaptation Protocol L2CAP

Sempre presente in tutti i dispositivi Bluetooth, ma è il primo livello software (richiede combinazione di pacchetti a livello fisico). Non viene usato per l'audio (maggior parte dei canali SCO).

Supporta solo canali ACL. Offre 3 tipi di **canali logici**:

- **Connectionless**: unidirezionale, ad esempio tra broadcast e slave, per quando non c'è la necessità di instaurare una connessione
- **Connection-oriented**: bidirezionale e supporto QoS (richiedere una certa qualità del canale)
- **Signaling**: bidirezionale usato per messaggi di controllo master/slave, controllo di un livello superiore

#### Formato dei pacchetti:



La dimensione è segnata in byte, i pacchetti sono molto più grandi di quelli a livello baseband, la **segmentazione ed assemblamento** dei frame inviati a livello baseband viene **effettuata da L2CAP** (stile ethernet e trasporto, il messaggio sopra viene ricostruito, anche se sotto è diviso in più messaggi).

I valori del Channel Identifier possibili sono:

- CID = 2: non c'è nessuna connessione, pacchetto per connectionless
- ID  $\geq 64$ : pacchetto connection-oriented, il numero indica la connessione
- CID = 1: usato per i pacchetti di signaling/controllo

Al posto di avere dati relativi all'applicazione, il payload di un pacchetto di controllo contiene

- Code: codice per il tipo di comando
- Id: per l'identificatore del comando

Poi lunghezza e dati.

### 3.4.7 Service Discovery Protocol SDP

Protocollo **client-server** in cui un server contiene le informazioni ed il client ha due possibili azioni:

- ricerca di un servizio
- browse dei servizi (lista dei servizi disponibili su un dispositivo)

Generalmente, master chiede (client), slave rispondono alle richieste (server).

### 3.5 Bluetooth Low Energy BLE

Ha come **obiettivi**:

- ridurre il consumo energetico dei dispositivi
- entrare nel mondo degli smart sensor (di solito difficilmente ricaricabili, quindi da fare il meno possibile)
- semplificare il sistema di comunicazione (l'inquiry è complicato)
- compatibile con più dispositivi Bluetooth

Introduce **nuove funzionalità**: prima l'unica struttura possibile era master-slave, ma non bastava più per gli usi possibili del Bluetooth. Si aggiungono **altre strutture** di comunicazione:

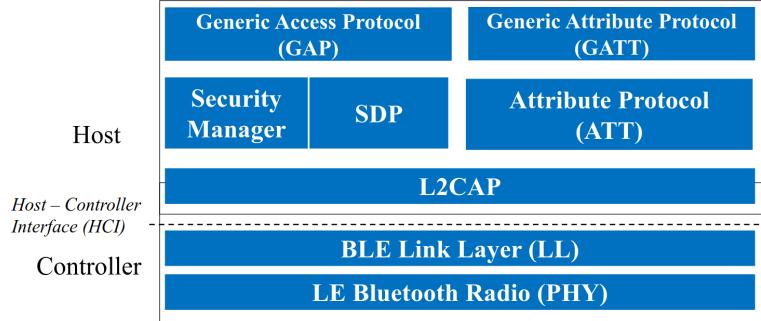
- **Broadcast**: più flessibile, chi è in range ascolta un broadcaster
- **Mesh**: struttura meno rigida, ogni dispositivo connesso a molteplici altri

Poi si aggiungono funzionalità di **positioning**:

- rilevare la **presenza** di un dispositivo
- rilevare la **distanza**
- rilevare la **direzione** dal dispositivo

Si tiene la stessa banda ISM, ma i canali sono 40 al posto che 79, rendendoli più resistenti ad interferenza (anche se abbassando il data rate).

### 3.5.1 Architettura



I livelli fisici cambiano in quanto è cambiata la trasmissione radio. Eccetto i primi 2, il resto è “equivalente” all’architettura del 2.1, con qualche modifica ovviamente.

### 3.5.2 Classi di potenza

| Power Class | Max Output Power ( $P_{max}$ ) | Min Output Power <sup>1</sup> |
|-------------|--------------------------------|-------------------------------|
| 1           | 100 mW (+20 dBm)               | 10 mW (+10 dBm)               |
| 1.5         | 10 mW (+10 dBm)                | 0.01 mW (-20 dBm)             |
| 2           | 2.5 mW (+4 dBm)                | 0.01 mW (-20 dBm)             |
| 3           | 1 mW (0 dBm)                   | 0.01 mW (-20 dBm)             |

La perdita di potenza (il minimo è minore) sono parzialmente “controbilanciate” dallo spettro più ampio che permette una migliore ricezione.

### 3.5.3 BLE Radio (PHY)

Non cambia lo spettro, sempre  $2.4GHz$  ISM, ma viene diviso in 40 canali, dei quali 37 usati per data packets, mentre gli ultimi sono usati come *advertising*.

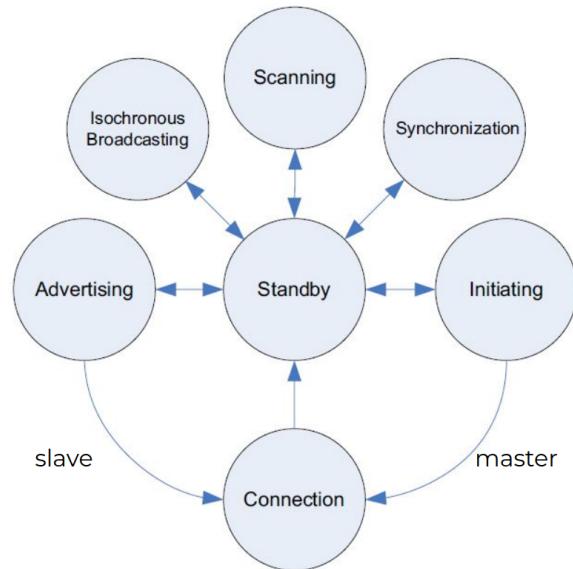
La sequenza di FH è determinata dalla formula

$$\text{channel} = (\text{curr\_channel} + \text{hop}) \bmod 37$$

il valore di “hop” è il “segreto” su cui fare viene fatto hopping.

Con la Gaussian Frequency Shift Keying (GFSK) con rate di modulazione si raggiunge  $1Mbps$ .

### 3.5.4 BLE State Machine



Tutti partono dallo standby.

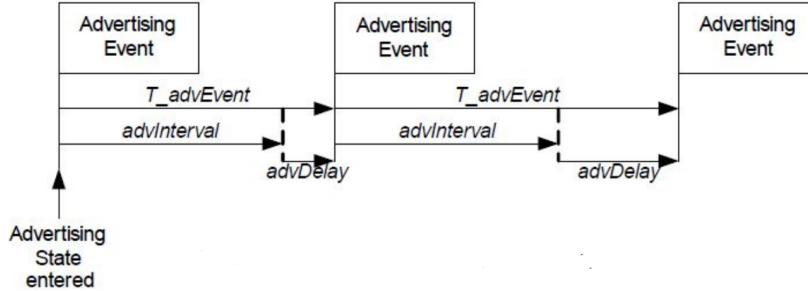
**Advertising:** usa i canali di advertising per farsi conoscere, non è più il master che cerca per creare la piconet, ma è lo slave che si annuncia.

**Initiating:** Lo stato in cui vengono ascoltati i messaggi di advertising.

**Isochronous Broadcasting:** broadcasting periodico (isocrono) per emettere delle informazioni.

**Scanning:** Il dispositivo si mette in ascolto.

### 3.5.5 Advertising



Ogni certa quantità di tempo viene inviato un **advertising event**, su uno o più dei canali di advertising. Il tempo tra un advertising event e l'altro è determinato da:

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

Dove

- $advInterval$  è un intero multiplo di  $625\mu s$  nel range  $20ms - 10,24s$  ed è determinato in base all'uso del dispositivo (per esempio, un sensore della temperatura non ha bisogno di inviare dati spesso quanto un accelerometro); il rate di invio determina il consumo della batteria
- $advDelay$  è un valore psudo-random nel range  $0-20ms$  (per evitare sovrapposizioni, sempre multiplo di  $625\mu s$ )

### 3.5.6 Generic Attribute Profile GATT

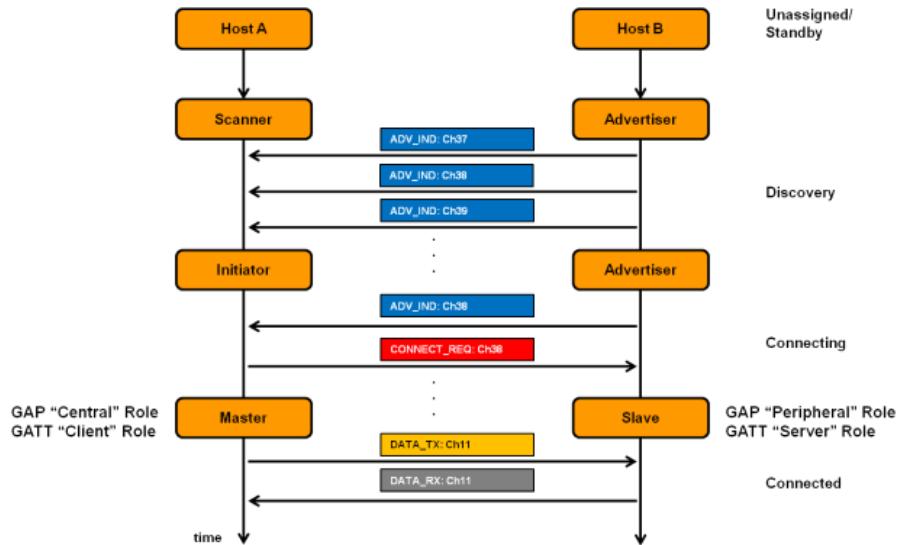
Viene gestito dal modulo GATT all'interno dell'architettura, permette uno scambio di dati strutturato tra server, che offrono servizi tramite profili (e.g., fascia che misura la frequenza cardiaca avrà un Heart rate profile) e client che richiedono (e.g., smartphone che chiede la frequenza cardiaca). Ogni profilo ha i suoi dati e caratteristiche associate. Il ruolo di server o client non è fisso, può cambiare dinamicamente in base alla necessità.

### 3.5.7 General Access Protocol GAP

Un'applicazione, in base al suo scopo, può decidere uno dei “ruoli fondamentali” definiti dal GAP:

- **Broadcaster:** spedisce pacchetti di advertising. Trasmissione di dati connectionless come eventi di advertising
- **Observer:** riceve advertising packet. Ricezione di pacchetti connectionless
- **Peripheral:** un peripheral device opera in slave (advertiser) mode a livello di Link Layer
- **Central:** un device opera in master (initiator) mode a livello di Link Layer

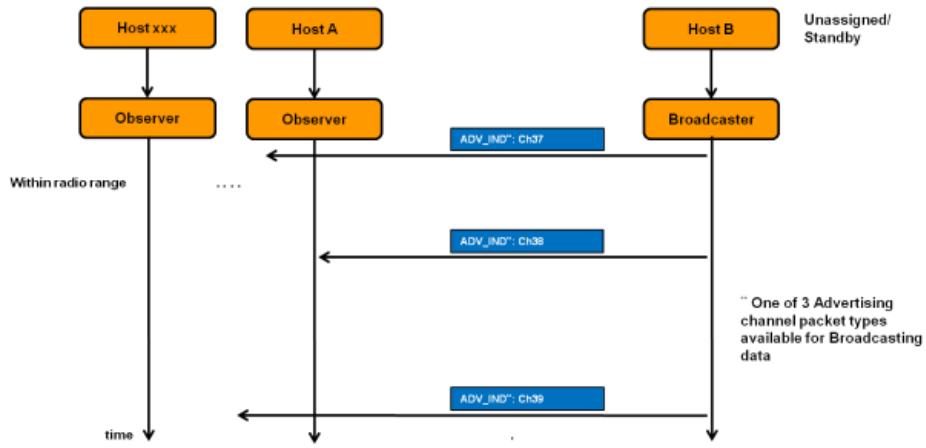
**Creazione di una Connessione Unicast (peer to peer):** Host A sarà il master, B lo slave.



In questo caso è il **master che ascolta** ed il dispositivo slave è un advertiser, vuole essere trovato dal master mandando advertising packets undirected (diretti a “nessuno”), vogliono solo dire ad un initiator che c’è *qualcuno* che vuole collegarsi. Vengono inviati sui 3 canali di advertising, di conseguenza il master ascolterà sugli stessi canali.

Una volta che il master “sente” il messaggio, invierà una connection request nello slot di tempo successivo (si mantiene sempre il TDD, sappiamo che nello slot di tempo successivo all’invio il dispositivo starà aspettando). Nella connection request ci sono anche le informazioni per il FH. Dopo la richiesta di connessione si passa a master e slave per la trasmissione.

**Connessione Broadcast:** Si può anche avere una connessione “broadcast”, non peer to peer, in cui abbiamo un broadcaster, che non ha interesse ad avere un master, vuole solo inviare dati.



Gli observer (quelli che ascoltano) saranno tutti i dispositivi all’interno del range di comunicazione.

**Passive scanning:** Lo scanner ascolta passivamente e periodicamente sui canali di advertising (solo passivo).

**Active scanning:** Sempre e solo usando i canali di advertising, lo scanner ascolta sul canale per poi richiedere dei dati tramite scan request (e di conseguenza otterrà la response). Quest’ultima parte è unicast con il dispositivo di broadcast.

### 3.6 ZigBee

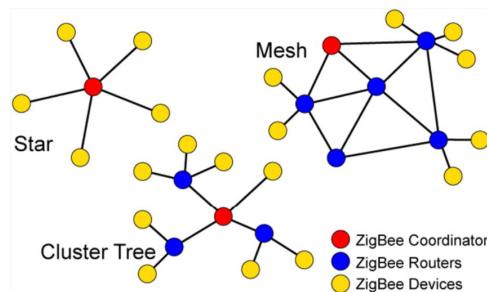
Sempre sotto 802.15, gli standard sono IEEE quindi siamo a livello fisico e data link.

I requisiti richiesti a ZigBee sono:

- Affidabilità
- Basso costo
- Lunga durata della batteria (molto più che Bluetooth)
- Bassa complessità
- Utilizzo delle bande ISM ( $2.4GHz$  Worldwide &  $915MHz/ 868MHz$  in base alla zona)
- Scalabilità (possibile un alto numero di nodi), con diversi possibili struttture
- Interoperabilità tra vendors
- Sicurezza

Gli utilizzi sono principalmente in ambiti di domotica, IoT, monitoring, ma possono essere i più svariati, qualunque ambito richieda comunicazione di “pochi” dati tra più dispositivi con un basso consumo di energia.

Topologie di rete Sono possibili più tipologie di rete: stella, albero, mesh.



Si introducono topologie multi-hop, quindi serve avere del routing.

Ci sono due macro classi di nodi:

- **Full Function Device FFD**
- **Reduced Function Device RFD**

Il nome è abbastanza autoesPLICATIVO, tra i FFD c'è un solo coordinatore (ZigBee Coordinator) ed uno o più Router, i quali possono fare instradamento. Gli end device sono (solitamente) RFD, non possono fare routing, possono solo comunicare e ricevere dati verso il coordinatore/loro router di competenza (questi sono sensori, raccolgono dati/piccole azioni).

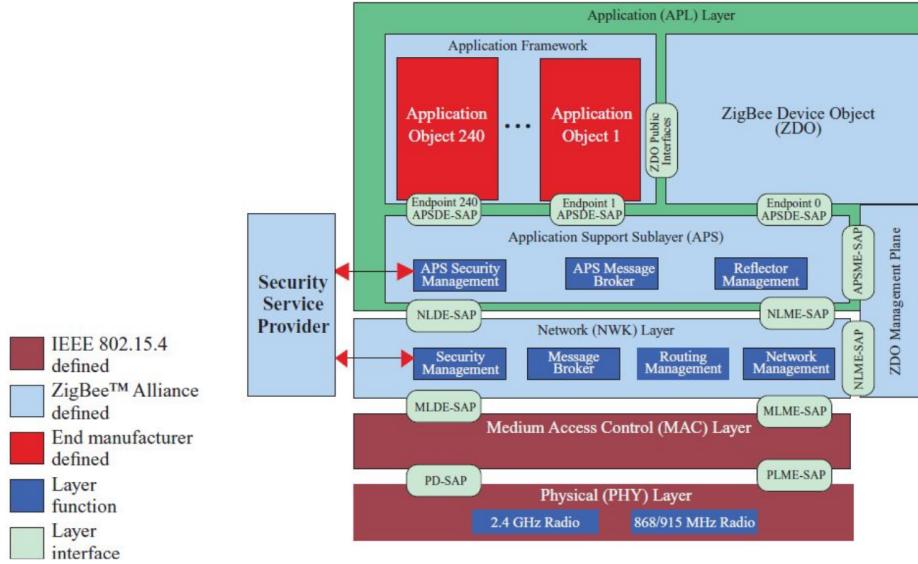
Quindi i dispositivi possono fare da:

- **Coordinator** (PAN coordinator) FFD: unico all'interno della rete, la crea e ne mantiene le informazioni (es. chiavi di sicurezza)
- **Router** FFD: nodi che hanno la capacità di inoltrare dati tra i dispositivi ZigBee
- **End device** RFD: da punto di vista di rete possiedono solo la funzionalità di parlare ad un router/coordinatore; ridotta complessità ed elevato risparmio energetico

**Tipologie di invio dati:** Ci sono diverse possibili tipologie di dati, in base al caso d'uso:

- **Dati periodici:** invio dopo un intervallo di trasmissione fissato (es: sensori per misurare qualcosa)
- **Dati intermittenti (asincroni):** i dati vengono comunicati in base ad un evento (es: qualsiasi tipo di attuatore come un interruttore)
- **Dati ripetitivi e a bassa latenza:** allocazione di time slot (es: mouse)

**Architettura:** L'architettura di ZigBee si presenta come:



### 3.6.1 Livello Fisico 802.15.4 (PHY)

Lo standard 802.15.4 specifica la tipologia di modulazione e spread spectrum per le 3 bande

| Banda           | # Canali | Spread Spectrum   |                    |                        | Data rate                                  |          |
|-----------------|----------|---|--------------------|------------------------|--|----------|
|                 |          | Tipo  | Chip Rate (chip/s) | Modulazione (dei chip) | Symbol rate                                | Bit rate |
| 868-868.6 MHz   | 1        | DSSS<br>1bit → 15 chip (max)                                  | 300.000            | BPSK<br>1 chip         | 20 ksym/s                                  | 20 kb/s  |
| 902-928 MHz     | 10       | DSSS  | 600.000            | BPSK<br>1 chip         | 40 ksym/s                                  | 40 kb/s  |
| 2400-2483.5 MHz | 16       | DSSS<br>4 → 32 chip (16 sequenze di 32 chip quasi ortogonali) | 2.000.000          | O-QPSK<br>2 chip       | 62.5 ksym/s (1 simbolo di 32 chip → 4 bit) | 250 kb/s |

Si usa multiplexing su canali all'interno della banda, spread spectrum DSSS per la comunicazione. Si può notare che i data rate sono molto limitati, ma i dati generalmente trasferiti da sensori/casi d'uso di ZigBee sono anch'essi abbastanza ridotti.

### 3.6.2 Livello Data Link 802.15.4 (MAC)

Questo livello si occupa di:

- Gestire l'invio dei beacon (se il dispositivo è PAN Coordinator)
- Sincronizzazione con i beacon del coordinatore (router & end device)
- Associazione/dissociazione alla Pan ascoltando i beacon
- Accesso al canale tramite CSMA/CA
- MAC Address (16 o 64 bit)
- Gestione del duty cycle del dispositivo

Si possono avere due tipi di trasmissioni:

- diretta: dispositivo  $\leftrightarrow$  coordinatore
- indiretta: coordinatore  $\rightarrow$  dispositivi

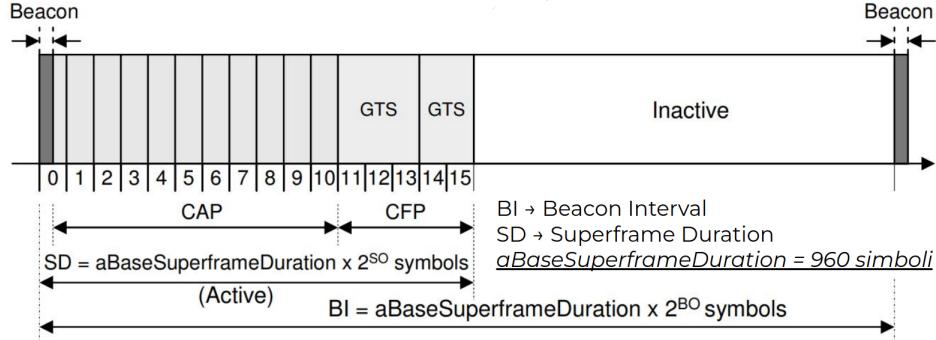
**Modalità di trasferimento:** Ci sono due modalità

- **Unslotted CSMA/CA** senza ausilio di beacon
- **Slotted CSMA/CA utilizzo di beacon**

**Slotted CSMA/CA:** La modalità slotted si fonda sull'invio di **beacon**. Questi sono inviati dal coordinatore ed eventualmente inoltrati dai router. Il coordinatore invia periodicamente dei beacon per:

- sincronizzare gli altri dispositivi
- organizzare i periodi di trasmissione per le diverse tipologie di trasmissione (periodiche, asincrone e bassa latenza)
- gestione della trasmissione indiretta: il coordinatore mantiene in una lista le frame non ancora mandate ai dispositivi (poi i dispositivi faranno “Ah devo ricevere qualcosa, ascolterò quello che ha da mandarmi”); nei beacon frame trasmette anche la lista dei dispositivi che hanno frame pendenti (gli dà il permesso di parlare); i dispositivi che ascoltano i beacon sanno se c’è qualcosa per loro

L'organizzazione che il coordinatore ha in mente (solo logica) per la gestione del tempo di comunicazione è definita **superframe** (sostanzialmente tutto ciò che passa tra un beacon e l'altro).



La prima metà è un momento di attività, la seconda è di inattività, poi si ripete il duty cycle. Il **duty cycle** vuol dire che si alternano periodi di attività, in cui la radio è accesa, a periodi di inattività, a radio spenta (momenti in cui non ci sono trasmissioni programmate, quindi è inutile tenere la radio accesa). Per sincronizzare il duty cycle, all'interno di ogni beacon è presente l'informazione su quando sarà il beacon seguente e questo accenderà la radio appena prima.

Se un dispositivo non deve inviare/ricevere niente, spegne la radio fino al superframe successivo.

Nello standard c'è una  $aBaseSuperframeDuration = 960$  symbols, quindi ogni beacon avrà una durata multipla di questa unità base. Due parametri che determinano il duty cycle:

- **Beacon Order BO:** determina il beacon interval 0-14, il quale dura

$$BI = aBaseSuperframeDuration \cdot 2^{BO} \text{ symbols}$$

- **Superframe Order SO:** determina la durata del superframe 0-14:

$$SD = aBaseSuperframeDuration \cdot 2^{SO} \text{ symbols}$$

960 simboli sono  $\sim 15.3ms$  in banda  $2.4GHz$ . Il duty cycle è dato da

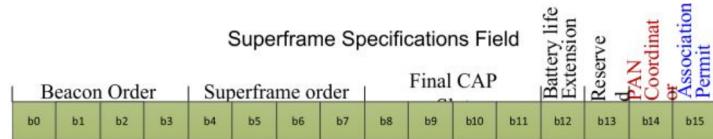
$$Duty - Cycle = \frac{2^{SO}}{2^{BO}}$$

Quindi, un sensore, potrebbe accendere la radio per circa  $15ms$  per poi spegnere la radio fino alla trasmissione successiva, che potrebbe essere dopo  $15$  minuti. In realtà si riaccende un pochino prima del beacon per tenere conto di una possibile deriva del clock.

**Beacon Frame:** Un Beacon Frame è composto da

| 2 Bytes       | 1 Bytes               | 2 Bytes       | 2/8 Bytes      | 2 Bytes          | variable  | variable              | variable       | 2 Bytes                    |
|---------------|-----------------------|---------------|----------------|------------------|-----------|-----------------------|----------------|----------------------------|
| Control Frame | Beacon Seq. No. (BSN) | Source Pan ID | Source Address | Superframe Specs | GTS Field | Pending Address Field | Beacon Payload | Frame check Sequence (FCS) |

Il campo Superframe Specs è a sua volta diviso in



Dove:

- Beacon Order: 4 bit per il beacon interval, la frequenza con cui vengono trasmessi i beacon
- Superframe Order: 4 bit, definisce la durata della parte attiva del superframe
- Final CAP: 4 bit, stabilisce dove termina il periodo CAP all'interno della parte attiva
- Battery Life Extension: 1 bit, se a 1 abilita un meccanismo di risparmio energetico
- Riservato: 1 bit
- PAN Coordinator: 1 bit, indica se chi trasmette il beacon è il coordinatore o un router
- Association Permit: 1 bit, indica se altri dispositivi possono associarsi alla rete tramite il dispositivo che trasmette il beacon

Gli altri campi all'interno del beacon sono:

- Control Frame: specifica il tipo di frame ed altri parametri relativi alle caratteristiche della trasmissione
- Beacon Sequence Number: valore incrementale per identificare il singolo beacon
- Source PAN ID: indica l'ID di rete del dispositivo che trasmette il beacon, identifica l'appartenenza del beacon ad una specifica rete
- Source Address: Indirizzo del dispositivo che trasmette il beacon
- Guaranteed Time Slot (GTS) Field: specifica i time slot riservati per la trasmissione di alcuni dispositivi collision free
- Pending Address Field: indica se il coordinatore ha dati in sospeso per uno o più dispositivi
- Beacon Payload: può contenere informazioni aggiuntive specifiche del livello di rete ZigBee o altre informazioni proprietarie del produttore
- Frame Check Sequence (FCS): campo per il controllo dell'integrità (CRC)

Se un dispositivo legge il GTS e il Pending Address e capisce di “non aver nulla da fare”, può spegnere la radio fino al prossimo beacon.

Il superframe viene diviso in **slot con diversi tipi di accesso**:

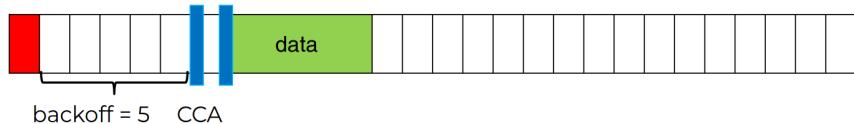
- **Contention Access Period CAP** (valore di default 16): Accesso al canale usando CSMA/CA
- **Contention Free Period CFP** (da 0 a 7 slot): Intervallo per le comunicazioni con banda riservata tramite Guaranteed Time Slot

**Slotted CAP CSMA/CA:** Canale a contesa, Carrier Sense Multiple Access con Collision Avoidance. Per capire quando è libero il canale viene fatto un **Clear Channel Assessment** (CCA) per brevi periodi (8 simboli). Ci sono dei parametri che determinano come questo viene fatto:

- $NB$  = Numero di Backoff, inizialmente 0, aumenta di 1 se il canale è occupato
- $BE$  = Backoff Exponent, indica il range del periodo di backoff  $[0, 2^{BE} - 1]$
- $CW$  = Contention Window, indica il numero di slot liberi consecutivi necessari prima di cominciare a trasmettere

Ogni contention slot è 20 simboli.

Viene scelto un intero casuale nell'intervallo determinato dal  $BE$ , si aspetta quel numero di contention slot, e poi vengono fatti  $CW$  CCA, se dopo gli assessment il canale è libero il dispositivo comincia a trasmettere. I CCA sono ravvicinati tra loro.



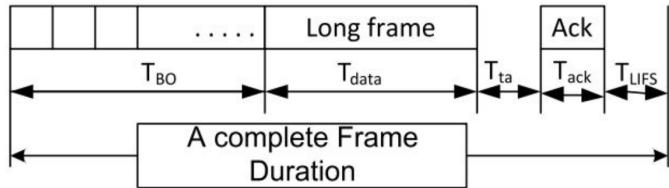
Se durante uno dei CCA viene ricevuta la trasmissione di un altro dispositivo viene allargata la finestra del random backoff (“siamo un po’ meglio aspettare per essere sicuri”), incrementando  $BE$  e  $NB$ , prima di fare un altro random backoff e tentativi di CCA. Dopo 4 tentativi ( $NB = 4$ ) il livello MAC dichiara transmission failure.

Se il numero di contention slot in CAP non è sufficiente (ho meno slot rispetto al valore casuale che ho estratto), il conteggio viene interrotto e ripreso al superframe successivo (per evitare che un dispositivo sfortunato non trasmetta mai).

In qualsiasi caso, tutta la comunicazione deve terminare prima degli slot GTS, tutto deve essere incluso nel CAP. Se un dispositivo non avrebbe tempo di trasmettere i dati che deve (troppi pochi slot rimanenti), la trasmis-

sione viene rimandata al superframe successivo.

Una trasmissione completa diventa:



Tempo di backoff  $T_{BO}$ , un frame di dati  $T_{data}$ , un tempo di turn around  $T_a$  (aspettare che la radio passi da TX a RX), ricezione dell'ack ed un tempo per chiudere la telecomunicazione  $T_{LIFS}$  (long inter-frame space).

**Unslotted Non Beacon Mode:** Nella modalità senza beacon i dispositivi accedono al canale usando CSMA/CA senza i vincoli di slot. Non c'è sincronizzazione, il tempo è continuo, il controller è più semplice.

Sostanzialmente si ripete la fase di backoff e CSMA/CA finché non il dispositivo non riesce a trasmettere. Non essendoci sincronizzazione, tutto ciò che non è end device deve essere sempre attivo (radio sempre accesa, non sapendo quando qualcuno trasmetterà).

### **3.6.3 Livello di rete**

Il livello di rete (Network Layer) di ZigBee (definito nello standard ZigBee, **sopra** l'IEEE 802.15.4) si occupa della gestione e del mantenimento della topologia di rete e delle operazioni di routing.

In particolare, gestisce formazione della rete, indirizzamento sopra al MAC e la topologia (supportando la riconfigurazione). Dato che ci sono trasmissioni che devono giungere a dispositivi non in connessione diretta è necessario routing, questo viene gestito a livello di rete tramite protocolli di routing mesh come **Ad-Hoc On Demand Distance Vector AODV**.

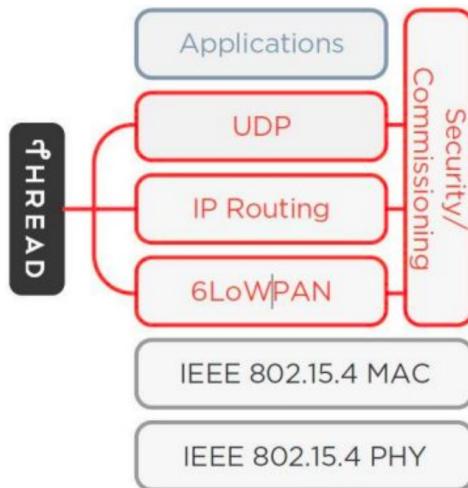
### **3.6.4 ZigBee Device Object ZDO**

Posizionato sopra il livello di rete, definito nello standard ZigBee. Definisce **il ruolo del dispositivo**, permette la scoperta di nuovi dispositivi e le loro funzionalità. Inoltre fornisce un'interfaccia comune per le applicazioni.

Stabilisce come un dispositivo si inserisce nella rete, come scopre gli altri nodi e come gestisce la sicurezza e le funzionalità di base. Ogni nodo ZigBee ha il proprio ZDO che fornisce servizi di gestione e supervisione, rendendo possibile l'interoperabilità fra dispositivi di diversi produttori.

### 3.7 Matter & Thread

Thread ha il livello fisico e MAC secondo standard 802.15.4, ma sopra usa uno stack



- Standard 6LoWPAN, compressione header, supporto a rete mesh
- IP Routing, indirizzamento IPv6, distance vector routing, costi dei link variabili in base a RSSI (Received Signal Strength Indicator)
- UDP

**Rete Thread:** Cambiano un po' le capacità dei dispositivi, ci sono:

- **Routing Full Thread Device:** possono essere
  - **router:** effettua routing e fornisce servizi di accesso e sicurezza (NoSleep, può essere “degradato” a Router-Eligible End Device REED)
  - **leader:** un router con funzionalità aggiuntive che può eleggere/destituire REED
- **Non routing Full Thread Device:** si dividono in
  - REED
  - Full End Device FED, non possono essere eletti router

I Non Routing Minimal Thread Devices sono dispositivi con requisiti HW minori:

- Minimal End Device (MED): comunica solo con il router genitore e radio sempre attiva
- Sleepy End Device (SED): comunica solo con il router genitore e duty-cycle
- Synchronized Sleepy End Device (SSED): comunica solo con il router genitore e duty-cycle secondo intervalli schedulati

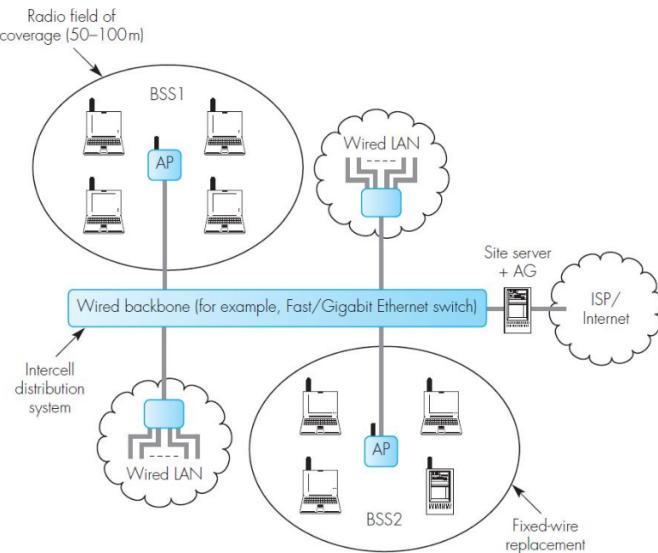
**Border router:** Sono dei dispositivi di tipo Full Thread Device che forniscono connettività verso le altre reti con un livello fisico diverso (Wi-Fi, Ethernet, ...). Offre funzionalità di routing all'esterno della rete thread.

## 4 WiFi 802.11

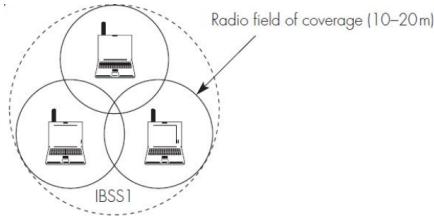
Lo standard considerato da adesso è 802.11. I requisiti generali sono:

- Throughput maggiore possibile
- elevato numero di nodi, gestiti da più celle
- connessione verso la dorsale (backbone) cablata
- Raggio 100-300m
- Utilizzo efficiente (non estremo) della batteria
- Più WLAN devono poter coesistere
- Operare nelle bande di frequenza unlicensed
- Configurazione dinamica (selezione canali, autenticazione)

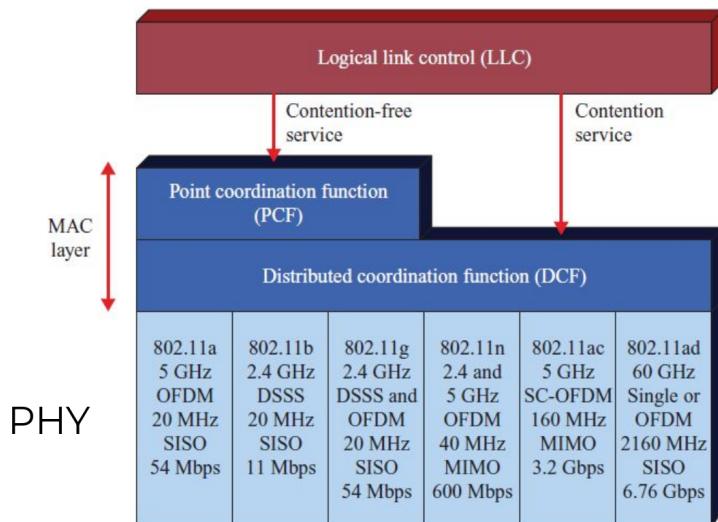
**Wireless Fidelity WiFi:** Si ha una rete con **uno o più Access Point AP**, coordinata da **Point Coordination Function PCF** (ovvero si ha un solo punto di coordinamento, l'AP). Un **Basic Service Set BSS** identifica la cella o rete WiFi (nome/caratteristiche della rete).



Si può anche avere una **rete Ad-Hoc**, senza una PCF, ma con una **Distributed Coordination Function DCF**, con un **Independent Basic Service Set IBSS**



### Architettura dei protocolli:



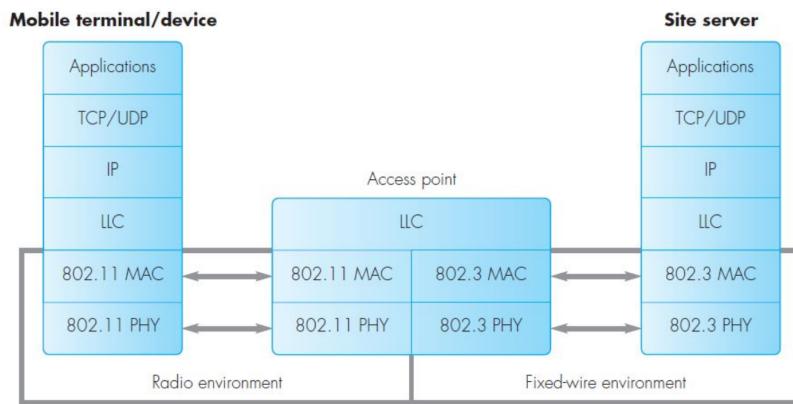
Mostrate in ordine da sinistra a destra, a livello fisico ci sono le versioni di WiFi, anche se l'ultimo rappresentato non è WiFi 6 (quello è 802.11ax). Sopra c'è il livello MAC, poi il Logical Link Control che permette di avere servizi.

Il throughput diverso (visibile [in questa tabella](#)) è determinato da dimensione del canale, aumentata nel tempo, frequenze utilizzate, modulazioni con più bit per symbol e MIMO (Multiple Input Multiple Output, fino a 16 antenne diverse per fare input/output).

**Servizi Logical Link Control LLC:** Offre principalmente tre servizi:

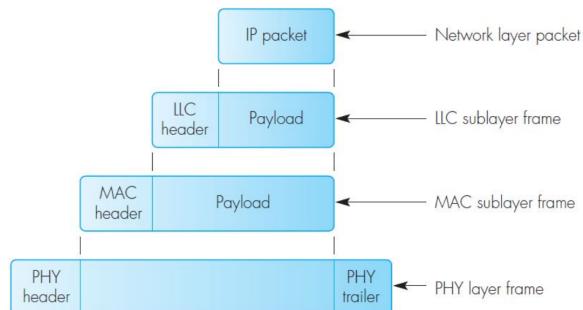
- **Unacknowledged connectionless service:** consegna non garantita, datagram indipendenti, senza controllo di errori o di flusso; cosa capita capita
- **Connection-mode service:** canale punto-punto, correzione degli errori e controllo di flusso; connessione affidabile
- **Acknowledged connectionless:** datagram indipendenti, acknowledged datagram; senza connessione, ma con ack

### Infrastruttura 802.11:



Il livello LLC è comune a 802.X.

**Livelli Pacchetti:** La struttura per l'incapsulamento dei livelli è



Ovvero, fisico, MAC, LLC, poi pacchetto di rete.

## 4.1 Sottolivello MAC

Abbiamo un canale *molto* inaffidabile (rispetto ad un canale cablato), con molte informazioni da trasmettere, e di conseguenza frame più complessi (rispetto a 802.3). Il frame di MAC 802.11 ha un payload massimo di  $2304B$ .

Il sottolivello MAC offre due servizi:

- **Servizio dati asincrono:** non ha garanzie sul delay e quindi sulla QoS, best effort
- **Servizio time-bounded:** offre garanzie sul delay, disponibile solo in presenza di un coordinatore (AP)

### 4.1.1 Distributed Coordination Function DCF

Opera in **CSMA/CA**: l'accesso al canale deve essere regolato aspettando del tempo per poter trasmettere. 802.11 prevede diversi tempi di attesa a seconda della tipologia di dati da trasmettere.

Definizioni:

- **Slot time:** una unità base di tempo (interna al dispositivo), non si tratta di una suddivisione temporale; tiene conto del ritardo di propagazione e della tipologia di trasmettitore utilizzato (tipo di WiFi fisico usato, es:  $20\mu s$  per 802.11b)
- **Short Inter-frame Spacing SIFS:** l'intervallo più breve di attesa usato per messaggi ad alta priorità, altra unità base di tempo; la durata dipende dalla tipologia di trasmettitore
- **DCF Inter-frame Spacing DIFS:** intervallo di tempo più lungo usato per messaggi a bassa priorità best effort:  $SIFS + 2 * slot\_time$
- **PCF Inter-frame Spacing PIFS:** intervallo di tempo intermedio per time-bounded:  $SIFS + 1 * slot\_time$

Nel caso di trasmissione con canale libero:

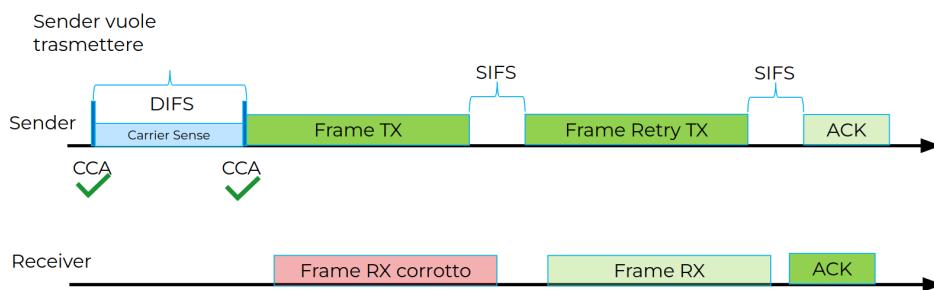
- il sender inizia ad ascoltare il canale:
- fa un Clear Channel Assessment CCA
- ascolta per un tempo *lungo* (ovvero DIFS) il canale
- fa un altro CCA

Se non ha sentito nulla tra il primo ed il secondo CCA trasmette.

Se **non è necessario ack**, una volta terminata la trasmissione il canale è libero.

Se **necessario l'ack**, l'attesa dell'acknowledgement ha durata *breve* SIFS (più breve del tempo per i CCA, altrimenti potrebbe andare in conflitto con CCA di altri dispositivi); dal RX l'ack viene inviato il prima possibile (giusto il tempo di turnaround e di preparare il pacchetto).

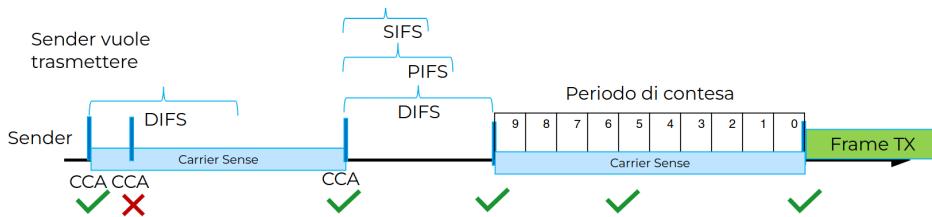
Se il **frame viene corrotto** prima della ricezione, mancherà l'ack, il TX aspetta tempo SIFS e se non riceve niente assume che la trasmissione non sia andata a buon fine e ritrasmette subito lo stesso frame; “subito” perché ha ottenuto l’accesso esclusivo al canale e lo tiene in maniera esclusiva finché la trasmissione non è terminata; il tempo è breve per evitare che altri “si intromettano”. Il rilascio del canale avviene solo a trasmissione completata e ack ricevuto. Da standard 802.11 è previsto un massimo di tentativi.



Se il **canale è occupato**, ovvero durante il tempo di attesa DIFS viene ricevuto un segnale, fallisce il CCA ed il dispositivo ascolta fino al termine della trasmissione dell’altro sender.

Quando il canale è di nuovo libero (nuovo CCA), si aspetta un periodo di tempo DIFS, PIFS o SIFS, deciso in base alla priorità del messaggio, prima di avere un periodo di contesa (tutti i dispositivi che devono trasmettere aspettano il momento di fine trasmissione).

Durante il periodo di contesa il dispositivo aspetta un numero random di slot\_time da attendere (Binary Exponential Backoff). Carrier sense durante tutto il priodo di backoff.



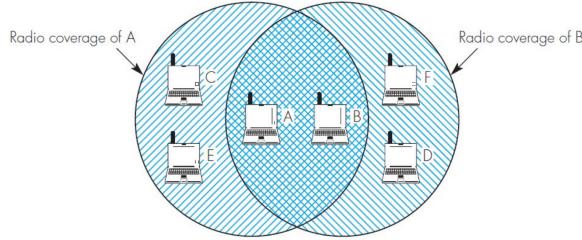
Da notare come la radio rimanga sempre accesa, non ci sono più requisiti di batteria come in Bluetooth o ZigBee.

Se durante il periodo di contesa il **canale diventa occupato**, due opzioni:

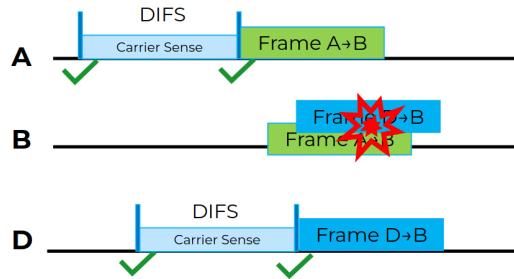
1. Riparto dalla contesa (con intervallo più ampio) al prossimo ciclo; non equa nei confronti delle stazioni che hanno “perso”, rischio attesa infinita
2. Blocco il timer al valore in cui era arrivato nel momento in cui è stato rilevato il canale occupato e nel **turbo successivo riparto** da quel valore

#### 4.1.2 Problema del terminale nascosto

Il meccanismo di carrier sense funziona se l'altro dispositivo che comincia a trasmettere può essere rilevato. Ad esempio, nel caso



A e D non si possono sentire in quanto fuori dal rispettivo raggio di copertura. Se entrambi volessero trasmettere a B vedrebbero il canale libero nello stesso momento, causando una collisione.



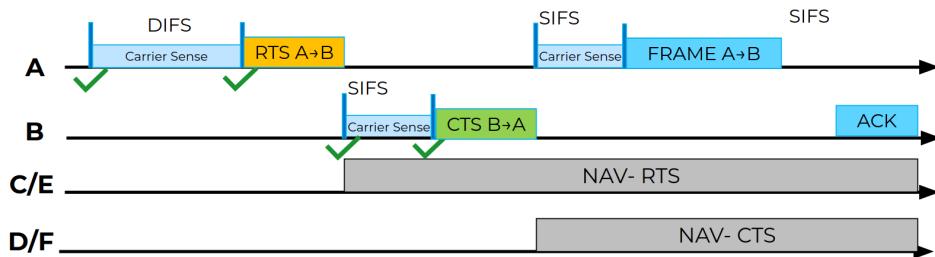
Per risolvere, dopo aver fatto carrier sense, il sender invia una **Request to Send RTS** contenente sorgente e destinazione della richiesta oltre alla durata stimata da RTS ad ack finale.

Il RTS viene ricevuto da tutti i terminali nel raggio di copertura del sender, i quali sanno che la richiesta non è per loro e usano le informazioni sul tempo previsto per allocare un **Network Allocation Vector NAV**, ovvero un tempo nella quale sanno che non potranno trasmettere.

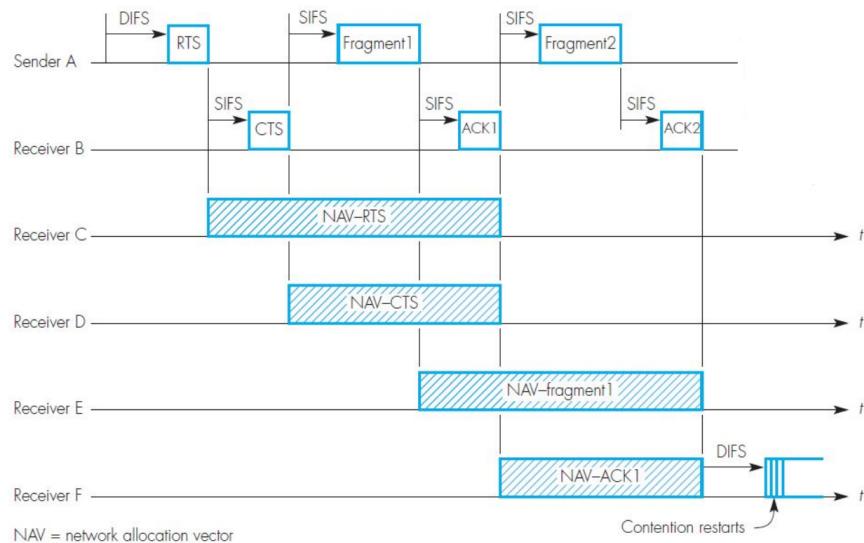
Il destinatario, se è libero, risponderà (dopo tempo SIFS, per evitare che comincino altre comunicazioni nel frattempo) con un **Clear to Send CTS**, contenente sorgente, destinazione ed il tempo rimanente fino alla fine della trasmissione.

Il CTS viene ricevuto da tutti i terminali nel raggio di del destinatario, i quali allocheranno un **NAV** per il tempo indicato nel CTS (ovviamente minore di quello del RTS, è passato del tempo). Vengono a conoscenza del fatto che un altro nodo all'esterno del loro raggio di copertura vuole comunicare con un nodo a loro visibile, quindi non tentano nemmeno di accedere al canale.

Dopo che il sender originale ha ricevuto il CTS aspetta SIFS per poi inviare il frame ed aspettare di ricevere l'ack per terminare la trasmissione.



**Frammentazione:** Il canale radio è molto più sensibile ad interferenza e rumore. Considerando la dimensione della frame Ethernet (1522 byte) è molto probabile che ogni frame abbia errori; meglio **ridurre la dimensione della frame MAC**. Frame LLC viene suddivisa in frammenti più piccoli la cui dimensione è variabile in base alle condizioni del canale.

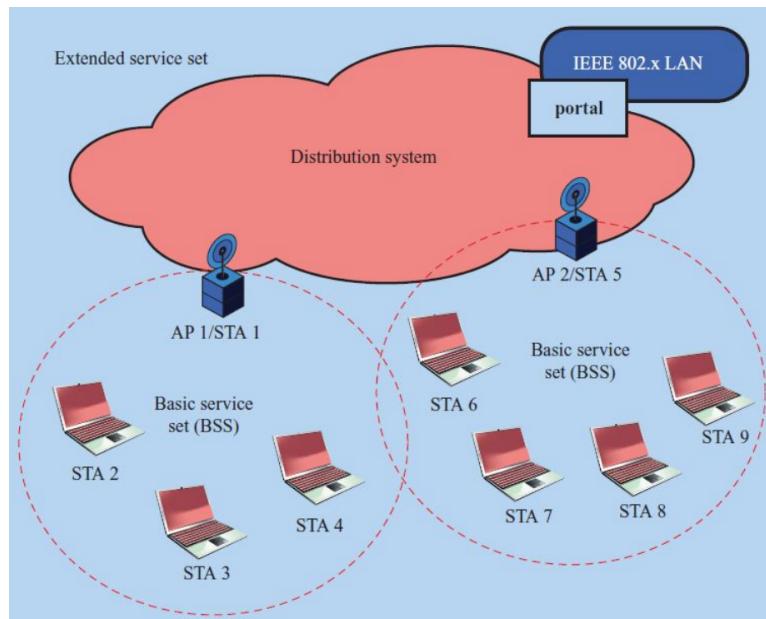


All'interno di ogni frame “frammentato” si inseriscono informazioni riguardo al NAV per i dispositivi non direttamente coinvolti nella comunicazione.

La frammentazione e correzione dei dati viene fatta a livello MAC, tra dispositivo e AP, dove il ritardo dovuto alla correzione è minimo. Si potrebbe lasciare la correzione dei dati al livello superiore (TCP), ma sarebbe il destinatario finale e non l'AP ad accorgersi dell'errore, allungando di molto i tempi di ritrasmissione.

#### 4.2 802.11 Con Infrastruttura

Nel caso più semplice si ha un solo AP con il relativo Basic Service Set BSS, l'insieme di stazioni controllate dal singolo coordinatore. Si può anche avere un Extended Service Set ESS, ovvero un insieme di più BSS interconnessi tramite un sistema distribuito (DS).



Il portale è un router/bridge che collega il sistema distribuito alla LAN. ESS viene visto come un unico BSS a livello LLC per funzionalità di roaming (muoversi nella rete e rimanere collegato) tra AP diversi (overlapping almeno 10% per garantire continuità).

#### 4.2.1 Point Coordination Function

In presenza di un AP, tutti i frame passano per l'AP. Questo permette di avere **servizi time-bounded**, con garanzie sul delay, non possibili con DCF (il random backoff non ha garanzie). Nella modalità PCF, l'AP controlla l'accesso al canale radio:

1. Tutto il traffico passa dall'AP
2. Le stazioni associate ad AP usano DCF con tempistiche SIFS e DIFS per accedere al canale
3. AP usa PIFS

In questo modo AP riesce ad “impossessarsi” del canale radio prima delle stazioni in attesa.

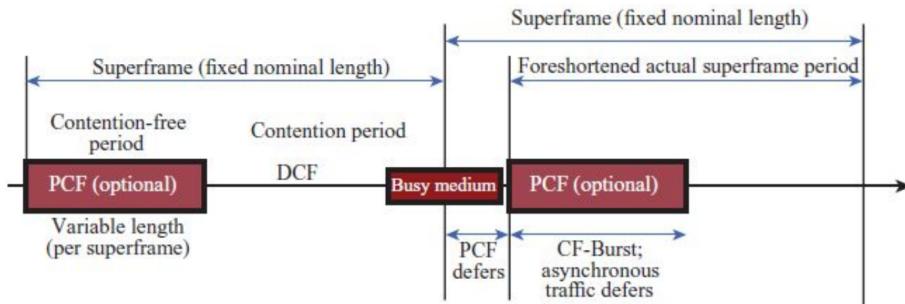
AP manda messaggi periodici (10-100s) detti **beacon frame** che sono frame di gestione contenenti

1. Parametri operativi PHY (bit rate & Modulation Coding Scheme)
2. Sincronizzazione (usato nelle prime versioni 802.11 che usavano FHSS)
3. Supporto a PCF con le relative informazioni
4. Invito per le nuove stazioni che non sono ancora associate

Ovvero, una rete manda dei beacon per “farsi scoprire” e comunicare le proprie caratteristiche per le nuove associazioni.

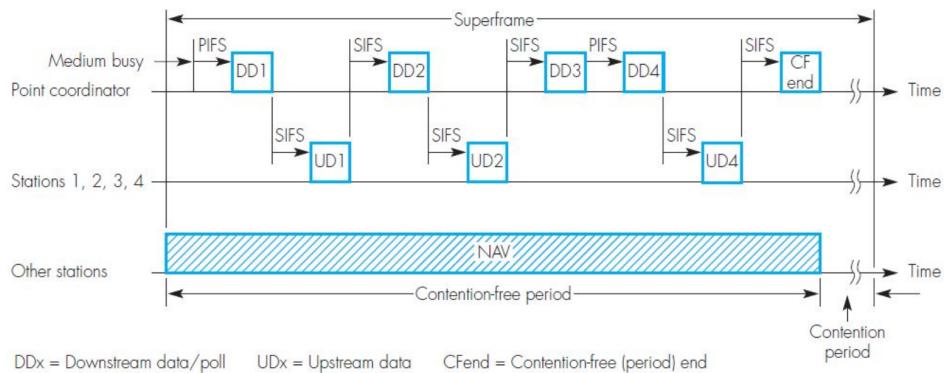
Il tempo è diviso in blocchi (superframe), all'interno dei quali si ha una divisione in due periodi:

- una parte **senza contesa** opzionale, ma necessario per servizi time bounded
- accesso **a contesa** (sempre presente)



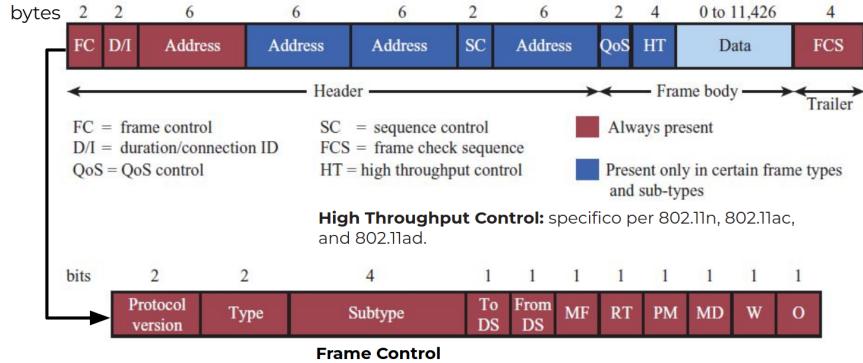
Se il canale radio è occupato oltre il limite del superframe il tempo non verrà recuperato (si aspetta).

Quindi la PCF ha un funzionamento del tipo:



Al termine della comunicazione precedente, tutti i dispositivi attendono DIFS ma l'AP aspetta PIFS, quindi prende il lock sul canale prima degli altri, iniziando il periodo senza contesa. In seguito invia il "permesso di trasmettere" per ogni trasmissione necessaria, aspettando SIFS tra una e l'altra (per non rischiare di perdere il lock sul canale). Al termine si manda un Contention Free End per marcare la fine del periodo senza contesa.

#### 4.2.2 Formato Frame MAC



In rosso la parte sempre presente, in qualsiasi frame 802.11 sono presenti i campi in rosso. Gli altri sono presenti solo se necessari.

I primi 2 byte sono **Frame Control** FC, il quale contiene:

- Protocol version: versione del protocollo
- 2 bit di Type
- 4 bit di sottotipo
- To DS/From DS sono bit che indicano se verso o dal Distributed System
- More Fragments MF: indica se c'è frammentazione
- Retry RT se si tratta di una retransmissione
- Power Management PM per indicare se è attivo il risparmio energetico
- More Data MD se ci sono ulteriori dati
- 2 bit di controllo

Il secondo gruppo di 2 byte contiene o la **Duration o Connection ID** (D/I). Contengono la durata della trasmissione rimanente (in  $\mu s$ ), per gestire i NAV. Nei frame Power Save Poll contiene l'Association ID del client che chiede informazioni.

Poi c'è l'**indirizzo di destinazione** del frame e **fino ad altri 3 indirizzi**, che possono esserci in base all'utilizzo, alla fine correzione degli errori.

**Frame Control:** Per quanto riguarda il tipo, ne esistono 3:

- 00 Management
- 01 Controllo
- 10 Data

Sono macro-tipi che possono assumere i frame, per andare nello specifico bisogna guardare il sotto-tipo. Ad esempio: nei frame di management 1000 sono i beacon, nei frame di controllo 1011 è una RTS mentre per i data frame 0101 indica un CF-Ack.

**Indirizzi:** Possono esserci **fino a 4 indirizzi**, il cui significato cambia in base ai valori di To DS e From DS:

| To DS | From DS | Address 1  | Address 2  | Address 3   | Address 4  | Caso di utilizzo   |
|-------|---------|--|--|---|--|--|
| 0     | 0       | <b>DA</b><br>Indirizzo MAC destinazione                      | <b>SA</b><br>Indirizzo MAC sorgente                    | <b>BSSID</b><br>della cella/<br>random se ad hoc        | -  | Rete ad hoc<br>Rete con infrastruttura singola cella   |
| 0     | 1       | <b>DA</b><br>Indirizzo MAC destinazione all'interno di BSSID | <b>BSSID</b><br>della cella a cui la frame è destinata | <b>SA</b><br>Indirizzo MAC sorgente                     | -  | Frame inviata attraverso DS ad un AP all'interno della cella che possiede BSSID dell'address 2 |
| 1     | 0       | <b>BSSID</b><br>della cella destinazione                     | <b>SA</b><br>Indirizzo MAC sorgente                    | <b>DA</b><br>Indirizzo MAC stazione che sta ricevendo   | -  | Frame inviata attraverso DS ad un AP di una cella diversa BSSID dell'address 1                 |
| 1     | 1       | <b>RA</b><br>Indirizzo AP destinazione all'interno di DS     | <b>TA</b><br>Indirizzo AP sorgente all'interno di DS   | <b>DA</b><br>Indirizzo della stazione che sta ricevendo | <b>SA</b><br>indirizzo della stazione che sta inviando | Frame tra AP di celle differenti usando DS   |

Le ultime 3 righe fanno riferimento alla casistica in cui sono presenti più celle e serve routing tra di queste. I casi sono:

- 01 From DS, verso un AP all'interno della cella, letto nell'indirizzo 2
- 10 Verso il DS, indico cella e indirizzo di destinazione
- 11 Da e Verso il DS, devo sapere da dove arriva e dove inviare, oltre che indirizzo originale e finale (routing tra celle)

#### 4.3 Orthogonal Frequency Division Multiple Access OFDMA

Si parla di WiFi 6, il quale utilizza OFDMA, sfrutta **subcarrier per fornire connettività a più dispositivi contemporaneamente**. Prima tutte le sottoportanti erano usate per un dispositivo alla volta (OFDM). Con OFDMA c'è la possibilità di fornire gruppi di canali diversi a dispositivi diversi.

Usando OFDM con TDMA potrebbe esserci una situazione tipo

|    | T1             | T2             | T3             | T4  | T5             |
|----|----------------|----------------|----------------|-----|----------------|
| F1 | Blue           | Green          | Yellow         | Red | Purple         |
| F2 | Blue           | Green          | Yellow         | Red | Purple         |
| F3 | Light Blue     | Green          | Yellow         | Red | Purple         |
| F4 | Non necessario | Non necessario | Non necessario | Red | Non necessario |
| F5 |                | Non necessario |                | Red |                |

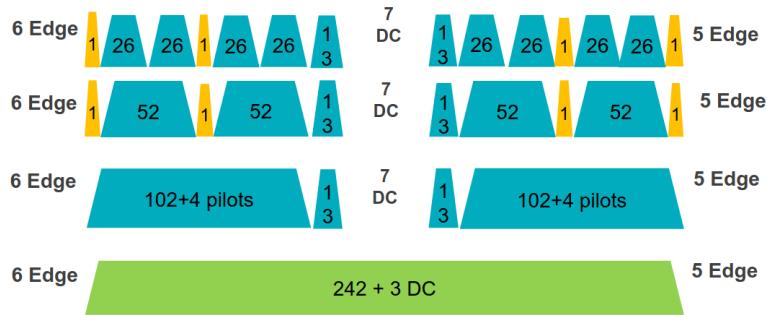
E dato che utenti diversi avranno necessità diverse questo potrebbe portare a "sprecare" frequenze. Con OFDMA si può avere un caso tipo

|    | T1    | T2  | T3     | T4    | T5  |
|----|-------|-----|--------|-------|-----|
| F1 | Blue  | Red | Yellow |       | Red |
| F2 | Blue  | Red | Purple | Green | Red |
| F3 | Green | Red | Purple | Green | Red |
| F4 | Green | Red | Blue   | Green | Red |
| F5 | Green | Red | Blue   |       | Red |

Vengono serviti **più dispositivi allo stesso momento**, gruppi di sottoportanti a dispositivi diversi. Facendo questo si complica lo scheduling, al posto di assegnare solo lo slot di tempo bisogna assegnare scheduling e gruppo di frequenze.

Serve definire quanto e in quale modo le frequenze possono essere raggruppate. **Resource Unit RU**: sono il gruppo di frequenze (solitamente adiacenti) allocabili ad un utente. Assegniamo “blocchi” di risorse ad ogni utente. La dimensione delle RU è variabile e dipende dalla banda disponibile e come l’AP vuole allocare le risorse agli utenti.

Le possibili RU per una banda di 20MHz divisa in sottoportanti da 78.125kHz sono 256 e si possono dividere come:



Non tutta la banda viene usata, sono presenti intervalli di guardia. Alcune sotto-portanti sono usate come **pilot**, trasmettono un’onda definita nello standard, in modo da poter stimare la qualità del canale e capire come correggere il segnale. Vengono ripetuti “ogni tanto” per rifare la misura.

L’AP deve usare frame di controllo: alcuni nuovi con 802.11ax per supportare le nuove funzionalità, altri già presenti nelle precedenti versioni.

Le informazioni di allocazione delle RU sono usate da PHY e MAC e vengono inviate. Ogni risorsa è identificata da una sequenza unica di 7 bit, ad esempio:

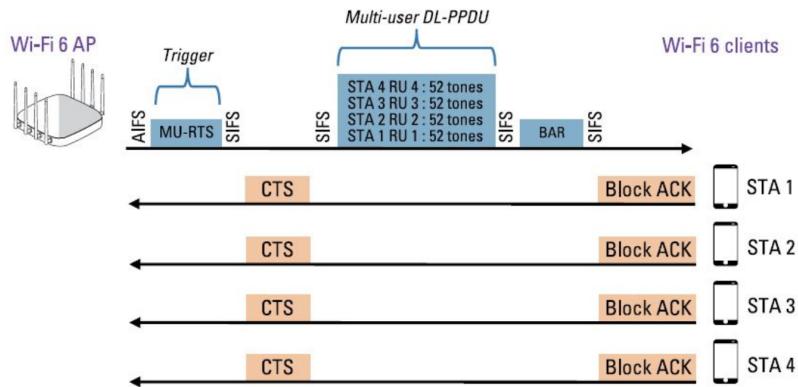
| 26 tone RU         | <b>RU-1</b> | <b>RU-2</b> | <b>RU-3</b>    | <b>RU-4</b> | <b>RU-5</b>  | <b>RU-6</b> | <b>RU-7</b> | <b>RU-8</b> | <b>RU-9</b> |
|--------------------|-------------|-------------|----------------|-------------|--------------|-------------|-------------|-------------|-------------|
| Subcarrier range   | -121:-96    | -95:-70     | -68:-43        | -42:-17     | -16:-4, 4:16 | 17:42       | 43:68       | 70:95       | 96:121      |
| RU allocation bits | 0000000     | 0000001     | 0000010        | 0000011     | 0000100      | 0000101     | 0000110     | 0000111     | 0001000     |
|                    |             |             |                |             |              |             |             |             |             |
| 52 tone RU         | <b>RU-1</b> | <b>RU-2</b> |                |             | <b>RU-3</b>  | <b>RU-4</b> |             |             |             |
| Subcarrier range   | -121:-70    |             | -68:-17        |             |              | 17:68       |             | 70:121      |             |
| RU allocation bits | 0100101     |             | 0100110        |             |              | 0100111     |             | 0101000     |             |
|                    |             |             |                |             |              |             |             |             |             |
| 106 tone RU        | <b>RU-1</b> |             |                |             | <b>RU-2</b>  |             |             |             |             |
| Subcarrier range   |             | -122:-17    |                |             |              | 17:122      |             |             |             |
| RU allocation bits |             | 0110101     |                |             |              | 0110110     |             |             |             |
|                    |             |             |                |             |              |             |             |             |             |
| 242 tone RU        |             | <b>RU-1</b> |                |             |              |             |             |             |             |
| Subcarrier range   |             |             | -122:-2, 2:122 |             |              |             |             |             |             |
| RU allocation bits |             |             | 0111101        |             |              |             |             |             |             |

Ciascun codice rappresenta un insieme di sottoportanti ed il numero indica il range di sottoportanti usate da quella RU. I codici sono tutti diversi ed ognuno identifica univocamente una RU, tale codice viene usato per indicare al dispositivo che frequenze utilizzare.

Una volta assegnata una risorsa, quelle sopra (nella tabella) non possono essere utilizzate (hanno risorse in comune, e per forza di cose gli insiemi di risorse devono essere mutualmente esclusivi); per il resto si può fare mix and match come è più comodo (non si è limitati ad usare lo stesso “livello” di RU).

#### 4.3.1 Downlink DL-OFDMA

L'AP ha da trasmettere e conosce la lista dei destinatari, deve inoltre comunicare l'assegnamento delle risorse.



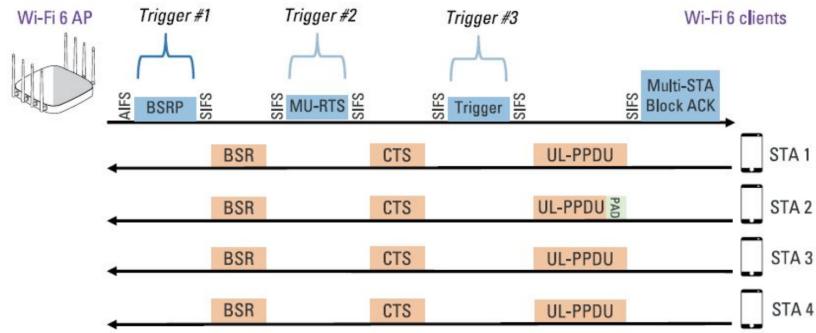
Viene usata una **Multi-User Request to Send MU-RTS**, un messaggio di controllo che ha il funzionamento di RTS e assegnamento delle risorse. Stiamo dando la possibilità di trasmettere alle stazioni designate all'interno del messaggio, allo stesso tempo indicando alle altre stazioni di allocare il NAV. Le stazioni sapranno le risorse a loro dedicate e rispondono con una CTS, tutte in contemporanea (le frequenze sono ortogonali, non collidono). Per la risposta, ogni stazione ascolta solo la porzione di canale a lei associata.

Dopo il termine dei frame i dispositivi non mandano subito l'ack, ma l'AP aspetta SIFS e invia un **Block Acknowledgement Request BAR** ed i dispositivi rispondono in parallelo con un **Block ACK**.

Se gli ack fossero immediatamente al termine della trasmissione per ogni dispositivo l'ack potrebbe andare perso se l'AP non è ancora in modalità di ricezione.

### 4.3.2 Uplink UP-OFDMA

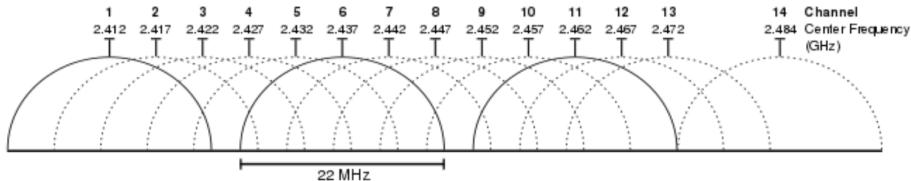
L'uplink è *meno prevedibile* e bisogna sincronizzare tutti i dispositivi che vogliamo trasmettano contemporaneamente. La trasmissione deve avvenire in modo sincronizzato, dopo aver comunicato ad ogni stazione la risorsa su cui trasmettere; tutto coordinato dall'AP.



Ci sono più step:

- **Buffer Status Report Poll BSRP:** l'AP chiede chi ha dati da trasmettere, in parallelo; interroga un sottogruppo di stazioni per chiedere chi ha da dire qualcosa
- I dispositivi rispondono con un **Buffer Status Report BSR:** chi ha da trasmettere risponde con quanto ogni dispositivo deve trasmettere (per l'allocamento delle risorse)
- l'AP **assegna risorse** in base alle risposte delle stazioni ed invia il **MU-RTS**, indicando la divisione delle risorse
- Le stazioni rispondono con un **CTS**
- Poi c'è un **trigger aggiuntivo** da parte dell'AP, per far sincronizzare i dispositivi ; tutte le stazioni devono trasmettere nello stesso momento (ad esempio, chi è più lontano comincia prima)
- **Ogni stazione trasmette in parallelo** sulle risorse che gli sono state allocate; chi trasmette meno fa padding
- Infine, se necessario, vengono inviati gli ack; **multi-station block acknowledgment Multi-STA Block ack** a tutte le stazioni che ne hanno fatto richiesta

In banda  $2.4GHz$  sono presenti **14 canali**, ma allo stesso tempo **non sovrapposti** possono essercene solo **3**.



Se tutti usassero la stessa sarebbe un casino.

Per la banda  $5GHz$  la configurazione è

| 5 GHz Channel Allocations |  |   |  |                                      |                 |              |      |      |                    |      |      |      |       |      |      |      |       | UNII-3 |      |      |        | UNII-3 |      |      |       |  |  |  |  |  |  |
|---------------------------|--|---|--|--------------------------------------|-----------------|--------------|------|------|--------------------|------|------|------|-------|------|------|------|-------|--------|------|------|--------|--------|------|------|-------|--|--|--|--|--|--|
| Frequency (GHz)           | 5.150  |   |  |                                      | 5.250           |              |      |      | 5.470              |      |      |      | 5.600 |      |      |      | 5.640 |        |      |      | 5.725  |        |      |      | 5.850 |  |  |  |  |  |  |
| 802.11 Allocations        | UNII-1   |   |  |                                      | UNII-2a         |              |      |      | UNII-2c (Extended) |      |      |      | TDWR  |      |      |      |       |        |      |      | UNII-3 |        |      |      |       |  |  |  |  |  |  |
| Center Frequency          | 5180   | 5200  | 5220   | 5240                                 | 5260            | 5280         | 5300 | 5320 | 5500               | 5520 | 5540 | 5560 | 5580  | 5600 | 5620 | 5640 | 5660  | 5680   | 5700 | 5720 | 5740   | 5760   | 5780 | 5800 | 5825  |  |  |  |  |  |  |
| 20 MHz                    | 36   | 40  | 44   | 48                                   | 52              | 56           | 60   | 64   | 100                | 104  | 108  | 112  | 116   | 120  | 124  | 128  | 132   | 136    | 140  | 144  | 149    | 153    | 157  | 161  | 165   |  |  |  |  |  |  |
| 40 MHz                    | 38   | 46  | 54   | 62                                   | 102             | 110          | 118  | 126  | 134                | 142  | 151  | 159  |       |      |      |      |       |        |      |      |        |        |      |      |       |  |  |  |  |  |  |
| 80 MHz                    | 42   | 58  | 106  | 122                                  | 138             | 155          |      |      |                    |      |      |      |       |      |      |      |       |        |      |      |        |        |      |      |       |  |  |  |  |  |  |
| 160 MHz                   | 50   | 114   | 120, 124, 128                                    | Devices Now Allowed                  | 144 Now Allowed | DFS Channels |      |      |                    |      |      |      |       |      |      |      |       |        |      |      |        |        |      |      |       |  |  |  |  |  |  |
| FCC                       | 1,000 mW Tx Power<br>Indoor & Outdoor<br>No DFS needed | 250 mw w/6dBi<br>Indoor & Outdoor<br>DFS Required | 250mw w/6dBi<br>Indoor & Outdoor<br>DFS Required | 120, 124, 128<br>Devices Now Allowed | 144 Now Allowed | DFS Channels |      |      |                    |      |      |      |       |      |      |      |       |        |      |      |        |        |      |      |       |  |  |  |  |  |  |
| DFS Channels              |  |   |  |                                      |                 |              |      |      |                    |      |      |      |       |      |      |      |       |        |      |      |        |        |      |      |       |  |  |  |  |  |  |

Per ogni ampiezza di banda si ha una serie di range possibili, ognuno con il proprio identificatore. Nel momento in cui un dispositivo si deve connettere, all'interno del beacon si trova l'informazione sul canale utilizzato. All'interno di questi canali ci sono le sottoportanti.

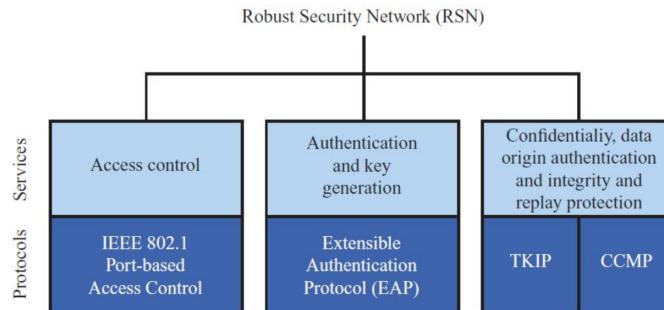
## 4.4 WLAN Security

All'interno di 802.11 sono definite delle feature per la sicurezza. Per definizione il canale radio è **molto** esposto, tutti possono ascoltare/inviare su un canale intrinsecamente broadcast. Di conseguenza si ha la necessità di cifrare il canale anche a livello di data link.

**Wired Equivalent Privacy WEP:** Caratteristiche:

- Algoritmo di cifratura RC4
- Opzionale, non è obbligatorio attivarlo
- Assenza di un sistema di gestione delle chiavi, tutto il traffico era cifrato allo stesso modo (la chiave non è la password wi-fi)
- Tutto il traffico viene cifrato con la stessa chiave

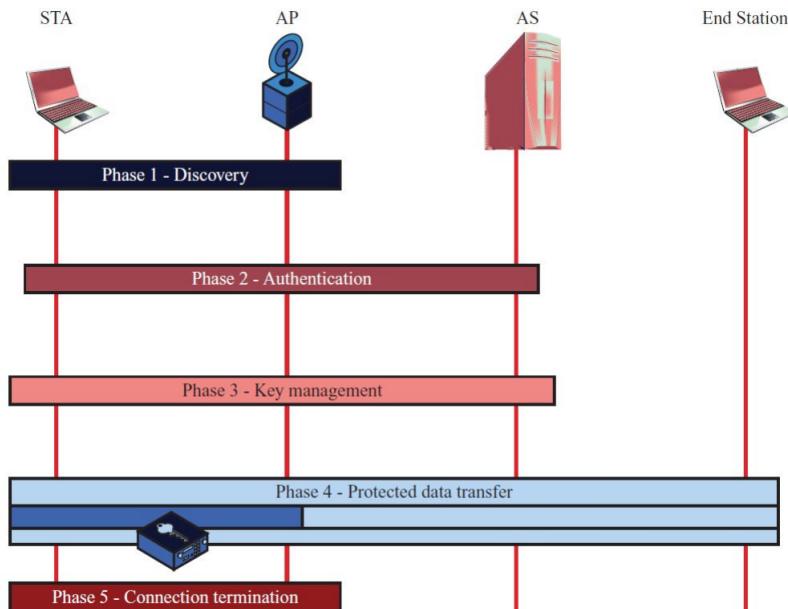
**Robust Security Network RSN:** Per sopperire alle lacune di WEP è stato introdotto un emendamento allo standard, 802.11i. Viene definito il RSN, con al suo interno diversi servizi:



- **Access control:** impone l'utilizzo di protocolli di sicurezza e assiste lo scambio delle chiavi
- **Authentication:** definisce lo scambio tra utente e Authentication Server AS e genera le chiavi temporanee per la comunicazione sul canale radio
- **Privacy with message integrity:** il payload MAC (LLC PDU) viene cifrato (di più non posso altrimenti non so nemmeno per chi sia il messaggio, l'header MAC deve essere in chiaro) e viene aggiunto al messaggio una parte per il controllo dell'integrità

Ci sono più fasi per le operazioni:

- **Discovery:** ovviamente il beacon non è cifrato
- **Authentication:** si evidenziano AP e Authentication Server, il quale può coincidere fisicamente con l'AP o meno.
- **Key management:** generazione delle chiavi specifiche per il singolo dispositivo
- **Protected data transfer:** una volta ottenuta la chiave, si comincia la comunicazione cifrata
- **Chiusura** della connessione



Nella fase di **discovery**:

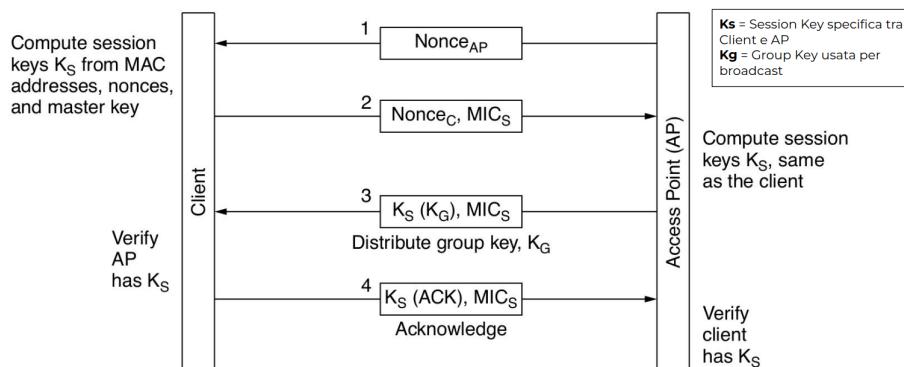
- AP manda il beacon, il quale fornisce anche i servizi RSN disponibili (la modalità di accesso alla cella)
- La station STA ascolta il beacon e capisce quali sono i servizi RSN che può utilizzare (negoziano le capabilities di ognuno, si decide la policy da utilizzare)
- Associazione AP e STA, arrivano ad un accordo sulle funzionalità di sicurezza da utilizzare. Può anche essere negata, da entrambe le parti

Durante la fase di **autenticazione e gestione chiavi**:

- STA richiede ad AP autenticazione tramite la comunicazione con un Authentication Server AS
- Il server può essere remoto in caso di utilizzo di Extensible Authentication Protocol EAP
- Consegna delle chiavi in modo sicuro e generazione delle chiavi: viene generata una master key (a partire dalla password del WiFi) che viene usata per generare le altre chiavi crittografiche, richieste in seguito (lo standard non definisce come avviene lo scambio, si affida a EAP)

Come vengono **generate le chiavi**:

- viene inviato un nonce da AP a Client
- la chiave di sessione  $K_S$  viene calcolata dal Client a partire dai MAC address, i nonce e la master key (generata a partire dalla password della rete)
- il client invia il suo nonce, assieme ad un  $MIC_S$ , ovvero Message Integrity Check, dipendente dalla sessione (quindi dalla session key, in parte)
- l'AP può calcolare la stessa  $K_S$  del Client
- L'AP invia la chiave di gruppo  $K_g$ , cifrandola tramite la chiave  $K_S$
- Il Client verifica che l'AP ha la  $K_S$  calcolata in modo corretto
- Il Client risponde con un ack, cifrato con la  $K_S$
- L'AP verifica che la chiave del Client sia corretta



**Protezione dati:** Lo standard 802.11i considera 2 alternative:

- **TKIP (WPA)**: aggiunge un codice di integrità 64 bit usando MAC di sorgente e destinazione; per la confidenzialità viene usato RC4; cambiamenti solo software rispetto a WEP
- **CCMP (WPA-2)**: integrità della cifratura tramite cipher-block-chaining; integrità e confidenzialità tramite AES 128 bit

Tra i due cambia l'algoritmo di cifratura, per il primo non serviva cambiare nulla a livello hardware, per il secondo serve il supporto a Cipher-Block-Chaining CBC con AES.

Per la sicurezza: il segreto sta nella master key (il resto sarebbe pubblicamente accessibile), il presupposto è che l'attaccante non ne sia a conoscenza. Nelle varie versioni di EAP cambia il metodo con cui la chiave viene generata, per mantenere valido questo presupposto. Attacchi di tipo MitM vengono evitati grazie ad integrity check e nonce.

## 4.5 WiFi Protected Setup WPS

Viene usato per evitare di mettere la password, su alcuni dispositivi in cui può essere comodo. All'interno del protocollo ci sono 3 tipi di dispositivi:

1. **Registrar**: entità che autorizza o revoca una stazione (AP o esterno)
2. **AP**
3. **Enrollee**: la stazione che vuole accedere

Le **modalità di attivazione** sono in-band, due opzioni:

- **PIN**: può essere dell'AP da immettere sul dispositivo Enrollee, oppure dell'Enrollee da immettere sull'AP
- **Push Button**: pressione di un bottone su AP e Enrollee, la procedura rimane attiva per massimo 2 minuti. Associazione FIFO (primo che entra è autenticato)

## 4.6 802.11e EDCA Enhanced Distributed Channel Access

Altro emendamento allo standard, per aggiungere EDCA. Quando i livelli superiori richiedono un certo livello di qualità viene migliorato la distribuzione dell'accesso al canale, modificando i parametri:

- **CWMin**: Minima dimensione della contention window (numero random da aspettare)
- **CWMax**: massimo dimensione della contention window
- **AIFSN**: numero di SIFS+N slot time; definisce la lunghezza del tempo di attesa; il +2 è il minimo, equivalente a DIFS in totale (per definizione di DIFS)
- **Max TXPO**: massimo tempo nel quale una stazione può trasmettere più frame senza rilasciare il canale; quanto tempo la stazione può tenere il canale (0 vuol dire senza limite)

| Access Category     | CWmin | CWmax | AIFSN | Max TXOP |
|---------------------|-------|-------|-------|----------|
| Background (AC_BK)  | 15    | 1023  | 7     | 0        |
| Best Effort (AC_BE) | 15    | 1023  | 3     | 0        |
| Video (AC_VI)       | 7     | 15    | 2     | 3.008ms  |
| Voice (AC_VO)       | 3     | 7     | 2     | 1.504ms  |
| Legacy DCF          | 15    | 1023  | 2     | 0        |

Si può vedere come per lo streaming video, e ancora di più per la voce, che la contention window viene ridotta, così come il tempo di attesa, ma il tempo massimo in cui possono occupare il canale è limitato. Si può notare inoltre come le contention window siano mutualmente esclusive, effettivamente creando un “ordine di priorità” voce, video, poi il resto.

Si tratta di una definizione più granulare delle politiche di accesso al canale in contesa, permettendo di ridurre i tempi per le applicazioni che ne necessitano.

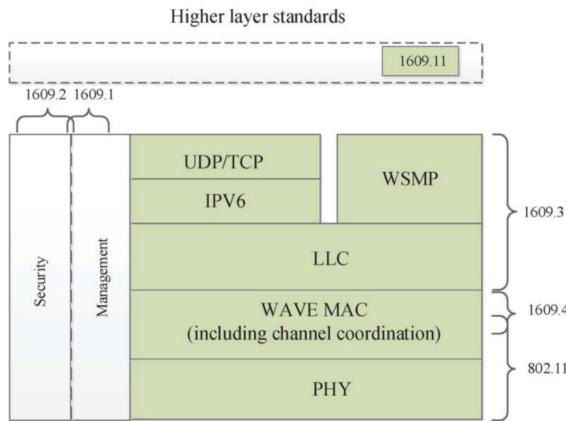
## 4.7 802.11p & Wave - V2V

Si tratta di un WiFi “veicolare”, rientra nelle **Dedicated Short-Range Communications DSRC** (con la banda dedicata  $5.9GHz$ ). Serve per la comunicazione ad-hoc tra veicoli usando banda unlicensed.

Si tratta di una soluzione specifica per il caso specifico:

- Rete *altamente* dinamica
- Supporto alla guida autonoma (alert, dati sensori, ecc.; scambiati tra veicoli)
- Associazione veloce (rete molto dinamica, non c'è troppo tempo)
- Supporto per servizi critici, come collisione (20ms), pedaggi (50ms), servizi a bordo strada (500ms), quindi con tempi anche molto ridotti
- Supporto infotainment

**Wireless Access for Vehicular Environment WAVE:** Si tratta di un protocollo specifico per la situazione descritta, include una parte “normale” per il trasferimento di dati (TCP e famiglia), ma è presente anche una parte dedicata ai servizi necessari nell’ambito veicolare.



Tutto lo stack si poggia sullo stesso standard IEEE (livello MAC e PHY), quindi viene usato l'emendamento 802.11e, che permette di gestire meglio diverse priorità a livello MAC.

Il canale fisico (802.11p) usa la banda  $5.9GHz$  e permette diverse modulazioni e coding rate in base al range e quantità di informazioni richieste. Solitamente viene usato *QPSK1/2*.

L’associazione deve essere molto veloce, quindi non sono previsti RTS e CTS.

## 5 Ad Hoc Distance Vector Protocol AODV

Ambito WLAN. Si tratta di un **protocollo di routing**: ha il compito di **creare e riempire le tabelle di instradamento**. Pensato per una rete **senza infrastrutture** in cui ogni nodo è anche un router (può fare instradamento). I cammini possono essere multi-hop. I nodi vicini sono quelli nel raggio radio, di conseguenza nodi e vicinati possono variare.

I percorsi non vengono creati in modo proattivo: quando serve mandare qualcosa viene richiesto il percorso.

Gli **obiettivi** principali del protocollo sono:

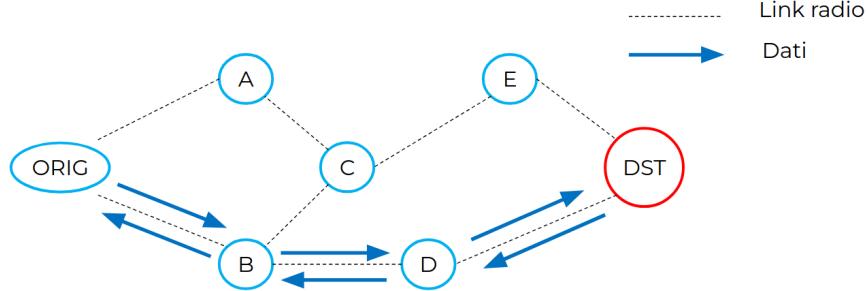
- Gestione dinamica della rete ad hoc, ogni nodo può operare secondo protocollo
- Auto inizializzante, non sono necessarie rotte preconfigurate; l'inizializzazione deve essere fatta in automatico e periodicamente (per "colpa" della dinamicità della rete)
- Loop-free (eliminando il problema del counting to infinity)
- Ottenimento di una rota per una nuova destinazione in tempi rapidi
- Risposta rapida alla rottura dei link e al cambio di topologia

Le **funzionalità** che offre sono:

- Scoprire e costruire i percorsi per le nuove destinazioni
- Mantenere i percorsi in modalità soft-state; ogni percorso ha una durata, se non "rinfrescata" la entry scade e può essere cancellata (eventually)
- Riconoscimento errori e cancellazione di percorsi; monitoraggio dell'attività locale e propagazione delle informazioni

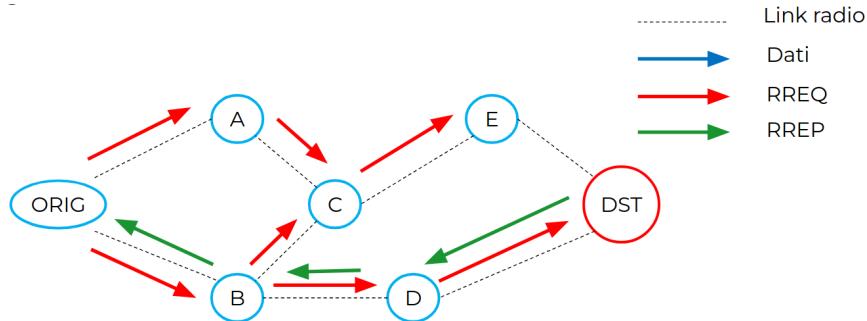
Si tratta di un protocollo a livello di applicazione (usa UPD 654) che permette di creare tabelle di routing a livello di rete. Ogni nodo è responsabile delle proprie entry, il protocollo è completamente distribuito.

I dati vengono inviati tra originator e destinazione tramite percorsi simmetrici, non ci sono percorsi indipendenti.



**Route Request:** I percorsi vengono costruiti tramite **Route Request RREQ**, sono pacchetti di controllo che permettono di chiedere la costruzione di un percorso verso la destinazione. Non sapendo niente, i messaggi RREQ vengono inviati in broadcast “controllato” (senza loop, il broadcast è fatto a livello IP, con l’indirizzo IP di broadcast). Ogni nodo inoltra RREQ e tiene traccia della provenienza.

**Route Reply:** La destinazione richiesta in una RREQ risponde in unicast con un messaggio **Route Reply RREP**.



La destinazione manda la risposta RREP solo alla prima RREQ (con lo stesso identificatore) ricevuta, in modo da evitare risposte duplicate e trovando (circa) il percorso più breve, dato che risponderà solo al messaggio che ci ha impiegato meno tempo (il primo arrivato).

Anche un nodo intermedio può fornire la risposta (inviare il RREP), se conosce l’informazione (sa come arrivare a destinazione) ed è abbastanza

“fresca” (lo ha saputo di recente).

**Route Error:** Un altro messaggio di controllo è **Route Error RERR**, usato nel caso un link si rompe, per qualsiasi motivo. Il nodo lo invia a tutti i vicini che sa utilizzano quel link per arrivare ad una qualche destinazione.

Lo stack è:

|                        | Messaggi Dati                              | RREQ/RREP/RERR |
|------------------------|--|----------------|
| Livello applicazione   | Protocollo applicazione                    | AODV           |
| Livello Trasporto      | Trasporto UPD/TCP/...                      | UPD Porta 654  |
| Livello Rete           | IP   |                |
| Livello data link (LL) | Ethernet (802.3)/WiFi(802.11)/802.15.4/... |                |
| Livello fisico (PHY)   | Ethernet (802.3)/WiFi(802.11)/802.15.4/... |                |

Quindi i messaggi possono essere dati “normali” tramite TCP oppure i messaggi di controllo AODV. Il livello di rete è IP, mentre i livelli di data link e fisico possono essere uno qualsiasi degli standard IEEE previsti.

## 5.1 Tabelle di Routing

Ogni nodo mantiene una tabella delle destinazioni conosciute e l’indicazione del prossimo hop lungo il percorso. Ogni entry in una tabella contiene:

- IP destinazione
- Sequence number della destinazione
- Flag di validità del sequence number della destinazione; permette di disabilitare/abilitare temporaneamente un percorso
- Stato del percorso (valido, invalido e altri)
- Interfaccia di rete
- Hop count, numero di hop per arrivare alla destinazione, ovvero il costo

- Lista dei precursori, quali sono i vicini che utilizzano “me” (il nodo) per arrivare a destinazione
- Lifetime della entry, ovvero il tempo di scadenza

**Sequence Number SN:** Tutto cambia in questa rete, quindi ogni entry della tabella possiede un SN che codifica informazioni circa “*la freschezza*” della entry stessa. Il SN è un valore per ogni nodo, ogni possibile destinazione (nodo) ha il proprio e viene modificato esclusivamente dal nodo stesso.

Il SN viene incrementato dal nodo in 2 casi:

1. Quando un nodo inizia una ricerca di percorso (RREQ), viene incrementato di 1, previene conflitti con i percorsi inversi stabiliti da una precedente RREQ
2. Quando un nodo risponde (ovvero il nodo è la destinazione) ad una richiesta di percorso (manda RREP); si aumenta di 1 (solo in alcuni casi in realtà)

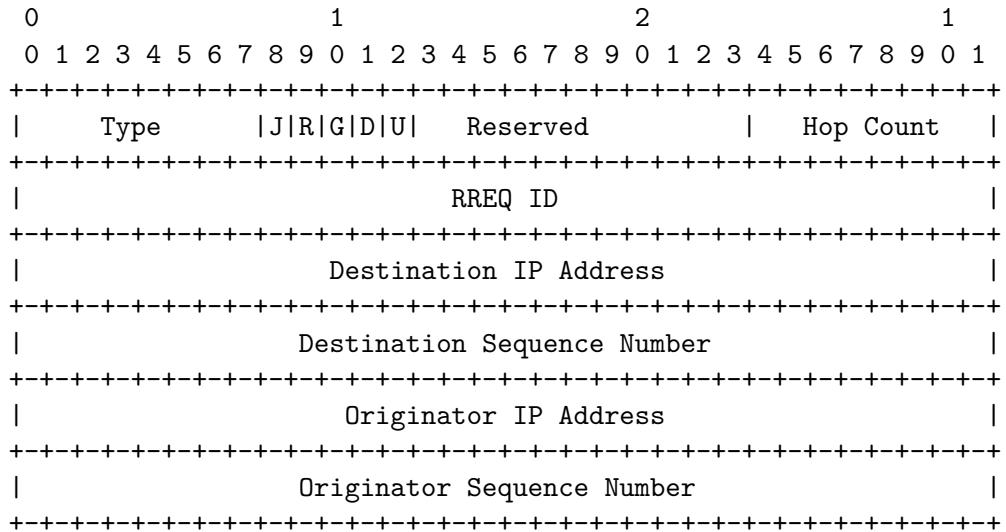
Gli altri nodi possono aggiornare il sequence number di una entry in tabella se:

- Il nodo stesso offre un nuovo percorso per se stesso (aggiorna la propria entry nella sua tabella di routing)
- Il nodo riceve informazioni più aggiornate per un destinazione
- Il percorso verso quella destinazione è scaduto/interrotto

Il SN viene confrontato per capire *chi ha l'informazione più aggiornata* per la destinazione. L’incremento viene fatto come se fosse unsigned, il confronto come signed (per gestire overflow).

## 5.2 RREQ

**Formato:** Il formato di una RREQ è



Il **tipo** è 1, rappresenta il tipo di pacchetto di controllo inviato (RREQ in questo caso).

Le **flag** sono:

- **D:** Destination Only, solo la destinazione può rispondere, niente nodi intermedi
- **U:** Unknown SN, l'origine non conosce SN della destinazione
- **G:** (Gratuitous) RREP, un nodo intermedio, oltre a rispondere all'origine, deve informare la destinazione della creazione di un percorso (reverse) con l'origine

L'**hop count** viene incrementato di 1 ogni volta che viene inoltrata la RREQ, serve a tenere traccia del “costo” del percorso, in realtà la destinazione vuole rispondere alla RREQ con hop count minore.

Il **RREQ ID** è l'identificativo della richiesta, generato dall'origine e mai modificato, serve a riconoscere la richiesta e le sue copie come un'unica richiesta all'interno della rete.

Il **Destination IP Address** è l'indirizzo della destinazione ed il SN indicato è l'ultimo conosciuto (o nessuno nel caso il flag U sia alzato). Se un nodo intermedio ha informazioni riguardo alla destinazione, ma con SN inferiore rispetto a quello indicato, sicuramente l'informazione è troppo vecchia quindi non la invia.

**Originator IP Address** e **SN** sono le informazioni riguardo l'originator, con il SN più recente inviato.

**RREQ Creation:** Una RREQ viene creata *quando serve*, ovvero quando un nodo non conosce la destinazione o se la entry è scaduta. Per inviare:

- Incremento RREQ ID e il proprio SN (dell'originator)
- Se la destinazione è sconosciuta, flag U= 1
- Mantiene una copia di Origine IP, RREQ ID per un tempo denominato **PATH\_DISCOVERY\_TIME**, definito nello standard, per evitare il riprocessamento e il reinvio dei pacchetto che può essere ricevuto dai vicini

### 5.2.1 Expanding Ring Search

Vogliamo evitare di propagare “inutilmente” una RREQ in tutta la rete, magari la destinazione è vicina. Il TTL dell'header IP viene usato per impostare il massimo numero di hop che una RREQ può fare.

Si hanno dei parametri:

- **TTL\_START**: parametro TTL per la prima richiesta
- **TTL\_INCREMENT**: incremento del TTL ad ogni tentativo
- **NET\_DIAMETER**: massimo valore del TTL

La request viene fatta inizialmente con un TTL basso, se dopo una certa quantità di tempo il nodo non riceve risposta viene aumentato il TTL ed effettuata una nuova richiesta (“bah, magari è più lontana, ci riprovo”). Ripete fino al TTL massimo. Si espande “l'anello” in cui cercare la destinazione.

Questo era *senza conoscenze a priori*, se sono presenti delle conoscenze riguardo un percorso verso la destinazione (scaduto o interrotto) uso la distanza nota in precedenza per iniziare a inviare RREQ. Se ho un'informazione vecchia: parto dalla distanza “vecchia” come TTL. Questo è uno dei motivi per cui non “buttare” subito le informazioni vecchie, vengono tenute per “*un po'*” (definito dallo standard).

**Retry Policy:** L'origine può riprovare ad inviare RREQ se il primo tentativo non è andato a buon fine. **RREQ\_RETRIES** è il parametro. Ad ogni nuovo tentativo si incrementano RREQ\_ID e SN.

### 5.2.2 RREQ Processamento e Inoltro

Quando un nodo riceve un messaggio RREQ:

- Controlla REQ\_ID e Originator IP sono già presenti (già conosciute), se uguali all'ultimo ricevuto entro PATH\_DISCOVERY\_TIME scarto il messaggio
- Aggiorno il percorso “reverse”, verso originator; ricevendo una RREQ posso capire come arrivare all'origine:
  1. Confronto Orig SN (in RREQ) e SN che ho nella tabella, se è maggiore lo aggiorno (potrebbe essere minore, magari la richiesta è molto vecchia, ha fatto giri strani)
  2. Segno la entry come valida (può essere scaduta nel frattempo)
  3. Aggiorno/aggiungo la entry impostando come “next hop” verso orig, il nodo da cui è arrivata la RREQ
  4. Il campo Hop Count della entry viene messo pari al hop count della RREQ (quanto ci ha messo ad arrivare al nodo)

Se il nodo intermedio non può rispondere con RREP (anche per flag D) allora deve inoltrare RREQ ai vicini. Per farlo, modifica il messaggio RREQ:

- incrementa di 1 Hop Count
- il SN della DST (in RREQ) deve essere il massimo tra quello nella

richiesta e il SN nella routing table (la richiesta è “vecchia”, ho informazioni più nuove)

- manda la RREQ in broadcast (indirizzo IP 255.255.255.255)

In questo modo, il propagarsi di una richiesta permette ai nodi che la ricevono di costruire un percorso verso l'orig.

### 5.3 RREP

## Formato:

Qua il tipo è 2.

Flag:

- A: acknowledgment richiesto in risposta per prevenire link non affidabili

Il **prefix size** viene utilizzato per subnet, indica la lunghezza del prefisso di rete dell'indirizzo IP di destinazione (in bit).

Il destination IP Address indica chi ha generato la risposta.

Il **destination Sequence Number** si riferisce al SN più aggiornato della destinazione.

L'originator IP Address indica a chi sta rispondendo questa richiesta.

**Lifetime:** determinato da chi crea RREP, millisencondi di validità della risposta.

**Creazione:** Chi può generare una RREP?

1. La destinazione

- Incrementa il suo SN prima di inviare
- Hop count = 0
- Aggiorna la lista dei precursori (chi usa la destinazione stessa come hop)
- Imposta il campo lifetime MY-ROUTE-TIMEOUT (default 6s)
- Invia RREP lungo il percorso reverse (unicast)
- Effettua il drop della RREQ (non viene più inoltrata)

2. Un nodo intermedio: può farlo, ma con delle condizioni:

- (a) Avere una entry con un percorso valido per la destinazione
- (b) Flag D == 0
- (c) DST SN della entry  $\geq$  DST SN della RREQ

Se tutte soddisfatte, allora:

- Hop count = valore Hop count della entry (distanza dalla destinazione, al posto dello 0 di prima)
- Aggiorna la lista dei precursori (il nodo a cui sto inviando la RREP userà "me" come hop per arrivare alla destinazione)
- Imposta il campo lifetime in base a quello che ho nella entry (il tempo che rimane nella entry)
- Invio RREP lungo il percorso reverse in unicast
- Drop della RREQ
- Se il flag G == 1 allora invio RREP anche alla destinazione (vedi dopo)

### 5.3.1 RREP Processamento e Inoltro

Quando un nodo riceve un messaggio RREP:

- Aggiorna la entry nella tabella se
  - La entry corrente non è valida
  - DST SN della RREP è maggiore di quello della entry
  - Il numero di hop è minore rispetto a quello della entry
- Aggiorna
  - Entry marcata come valida
  - Next hop della entry = nodo da cui proviene RREP (destinazione della RREQ)
  - Aggiorno RREP hop count, incremento di 1
  - Aggiorno campo lifetime della entry
  - Aggiorno la lista dei precursori (chi sta usando “me” per raggiungere l’originator)

Mentre torna la RREP, ogni nodo intermedio crea il path da sé stesso alla destinazione originale della RREQ. Ma il percorso “bidirezionale” viene creato solo nella porzione di rete tra il nodo intermedio e l’originator, non anche verso la destinazione.

Dopo la RREP di un nodo intermedio, dato che la RREQ originale era in broadcast, anche la sorgente (eventually) invierà la sua RREP. Entrambe le RREP torneranno all’originator, probabilmente quella della destinazione arriverà dopo, in qualsiasi caso la migliore viene tenuta (SN maggiore e Hop count minore).

**RREP a RREQ con flag Gratuitous:** Se un nodo intermedio riceve una RREQ con flag gratuitous attivo, il nodo deve occuparsi di “costruire” il rimanente percorso verso la destinazione della RREQ.

Il nodo quindi invia 2 RREP indipendenti

1. Verso il nodo Origine (che ha creato la RREQ)

2. Verso la destinazione della RREQ (Gratuitous), con

- Hop count = numero hop nella entry verso origine RREQ
- Destinazione = IP origine delle RREQ
- Destinazione SN = SN nella entry verso origine RREQ
- Originator IP = DST della RREQ
- Lifetime = lifetime nella entry verso origine RREQ

Serve a “simulare” che la destinazione abbia fatto una reply, finisco di costruire il percorso tra nodo intermedio e destinazione. Mi garantisce che il nodo di destinazione e tutti quelli sul percorso conoscano la strada per arrivare anche all’originator della RREQ iniziale. Come se il nodo destinazione avesse fatto una RREQ verso l’originator (i parametri della RREP sono quelli), completando il path bidirezionale.

### 5.3.2 Hello Message

Ogni nodo può indicare informazioni riguardo la propria connettività inviando periodicamente in broadcast degli “hello message” ai propri vicini. Si tratta di un messaggio broadcast con TTL = 1 (senza incrementare il SN, lascio quello più recente).

Si tratta di una RREP speciale, con i seguenti campi:

- DST IP = IP del nodo stesso
- DST SN = SN del nodo stesso
- Hop Count = 0
- Lifetime = ALLOWED\_HELLO\_LOSS \* HELLO\_INTERVAL

Si tratta di un meccanismo facoltativo e sono usati solo se non si ricevono altri pacchetti da un vicino, qualunque pacchetto valido può essere usato per aggiornare la “vicinanza” di un nodo.

## 5.4 Mantenimento della Connettività Locale

Ogni nodo ha come compito tenere traccia della connettività con i nodi indicati come “next hop” nelle entry della tabella di routing.

Ci sono diversi meccanismi:

1. **Livello data-link:** invio pacchetto RTS/CTS/ACK, in caso di mancanza di CTS/ACK *probabilmente* il link non è più valido
2. **Livello di rete:** la ricezione di qualsiasi pacchetto dal next hop è *un'informazione* (quindi il link è attivo e posso mantenere entry), ad esempio una RREQ con destinazione il next hop, ICMP Echo unicast per il next hop (ping)

**Percorso interrotto/scaduto e cancellazione:** Quando un nodo identifica un link interrotto il quale è parte di un percorso attivo:

1. Vengono invalidati i percorsi esistenti
2. Identifica le destinazioni per le quali viene usato come next hop il link interrotto
3. Determina quali vicini possono essere affetti da questo problema (lista dei predecessori)
4. A questi vicini invia un messaggio di **Route Error RERR**

**Formato RERR:**

| 0 | 1   | 2   | 1         |
|---|---|---|-----------|
| 0 | 1   | 2   | 3         |
| 4 | 5   | 6   | 7         |
| 8 | 9   | 0   | 1         |
| 2 | 3   | 4   | 5         |
| 6 | 7   | 8   | 9         |
| 0 | 1   | 2   | 3         |
| 4 | 5   | 6   | 7         |
| 8 | 9   | 0   | 1         |
| + | +   | +   | +         |
|   | Type  | R A   | Reserved  |
|   |   |   | Prefix sz |
|   |   |   | Hop Count |
| + | +   | +   | +         |
|   |   | Unreachable Destination IP Address (1)      |           |
| + | +   | +   | +         |
|   |   | Unreachable Destination Sequence Number (1) |           |
| + | +   | +   | +         |
|   | Additional Unreachable Destination IP Addresses (if needed)     |   |           |
| + | +   | +   | +         |
|   | Additional Unreachable Destination Sequence Numbers (if needed) |   |           |
| + | +   | +   | +         |

Il tipo è 3.

L'unica **flag N** indica alla destinazione della RERR di non eliminare la entry (No Delete flag) perché il percorso è stato riparato localmente. Il nodo che invia con il flag N alzato si è reso conto che il link era rotto, ma lo ha anche riparato (non sullo stesso link, altra RREQ, non importa come), quindi chi riceve il RERR sa che c'è una soluzione e potrebbe mantenere il link, anche se potrebbe essere più lungo o simili. Insomma, si era rotto ma ci ha pensato il nodo.

Il **Dest Count** indica il numero delle destinazioni che non sono più raggiungibili contenute in questo messaggio (quanti link si sono rotti); il numero di coppie a 32 bit all'interno del messaggio.

**Quando viene inviato un messaggio RERR?** I casi sono:

1. Viene identificato un link interrotto quando il nodo deve inoltrare pacchetti DATA lungo quel link; qualcuno sta usando il nodo per arrivare alla destinazione
2. Se riceve un pacchetto DATA per una destinazione per cui non possiede una entry; chi precedeva nel percorso sta usando informazioni troppo vecchie, si tratta di un errore di rotta, non sa come arrivare alla destinazione
3. Se riceve un pacchetto RERR da un vicino per uno o più dei percorsi attivi (informazione contenuta in RERR)

RERR può essere inviato broadcast/unicast ai vicini (lista predecessori).

#### 5.4.1 RERR Processamento e Inoltro

Quando un nodo riceve una RERR

- Marca come invalide le entry delle destinazioni indicate nel RERR
- Ogni entry che viene invalidata viene preservata per un tempo **DELETE\_PERIOD**
- Inoltro RERR ai predecessori (sempre TTL=1, broadcast/unicast)

### 5.4.2 Local Repair

Se un nodo riceve un pacchetto per una destinazione lungo un percorso interrotto e la destinazione non è “troppo lontana” (dove “lontano” viene definito come  $\text{MAX-REPAIR-TTL}=0.3 \times \text{NET-DIAMETER}$ ). Il nodo che si accorge che “qualcosa non funziona”, ma magari si può trovare una soluzione. Il nodo prova a riparare il percorso per quella destinazione.

Se altre destinazioni non sono raggiungibili tramite lo stesso link, la riparazione viene fatta solo se arrivano pacchetti (solo se serve, riparazione solo reattiva non proattiva).

Per “riparare”, il nodo invia una RREQ con un TTL impostato in modo da non raggiungere la sorgente (del pacchetto DATA, l’originator della richiesta che doveva passare sul link rotto), guardando hop count della entry verso la sorgente; i risultati possibili sono:

- la procedura fallisce: manda RERR
- trova un percorso alternativo: aggiorna la propria entry

Se il percorso trovato è più lungo del precedente il nodo invia un messaggio RERR con il flag N (No Delete) impostato a 1; se più corto tutto a posto, va bene lo stesso. La sorgente deciderà autonomamente se procedere con una nuova RREQ. Sa che la destinazione è più lontana di quanto pensava, deve stabilire se il percorso più lungo è accettabile o meno (la nuova lunghezza può causare problemi? Si tratta di dati real-time?).

### 5.4.3 Reboot

Quando un nodo viene acceso per la prima volta o riavviato. Dopo il reboot un nodo può avere informazioni vecchie sullo stato della rete, quindi deve:

- Aspettare un tempo `DELETE_PERIOD` prima di trasmettere qualsiasi RREQ
- Durante questo tempo **non inoltra** alcun messaggio
- Se riceve dei pacchetti DATA per altre destinazioni risponde con RERR

## 6 Mobile Network

### 6.1 Introduzione alle reti mobili

**Pre-Cellulare:** Prima degli anni '80 esisteva un servizio di telefonia mobile con trasmettitori e ricevitori ad elevata potenza, con 25 canali e 80km di raggio di copertura. Capacità insufficiente per fornire un servizio di telefonia (voce) comparabile con i servizi di telefonia fissi.

L'idea dietro la rete cellulare invece è usare **molteplici trasmettitori** con una potenza "bassa", minore di 100W. Meno potenza, meno raggio di copertura: l'area viene divisa in celle (da qui "cellulare"), ognuna con una propria antenna (o più).

Ogni cella è servita da una **Base Station (BS)**:

- Trasmettitore
- Ricevitore
- Unità di controllo

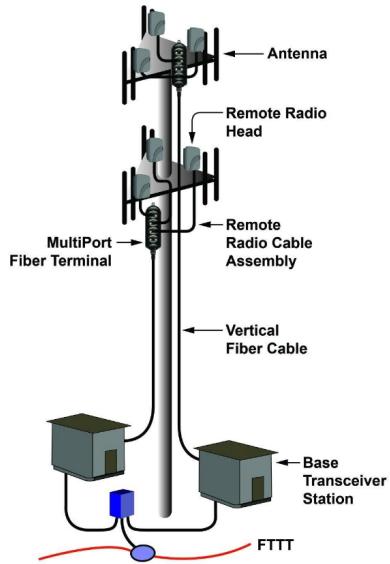
Può operare in licensed/unlicensed spectrum.

La progressione è:

- 1980 **1G Advanced Mobile Phone Service (AMPS)**: Voce analogica in mobilità
- 1990 **2G Global System for Mobile Communication (GSM)**: Voce digitale (compressa, ...), prima rete globale
- 2000 **3G Universal Mobile Telecommunications System (UMTS)**: Introduce i servizi internet
- 2010 **4G Long Term Evolution (LTE)**: convergenza IP e aumento delle prestazioni
- 2020 **5G**: Networks softwarization & virtualization, slicing & bassa latenza
- 2030 **6G**: Network intelligence (AI all'interno della rete, ottimizzazione secondo AI)

Gli standard della rete cellulare si possono vedere su **Third Generation Partnership Project** (3GPP). Sono tutte le release su cui si basano i rilasci commerciali.

**Base Station BS:** Un esempio di BS può essere



Le componenti sono:

- **Antenna:** trasmette e riceve onde radio
- **Remote radio head:** riceve i segnali analogici e li converte in digitale e viceversa; vicino all'antenna per ridurre la perdita di segnale nei cavi
- **Remote radio cable assembly:** gruppo di cavi che fornisce alimentazione e/o collegamento dati tra la stazione base e i Remote Radio Heads
- **MultiPort Fiber Terminal:** punto di terminazione o distribuzione in cui la fibra ottica in entrata viene divisa o collegata a più uscite per servire le diverse RRH. In pratica, funge da “scatola di giunzione” per organizzare e connettere i vari cavi in fibra destinati ai moduli radio remoti
- **Base Transceiver Station (BTS):** L'elemento principale dell'infrastruttura della rete mobile, gestisce il traffico dati e vocale, coordina i proto-

colli radio, si interfaccia con la rete di trasporto verso il core network dell'operatore e invia i segnali digitali agli RRH. Al suo interno si trova l'elettronica di baseband (elaborazione del segnale, modulazione/demodulazione, protocolli) e vari componenti di controllo

#### 6.1.1 Organizzazione Geometrica delle Celle

Bisogna capire come disporre le celle. I requisiti sono:

- coprire “bene” l’area
- avere una disposizione uniforme

Una buona disposizione usa celle esagonali per una copertura e disposizione uniforme. Ovviamente si tratta della disposizione ideale, nella realtà ci sono dei vincoli, di posizionamento e diffusione del segnale.

**Riuso delle frequenze:** Si ha il problema di avere celle vicine con la stessa banda di frequenza: dispositivi sui bordi ricevono dati da entrambe le celle.

La prima soluzione è usare **frequenze diverse tra celle vicine**, ma servono più bande (licensed spectrum, costa e ne uso solo una parte per volta). 2G opera in questo modo.

Un’altra soluzione, per non “sprecare” banda, è usare la stessa frequenza e **tecniche di codifica** per evitare le interferenze tra celle vicine (**CDMA**).

L’ultima soluzione è:

- al **centro** di ogni cella usare l'**intera banda** disponibile (tranne un pezzo), per gli utenti interni
- al **bordo, celle vicine** hanno **frequenze diverse**

Questo permette bandwidth maggiore per utenti interni ma richiede un sofisticato controllo di potenza e coordinamento tra BS (4G e 5G). Bisogna posizionare all’interno della cella i dispositivi in maniera abbastanza precisa per stabilire che frequenze utilizzare.