Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Informatica

Progettazione ed implementazione di un prototipo per la geolocalizzazione indoor dei device wifi in ambito aeroportuale

Relatore:
Chiar.mo Prof.
Vittorio Ghini

Presentata da:
Tiziano Fogli

Sessione III
Anno Accademico 2013-2014

A mio nonno.
A Checcho.

Vi ricordo con affetto.

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

Nel corso della tesi vengono affrontate tematiche relative alle tecniche di geolocalizzazione in ambienti interni. Il titolo cita il contesto aeroportuale per la sensibilità con cui queste tematiche vengono trattate in questo ambito considerato il loro possibile impatto sulla sicurezza e sulla relativa gestione. In realtà, le soluzioni analizzate e proposte sono generalizzabili a tutto il mondo indoor. Come sarà possibile osservare durante la trattazione del documento, la tecnologia fisica adottata su cui analizzare e costruire tecniche ed algoritmi di posizionamento è la tecnologia wireless 802.11.

L'obiettivo della tesi, è quello di analizzare le caratteristiche del traffico generato dai dispositivi mobili e di utilizzarle, in una seconda fase, per realizzare un prototipo in grado di individuare i device vicini a determinati punti strategici (i.e: $\pm \varepsilon m$ da un punto prefissato). L'intero sistema è quindi incentrato sulla necessità di individuare la posizione degli utenti. Il punto di vista non è per cui quello dell'utilizzatore finale, come può ad esempio accadere nel caso di applicazioni *location aware*, ma è quello di un supervisore in grado di conoscere la posizione di tutti i device individuati all'interno di una determinata area. Un sistema di questo tipo ha ovvi riscontri per quanto riguarda tematiche di sicurezza ed economiche.

Per giungere a tale obiettivo, sono state, dapprima, studiate le tipologie di soluzioni disponibili in letteratura con relativi pro e contro, sono state in seguito approntate delle semplici simulazioni sulla base di alcuni modelli di propagazione e, successivamente, si è realizzata una infrastruttura che implementa le scelte operate e le confronta con i risultati ottenuti dalle simulazioni.

L"approccio introdotto nel corso della tesi è frutto di un progetto aziendale e ha gettato le basi per un sistema ora in produzione in un importante aeroporto italiano. Tutto il lavoro svolto, viene descritto nei vari capitoli di questo documento come segue.

Nel primo capitolo vengono esplicate le motivazioni per cui si è scelto di concentrare l'attenzione su 802.11. Vengono elencati alcuni dati statistici che mostrano come, sempre più massicciamente, vengano adottati dispositivi mobili dotati di interfacce wireless e come sempre di più gli utenti utilizzino la connettività a loro disposizione. Infine, viene formalizzato l'obiettivo del prototipo applicativo che si vuole realizzare e i vincoli che dovrà rispettare.

Nel secondo capitolo, vengono introdotte le principali caratteristiche dello standard 802.11, utili a motivare certe scelte tecniche operate, oltre che ad alcune soluzioni business oriented ad esso inerenti. Inoltre, vengono brevemente descritti alcuni protocolli e strumenti marginali ma comunque adottati nel corso della tesi per quanto riguarda sopratutto la gestione dell'architettura distribuita che verrà presentata.

Il terzo capitolo, presenta alcune analisi preliminari su come i dispositivi mobili si comportano in determinate situazioni. Si pensi ad esempio alla frequenza con cui eseguono scansioni attive delle reti disponibili inviando dati potenzialmente utilizzabili per geolocalizzarli. Oltre a queste prime analisi, viene presentato, nel quindi capitolo, lo stato dell'arte accademico ed una soluzione commerciale. Sulla base dei prerequisiti visti e dello stato dell'arte, verranno elencate, nel sesto capitolo, le scelte implementative adottate e presentato uno studio simulativo, attuo a verificare la fattibilità teorica della soluzione presentata.

Nel settimo capitolo, viene descritta la realizzazione del prototipo. Vengono presentati i componenti hardware e le parti software adottate oltre al funzionamento dell'architettura realizzata nella sua totalità. Inoltre, vengono eseguiti dei test in linea con quelli eseguiti nel capitolo precedente per quanto riguarda l'approccio simulativo così da poterne confrontare i risultati.

Completano la tesi, le conclusioni e i possibili sviluppi futuri.

Capitolo 1

Scenario

La geolocalizzazione di un dispositivo consiste nell'individuazione della sua posizione. Questo risultato può essere ottenuto con tecniche/tecnologie differenti. Si pensi ad esempio ai moderni smartphone dotati di un dispositivo **GPS** (*Global Positioning System*) e quindi in grado, avendo visibilità dei satelliti, di fornire **all'utente** la propria posizione con una precisione nell'ordine dei metri.

Alternativamente, si può mettere in relazione l'attenuazione del segnale rispetto alle celle telefoniche adiacenti a quella di appartenenza, in quanto la posizione delle stazioni radio base è nota.

E' possibile anche comunicare, tramite apposite API JSON, gli SSID visibili dal dispositivo a servizi remoti quali "*Google Maps Geolocation API*" [20].

A queste prime tecniche si aggiungono gli approcci *social*: basti ricordare la popolarità che ha avuto, e che ancora ha, *Foursquare* (oggi divisa in due app: *Swarm* e *Foursquare*) con il suo approccio cooperativo in cui sono gli utenti ad indicare la propria posizione, l'attività svolta, eventuali "accompagnatori", ...

Tutti questi approcci hanno uno o più aspetti in comune ed in generale sono una combinazione di questi punti:

- il punto di vista è spesso quello dell'utente finale;

- è necessaria una applicazione lato device;
- è previsto l'input da parte di un utilizzatore.

D'altro canto esistono una serie di casi d'uso in cui il punto di vista non è quello dell'utente finale ma quello di un utente, con maggiori privilegi, in grado di vedere la posizione di ogni device nella propria area di competenza. E' necessario quindi un sistema in cui non è prevista la presenza di una applicazione *ad-hoc* sui dispositivi, non è previsto che l'utilizzatore indichi esplicitamente la propria posizione e, sopratutto, i dati finali non sono più ad uso e consumo dell'utente ma dell'ente che adotta tale sistema.

I risvolti, che l'adozione di questo sistema può avere, sono legati sostanzialmente a tematiche di sicurezza e profitto. Per quanto concerne la sicurezza, avere una stima dell'affollamento di un locale o, più nel dettaglio, sapere anche quali persone si trovano in un determinato punto, può aiutare a definire la quantità corretta di personale attuo ad operare in tale zona o a restingere il campo d'indagine in caso di infrazioni. Relativamente agli aspetti economici, conoscere i pattern di movimento all'interno di un edificio di grandi dimensioni, quali possono appunto essere un aeroporto, una stazione o un grande centro commerciale, può motivare scelte commerciali in quanto, ad esempio, l'affitto dei vari locali può essere ponderato e giustificato da dati analitici.

Per la realizzazione di questo tipo di applicazione, esistono diverse tecniche basate su approcci metodologici differenti. Alcuni di questi utilizzano tecniche di *computer vision*, dove telecamere, posizionate in punti strategici per massimizzare il raggio di azione, contano i profili delle persone. Altri approcci si basano invece sulla divisione in compartimenti stagni dei vari settori di interesse e sull'applicazione di *tornelli* fisici (o basati su fotocellula) all'ingresso e all'uscita di questi. In questo modo, è possibile contare le entrate e le uscite in ogni zona e sapere esattamente quante persone sono all'interno di ognuna di esse. Entrambi questi approcci forniscono una misura quantitativa, non una informazione certa riconducibile a **quali** persone si trovano nelle aree di interesse.

Sfruttando la presenza di una infrastruttura wireless, oltre al dato quantitativo, è disponibile una serie di altre informazioni che possono ricondurre ai dati anagrafici dei soggetti individuati. Rispetto alle prime soluzioni prospettate tuttavia, vi è una approssimazione sul dato quantitativo.

Nei paragrafi che seguono, vengono analizzati alcuni aspetti statistici legati all'uso delle connessioni dati, con enfasi sul WiFi, per definire il grado di approssimazione sul valore quantitativo introdotto da tale tecnica.

1.1 Analisi traffico dati

Le immagini in figura 1.1 e 1.2 mostrano, su scala mondiale, la diffusione degli smartphone in alcuni stati sviluppati di differenti aree geografiche. Si nota come in 1.1 il trend mostri un +8-10% circa annuo in tutti gli stati[21]. Nella figura 1.2 è invece possibile notare come nel 2012 circa il 25% dei possessori di smartphone abbia usato per navigare la tecnologia WiFi. Questo dato comprende sia le connessioni a infrastrutture WiFi esistenti, sia l'uso dello smartphone come HotSpot WiFi.

I grafici in 1.3, indicano la distribuzione, nell'arco di una giornata, del traffico originato dagli smartphone nel Regno Unito[31, 32]. Il primo grafico in alto mostra l'andamento nei giorni feriali, mentre il secondo, nei giorni festivi. Ciò indica come il WiFi sia la tecnologia più adottata al di fuori delle ore lavorative o nei momenti in cui si è "fuori dalle mura di casa". Ciò nonostante, si nota che anche nelle ore lavorative, lo scarto, difficilmente, supera i 2 punti percentuali.

Alcune statistiche presentate in [16], mostrano come negli *states* l'84% del tempo totale che un utente passa utilizzando un dispositivo mobile (circa 132 minuti / giorno), lo usa per comunicare attraverso mail e social network. Interessante, è la classifica delle applicazioni più utilizzate:

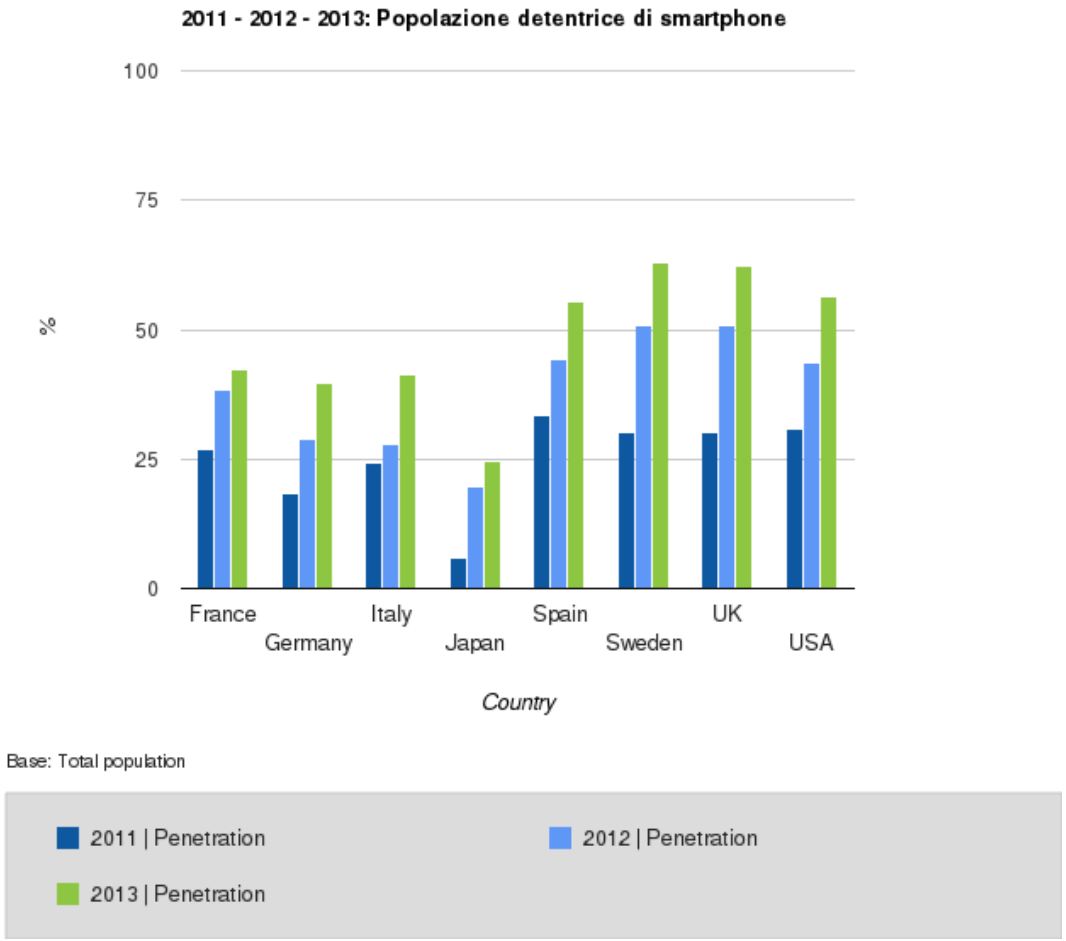


Figura 1.1: Penetrazione Smartphone

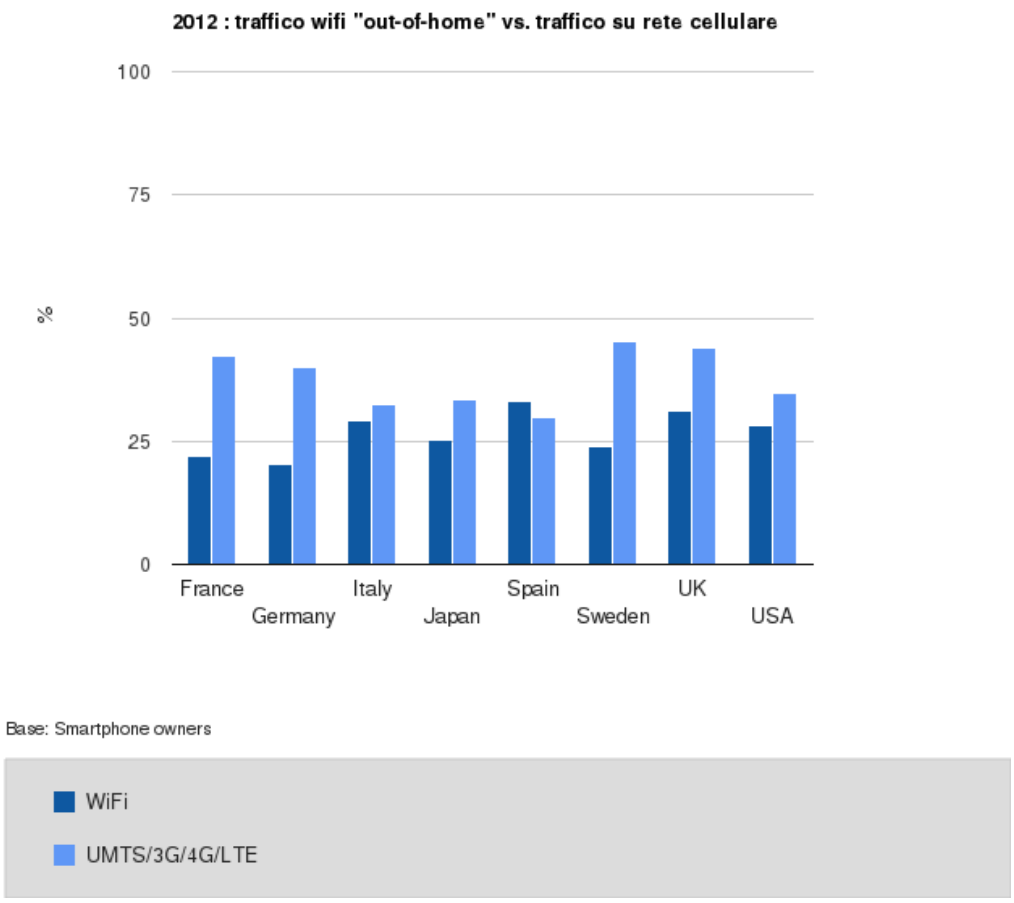
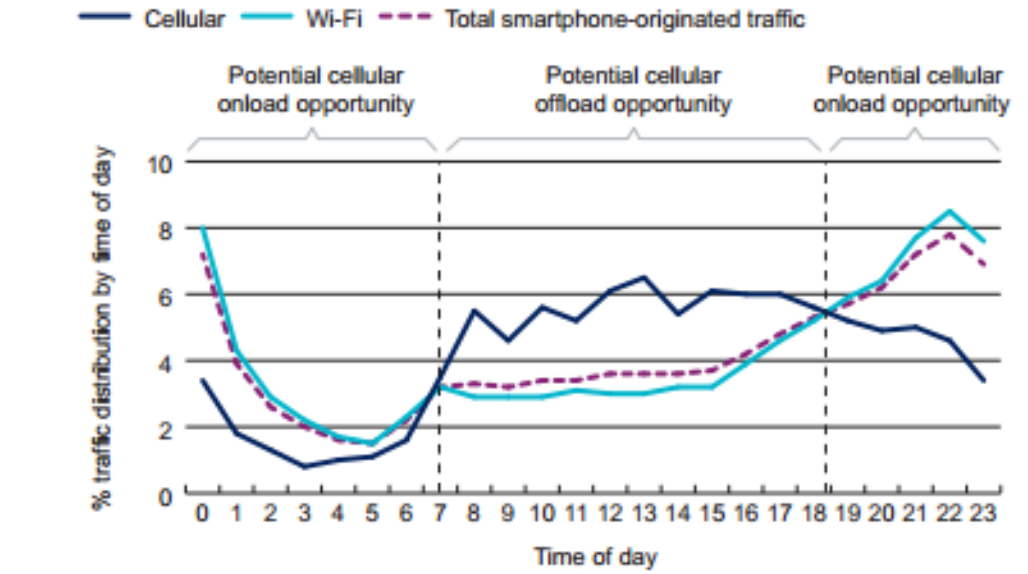


Figura 1.2: Uso connessione dati *out-of-home*



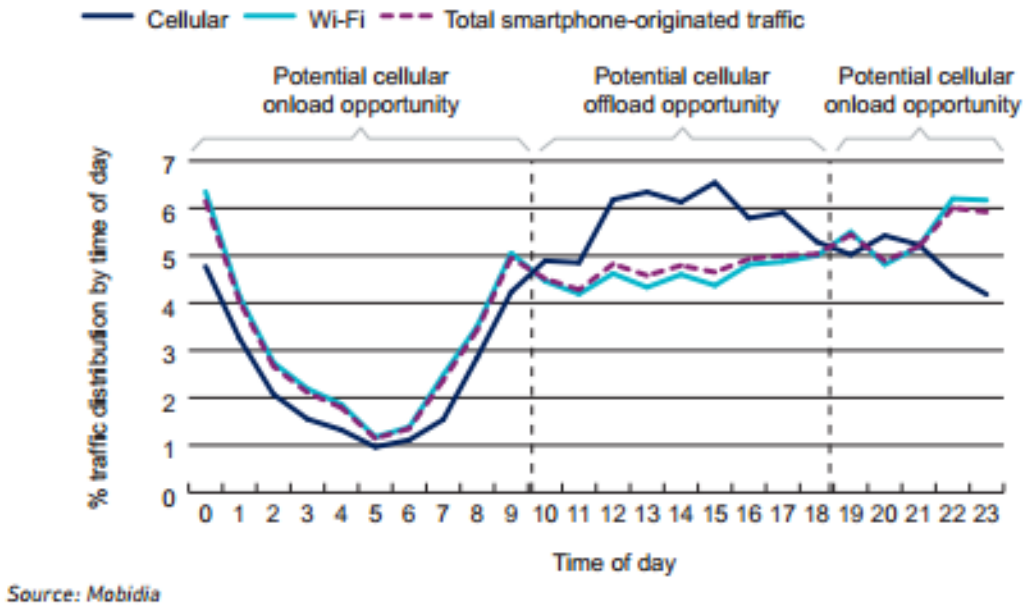


Figura 1.3: Distribuzione traffico smartphone per tecnologia nei giorni feriali (in alto) e festivi (in basso)

- 78% : Mail;
- 73% : Navigazione internet;
- 70% : Facebook;
- 64% : Consultazione mappe;
- 60% : Giochi;
- 53% : Condivisione contenuti multimediali;
- ...

Questi dati mostrano come l'uso degli smartphone, e più in generale dei dispositivi mobili, sia ormai più orientato all'ottenimento di informazioni e agli aspetti social rispetto allo storico uso del cellulare. In pratica, è possibile supporre che circa il 15-20% degli utenti possessori di smartphone utilizza la connessione dati (in egual misura WiFi e/o cellulare) costantemente: ad esempio, mediamente vengono effettuate 13 connessioni ad internet giornaliere per ogni utente possessore di smartphone.

1.2 Obiettivo e prerequisiti

I dati mostrati in 1, indicano come possano ritenersi presenti le basi per lo sviluppo di una applicazione per l'enumerazione e l'identificazione dei dispositivi in zone d'interesse opportunamente scelte.

Obiettivo della tesi, sono *(i)* l'analisi dei prerequisiti e degli elementi caratterizzanti, *(ii)* lo studio delle tecniche di geolocalizzazione ad oggi utilizzabili, e *(iii)* la realizzazione di un prototipo applicativo che fornisca dati utili per inferire il numero di persone in determinate aree circoscritte.

Il contesto aeroportuale, citato nel titolo dell'elaborato, non vincola l'uso della soluzione in altre strutture ma prende spunto da una precisa richiesta di ENAC (*Ente Nazionale Aviazione Civile*) per la gestione della sicurezza aeroportuale.

Nel caso specifico, si considerino gli usi che una tale applicazione potrebbe avere in tre momenti specifici:

- check-in;
- controlli di sicurezza;
- gate.

L'apertura, o chiusura, di un check-in aggiuntivo potrebbe essere guidata automaticamente per evitare il formarsi di file o per rallentare il flusso verso i controlli di sicurezza, per cui vale il discorso analogo.

Va considerato altresì che in molti aeroporti è a disposizione dei viaggiatori l'uso del WiFi (gratuito o a pagamento), previa registrazione a mezzo SMS. Grazie a questa tecnica di autenticazione si viene in possesso di due dati che possono portare all'identificazione non solo della posizione di un dispositivo, ma anche dell'identità del proprietario ottenibili incrociando, grazie ai *log* a disposizione, il mac address con cui è stata effettuata la richiesta di registrazione e il numero di telefono a cui vengono inviate le credenziali di accesso. Ovviamente, questo può avere dei risvolti qualora si renda necessario l'intervento, anche durante un'indagine a posteriori, della Polizia Aeroportuale.

Per non dipendere dagli utenti l'applicazione non deve prevedere l'installazione di una *app* lato client.

1.2.0.1 Definizione del problema di localizzazione

La definizione del problema della localizzazione dei device può essere formulata come segue[24].

Si consideri una rete dove i device, sono posti in posizioni sconosciute $D = \left\{d^1, d^2, ..., d^N\right\}$. Sia d^i_X la coordinata sull'asse x della *i-esimo* dispositivo e d^i_Y e d^i_Z le coordinate sugli assi Y e Z. Qualora si consideri un sistema di riferimento in 2D, si forzi $d^i_Z = 0$.

Determinare il valore di d^i_X,d^i_Y e d^i_Z sintetizza il concetto di algoritmo di localizzazione. L'individuazione di tali punti, può essere raggiunta tramite

l'uso di altri nodi, detti *beacon* o *sonde*, $B = \{b^1, b^2, ..., b^M\}$ posti in punti noti b^i_X,b^i_Y e b^i_Z.

Il problema diviene quindi così formulabile: Data una rete rappresentata da un grafo $G = (V, E)$, un insieme di *beacon* B e le relative posizioni $\{\{b_X, b_Y, b_Z\} \mid \forall b \in B\}$, si vuole ottenere l'insieme $\{\{d_X, d_Y, d_Z\} \mid \forall d \in D\}$.

Una categorizzazione, ed un breve *excursus*, degli approcci utilizzabili sia dal punto di vista accademico che di quello commerciale, viene visto in 4. Prima di tale dissertazione, vengono introdotte le basi delle tecnologie adottate nel corso di questo documento, o utili alla comprensione dello stesso.

Capitolo 2

Tecnologie

2.1 Simple Network Management Protocol

Il *Simple Network Management Protocol* o **SNMP**, è un protocollo posto al livello 7 dello stack ISO/OSI, che consente la configurazione, la gestione e la supervisione (o *monitoring*) di ciascun nodo collegato ad una rete e correttamente configurato per essere gestito attraverso tale protocollo. Esistono 3 versioni di SNMP: *(i)* la versione 1 è ormai in disuso e adottata dai soli apparati in *end-of-sale*[5], *(ii)* la versione 2 estende la 1 con nuovi tipi di dato a 64 bit [6], *(iii)* la terza ed ultima versione del protocollo introduce meccanismi di crittografia ed autenticazione utilizzabili insieme o separatamente.

Per la versione 1 e 2 gli unici meccanismi di sicurezza (facilmente aggirabili) sono: la definizione di una stringa, detta *community* (di default con valore **public**), che gli interlocutori remoti devono conoscere e comunicare ad ogni transazione e la possibilità di definire ACL (*Access List*) sugli ip dei nodi che possono fare richieste.

SNMP sfrutta come livello trasporto UDP (porta 161 per protocollo richiesta / risposta, porta 162 per protocollo ”push” previsto dall'uso di TRAP).

Il protocollo prevede tre componenti logici fondamentali:

1. *Managed Object* : ogni dispositivo dotato di un demone (Management Agent) che fornisca un'interfaccia di gestione SNMP;

2. *Management Agent* (e relativi sub-agent) : demone che consente l'interfacciamento via protocollo SNMP fra i manager remoti e i subagent dedicati per i contesti specifici sul Managed Object;

3. *Remote manager* : tool utilizzato per effettuare le interrogazioni ai Managed Object.

Per la definizione degli oggetti con cui è possibile interfacciarsi da remoto, SNMP definisce, per ogni sottosistema, una base di dati detta **MIB** (*Management Information Base*). Questa rappresenta la porzione di stato del sottosistema che si vuole rendere disponibile tramite il Management Agent e come il management agent deve mostrare tale stato. Considerato che SNMP consente non solo il monitoring ma anche il management dei subagent remoti, l'accesso alle MIB può essere in sola lettura o in lettura/scrittura. Sovrascrivendo un valore di una MIB, il subagent va a modificare realmente lo stato del sottosistema gestito.

L'interrogazione ai demoni SNMP avviene con un protocollo di tipo *"richiesta/risposta chiave->valore*" in quanto i valori della MIB vengono acceduti con richieste di tipo *GET* comunicando la chiave (detta **OID**) del campo di cui si vuole conoscere il valore. Negli esempi seguenti si vedono 2 interrogazioni analoghe: nella prima viene mostrato il nome della MIB acceduta (*ifdescr*) ed il relativo valore, nella seconda viene mostrata la OID relativa alla MIB acceduta e il relativo valore.

```
1. # snmpwalk -v2c -c public localhost ifdescr -mALL

IF-MIB::ifDescr.1 = STRING: lo
IF-MIB::ifDescr.2 = STRING: eth0
IF-MIB::ifDescr.3 = STRING: eth0.10
IF-MIB::ifDescr.4 = STRING: eth0.100
IF-MIB::ifDescr.5 = STRING: eth0.55
```

```
IF-MIB::ifDescr.6 = STRING: tun0

2. # snmpwalk -v2c -c public localhost ifdescr -mALL -On

.1.3.6.1.2.1.2.2.1.2.1 = STRING: lo
.1.3.6.1.2.1.2.2.1.2.2 = STRING: eth0
.1.3.6.1.2.1.2.2.1.2.3 = STRING: eth0.10
.1.3.6.1.2.1.2.2.1.2.4 = STRING: eth0.100
.1.3.6.1.2.1.2.2.1.2.5 = STRING: eth0.55
.1.3.6.1.2.1.2.2.1.2.6 = STRING: tun0
```

2.2 Lo standard WiFi

La tecnologia wireless nasce per trasmissioni dati punto a punto ed annovera tra i pionieri Guglielmo Marconi e Nikola Tesla. Successivamente, nel '71, con il progetto ALOHANET[1], sviluppato presso l'università delle Hawaii, furono introdotti i primi protocolli per l'accesso multiplo ad uno stesso mezzo condiviso che, trattandosi di wireless, è a tutti gli effetti una frequenza radio.

Per prendere piede questa tecnologia ha impiegato diversi anni in quanto negli Stati Uniti l'ente regolatore FCC (*Federal Communications Commission*) ha liberalizzato le bande per uso ISM (*Instrumental, Medical Scientifical*) solo nel 1985 e ha consentito solo allora ai costruttori di produrre hardware senza dover acquistare licenze. Le prime soluzioni sono apparse così nei primi anni 90 fino allo standard 802.11[8] del 1997 che segna l'evoluzione delle tecnologie wifi così come le conosciamo oggi.

Le lunghe tempistiche che sono state necessarie a tali tecnologie per divenire popolari sono state dettate da una serie di sfide da vincere rispetto a quanto accaduto per le reti cablate:

- nella ethernet moderna ogni interlocutore utilizza un suo canale fisico; nelle reti wireless il canale è unico e condiviso;

- rispetto a quanto accade nelle reti cablate moderne la banda è un bene prezioso;
- la propagazione in un canale quale quello radio è soggetta ad interferenze e disturbi;
- le trasmissioni radio non sono ”full duplex”: ogni stazione riceve o trasmette ed una sola per volta trasmette;
- il mezzo radio è diffuso e non delimitabile per cui rispetto al classico ”cavo” esistono problematiche di sicurezza.

2.2.1 Terminologia ed Architettura

Per l'introduzione dello standard e per una maggiore comprensione è necessario introdurre in parte la terminologia adottata e introdotta dallo standard stesso partendo dagli ”attori” in gioco:

STA o stazione: è definita come l'unità minima di indirizzamento. Ciò sta ad indicare che non è nient'altro che l'origine o la destinazione di un messaggio;

BSS o basic service set: è il componente base di una WLAN. E' immaginabile come un insieme di STA in grado di comunicare tra loro. Per far parte di un BSS una STA deve portare avanti una procedura di *associazione* (tramite apposite primitive e frame). Le STA possono inoltre abbandonare esplicitamente il BSS o implicitamente quando escono dalla *BSA* sfruttando quindi un meccanismo dinamico. Un BSS è caratterizzato da un BSSID: il MAC address dell'Access Point in caso si tratti di un *Infrastructured BSS*. E' possibile avere un BSS senza AP: in questo caso si ha una rete *ad-hoc* e parliamo di *Independent BSS*.

BSA o basic service area: è l'area di copertura massima dove le STA che appartengono ad una BSS riescono a rimanere in comunicazione fra loro.

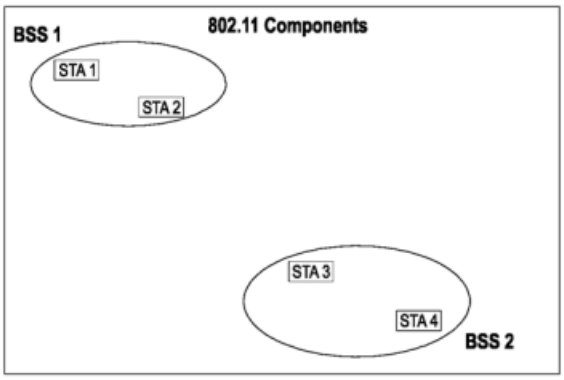


Figura 2.1: STA, BSS e BSA

DS o distribution system: le STA riescono a comunicare solo se all'interno della stessa BSA, cioè se appartenenti allo stesso BSS. In molte applicazioni questa è una copertura sufficiente mentre per altri contesti è necessaria una copertura maggiore. Il componente usato per interconnettere vari BSS è chiamato DS. La tecnologia con cui è implementato il DS, oggi tipicamente ethernet, è detta **DSM** (*distribution system medium*).

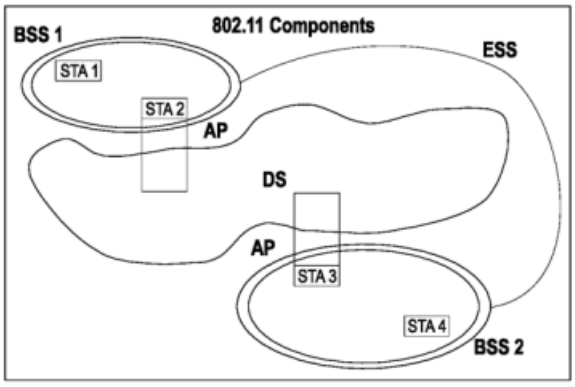


Figura 2.2: ESS

ESS o extended service set: i BSS interconnessi attraverso il DS possono creare un ecosistema wireless complesso e di dimensione arbitraria. Un ESS è quindi definibile come una unione di BSS che condividono uno

stesso SSID (*service set identifier*: valore arbitrario al massimo di 32bit che lo standard non indica come interpretare ed è quindi interpretato *de facto* come ASCII) connessi tramite un DS. Come indicato nello standard, il concetto chiave è che una STA, all'interno di un ESS ed appartenente ad un BSS specifico, può comunicare con una STA di un qualsiasi altro in maniera trasparente per il livello LLC (*logical link layer*): l'intera rete ESS è, cioè, vista dal livello LLC come fosse locale al BSS. I BSS, per poter formare un ESS, devono essere quindi interconnessi tramite il DSM e, perché questo sia possibile, è necessario un AP per ognuno.

AP o access point: collega il wireless medium (WM) con il DSM. Ogni AP ha funzionalità di STA ed abilita l'accesso al DS per le STA associate.

IBSS o infrastructure BSS: un BSS appartenente ad un ESS deve necessariamente avere un AP in quanto la STA dell'AP parla con le altre STA (che non parlano mai direttamente tra loro) e con il DS. In questo caso si dice che il BSS lavora in modalità infrastruttura e la comunicazione è centralizzata.

Roaming : è una procedura eseguita dalle STA appartenenti ad un BSS. Queste effettuano continuamente delle ricerche per identificare BSS migliori ed appartenenti allo stesso ESS. Una volta identificato spontaneamente si disassociano dal BSS attuale per associarsi al nuovo.

2.2.2 802.11a/b/g/n

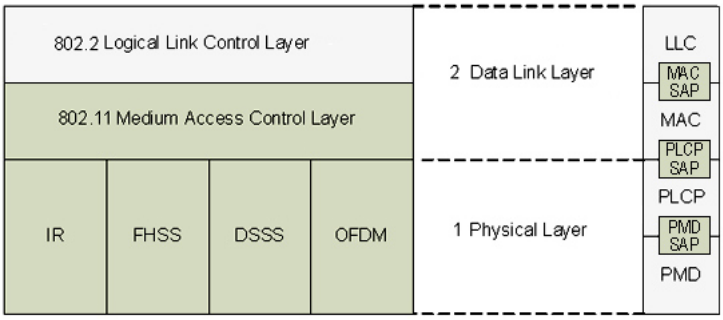


Figura 2.3: Stack 802.11

Lo standard 802.11 copre i livelli 1 e 2 dello stack iso/osi. Le principale mansioni del livello 1, o *PHY*, sono tre:

- l'ascolto del canale per capire quando stanno arrivando dati da ricevere;
- l'ascolto del canale per capire quando lo stesso è libero e pronto per una trasmissione (i.e: implementa CSMA/CA);
- la modulazione e la demodulazione dei segnali radio.

2.2.2.1 802.11 (DSSS)

Standard del 1997 con tre livelli PHY alternativi:

- DSSS
- FHSS
- INFRARED

Opera in un canale fra 14 disponibili (con varie limitazioni in base alla nazione) sfruttando DBPSK per le trasmissioni ad 1Mbps e DQPSK per le trasmissioni a 2 Mbps. Il più comune fra questi livelli PHY è quello basato su DSSS che sfrutta le frequenze nell'intorno dei 2.4GHz dove ciascuno dei 14 canali ha una banda pari a 22MHz ed è distanziato dal precedente e dal

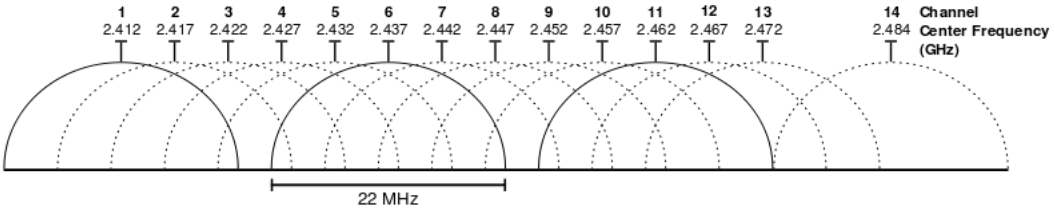


Figura 2.4: Canali 802.11

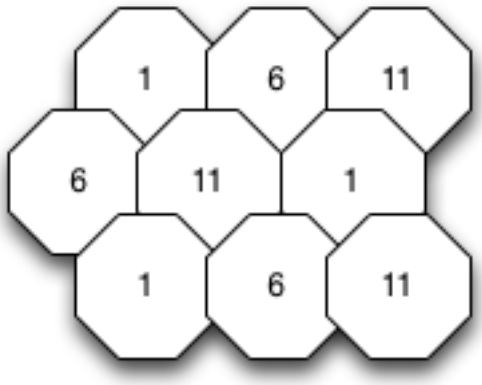


Figura 2.5: Pattern riuso canali 802.11

successivo di 5MHz. Peculiarità dei canali è la loro sovrapposizione parziale. Lo schema dei canali in figura 2.4 è di grande importanza in quanto comune a tutti gli altri standard che utilizzano portanti a 2.4GHz.

Per evitare interferenze è possibile adottare i soli canali 1,6 ed 11. Dovendo coprire aree fisiche molto estese è possibile adottare ulteriori canali teoricamente "overlappanti" ma effettivamente non interferenti in quanto fisicamente coprenti aree non adiacenti (esempio in fig.: 2.5).

2.2.2.2 802.11a

Questo standard ha visto i primi prodotti uscire nel 2001 (fu ratificato nel '99) dopo la ratifica di 802.11b. In pratica, questo ne ha determinato una diffusione limitata ma alcuni principi adottati restano validi per standard

successivi.

In 802.11a viene utilizzata la modulazione *OFDM* con portanti nella banda dei 5Ghz così da mandare il segnale su più carrier e raggiungere bit rate di 6, 9, 12, 18, 24, 36, 48 o 54 Mbit/s.

2.2.2.3 802.11b

E' una versione High rate di 802.11 in quanto è retrocompatibile e permette rate aggiuntivi quali 5.5 e 11Mbit/s. Adotta i canali a 2.4GHz.

2.2.2.4 802.11g

Ratificato nel 2003, aggiunge a 802.11b (mantenendo la retrocompatibilità) la modulazione OFDM ed utilizza le portanti a 2.4GHz. I rate disponibili sono 1, 2, 5.5, 6, 9, 11, 12, 18, 24, 36, 48 e 54 Mbit/s.

2.2.2.5 802.11n

E' uno standard del 2009 ratificato dopo un lungo periodo di gestazione che ha dato vita a prodotti non conformi. Può utilizzare entrambi i set di canali a 2.4GHz e a 5GHz: i prodotti che supportano entrambi sono detti *dual-band*. E' retrocompatibile con 802.11a, b e g. Viene introdotta la tecnologia MIMO (*Multiple Input - Multiple Output*) che sfrutta l'interferenza costruttiva del *multipath*. In pratica, offre miglioramenti in termini di throughput e distanza trasmissiva senza dover utilizzare più banda o potenze superiori. Il massimo rate utilizzabile è pari a 300Mbit/s.

2.2.3 802.11 Mac Layer

Come si può osservare in figura 2.3, il livello LLC di 802.11 viene ereditato da 802.2. 802.11 infatti definisce i soli livelli MAC e PHY. Non viene per cui utilizzato l'incapsulamento tipico della Ethernet (DIX / Ethertype) e le trame vengono direttamente tradotte in trame SNAP (*SubNetwork Access Protocol*) che non è altro che una trama 802.2 con destination e source access point pari

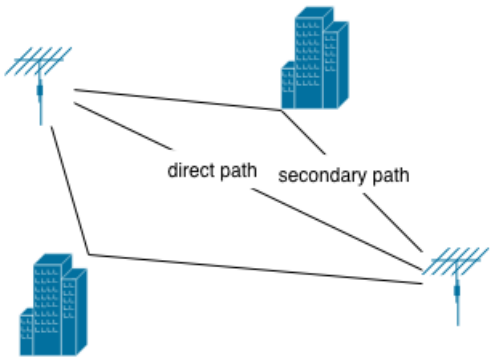


Figura 2.6: Multipath Propagation

a **AA**, control pari a **3** (cioè è un datagramma o *Unnumbered Information*), 3 byte posti a 0, ethertype (e.g: 0x800 per IP) e dati.

I servizi offerti dal MAC 802.11 sono i seguenti:

- Servizi tra STA:
  - Autenticazione (le STA si autenticano prima di associarsi ad un E/B-SS);
  - Deautenticazione (meccanismi di de-autenticazione esplicita delle STA);
  - Confidenzialità.
- Servizi sul DS forniti dagli AP:
  - Associazione (le STA si associano ad un AP per comunicare con tutto l'ESS);
  - Disassociazione (meccanismi di de-autenticazione esplicita delle STA o su iniziativa dell'AP);
  - Riassociazione (dovuta ad una disassociazione e ad una nuova associazione con un AP appartenente allo stesso ESS: roaming. Avviene sempre su iniziativa della STA wireless);

  - Distribuzione (trasmissione dati provenienti dai BSS attraverso il DS).

- Gestione accesso multiplo al canale

  - DCF (*Distributed Coordination Function*): CSMA/CA, concettualmente simile al CSMA/CD della Ethernet ma con alcune differenze importanti dovute al fatto che su WM non è possibile trasmettere e fare carrier sensing contemporaneamente ed è quindi impossibile fare *collision detection.*

    * Completamente decentralizzato;
    * Paritario;
    * Unico meccanismo diffuso nei dispositivi sul mercato;
    * Non potendo rilevare le collisioni, si tenta di evitarle.

  - PCF (Point Coordination Function)

2.2.3.1 CSMA/CA

L'obiettivo del meccanismo CSMA/CA adottato da DCF è quello di evitare le collisioni partendo dal presupposto che queste non sono rilevabili. Per prima cosa ogni STA, prima di trasmettere, verifica che il canale fisico sia libero e mantiene un contatore, detto NAV (*Network Allocation Vector*), opportunamente inizializzato, di volta in volta, con i meccanismi che saranno introdotti e che indica per quanto tempo il canale sarà occupato.

Il lasso di tempo in cui una STA ascolta prima di trasmettere è detto DIFS (*DCF InterFrame Space*) ed è fissato ed indipendente dal bit rate. Se dopo un DIFS:

- il canale è libero, la STA inizia a trasmettere;

- il canale è occupato, la STA aspetta un ulteriore DIFS più un tempo di backoff casuale. Se dopo questo tempo il canale è libero inizia a trasmettere;

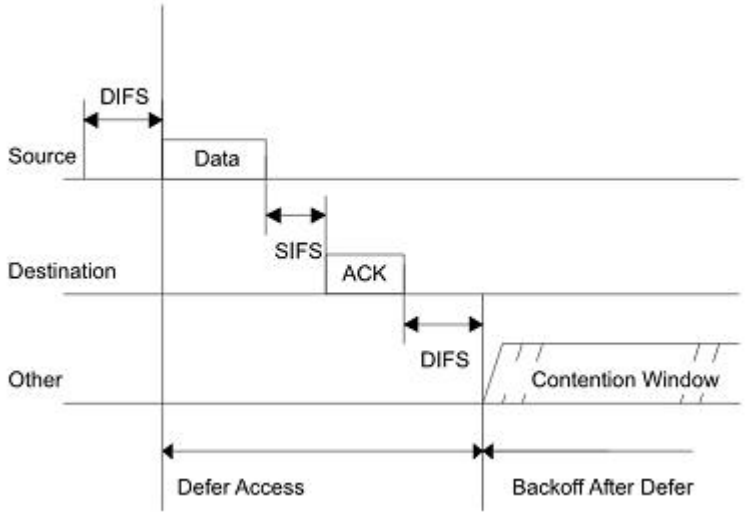


Figura 2.7: DCF

Considerate le caratteristiche "ostiche" del canale radio, 802.11 utilizza un sistema di ack positivi, per cui una trama non è considerata trasmessa correttamente a meno che non ne sia stato ricevuto il relativo ack. Le STA riceventi per trasmettere l'ACK non devono aspettare un intero DIFS ma un intervallo più breve detto SIFS o *Short InterFrame Space.*

Tutte le STA di una BSA, quanto sentono l'inizio di una trama settano il NAV per una durata pari alla somma di:

1. Lunghezza della trama;

2. SIFS;

3. ACK.

Finché il loro NAV non è esaurito le STA non possono trasmettere. Terminato il NAV, devono aspettare un nuovo DIFS ed un altro tempo di backoff casuale.

2.2.3.2 Hidden/Exposed Terminal e Frammentazione

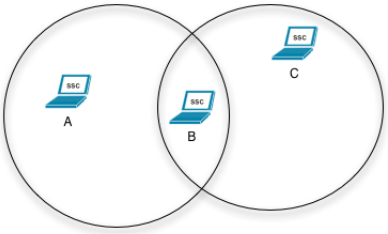


Figura 2.8: Hidden Terminal

- A trasmette verso B ;
- Se ora C ascolta il mezzo, lo troverà libero e sarà convinto di poter trasmettere a B ;
- Cominciando a trasmettere disturberà la trasmissione di A, impedendo a B di riceverla e sia A che C saranno costrette a ritrasmettere.

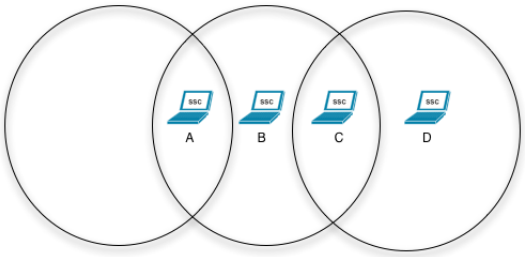


Figura 2.9: Exposed Terminal

- B trasmette verso A e C voglia trasmettere verso D ;
- effettuando il Carrier Sensing C sentirà che il mezzo è occupato (trasmissione di B) e concluderà erroneamente di non poter trasmettere; invece, essendo D fuori della portata di B ed A fuori della portata di C, le due trasmissioni potrebbero avvenire parallelamente senza interferenza.

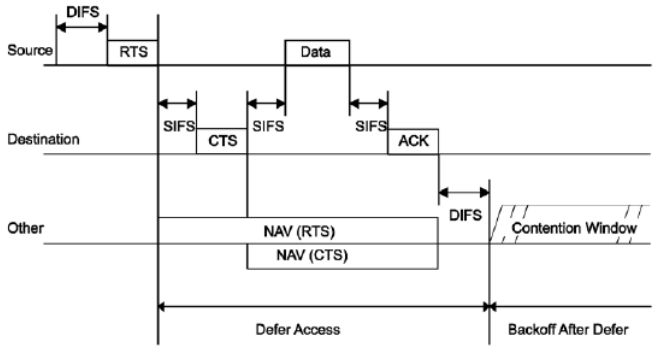


Figura 2.10: CTS-RTS CSMA/CA

Il problema dell'Hidden Node, mostrato in figura 2.8, è risolvibile introducendo due pacchetti ulteriori per poter trasmettere trama e relativo ack a fronte di un maggiore overhead. Per limitare l'overhead viene impostata una soglia, detta RTS Threshold, al di sotto della quale il meccanismo di RTS/CTS non è adottato. Questi ulteriori pacchetti da trasmettere necessitano quindi di essere gestiti anche per quel che riguarda i tempi di attesa di CSMA/CA.

Sostanzialmente, prima che A invii la propria trama a B, invia un pacchetto chiamato RTS (*Request To Send*) a B. Tale operazione setta il NAV in tutte le STA che ricevono RTS. B risponde ad A con CTS (*Clear To Send*) dopo un SIFS e setta quindi il NAV in tutte le STA della sua BSA. I pacchetti RTS e CTS permettono alle STA di settare il NAV in quanto contengono il campo *Duration* relativo alla trama oggetto della comunicazione.

In 802.11 è inoltre prevista la possibilità di frammentare le trame della trasmissione. Questo dà la possibilità di gestire trame di dimensione superiore alla massima consentita (tipicamente pari alla soglia e di RTS Threshold) e di resistere meglio alle interferenze di breve durata. Per ogni frammento è previsto un ACK. La trasmissione di una sequenza di frammenti di una stessa trama è atomica ed è detta Frame Exchange Sequence (questo è implementato aspettando solo un SIFS fra la ricezione dell'ACK e l'inizio della trasmissione del frammento successivo). Ogni frammento prenota il NAV per il frammento successivo e il suo ACK.

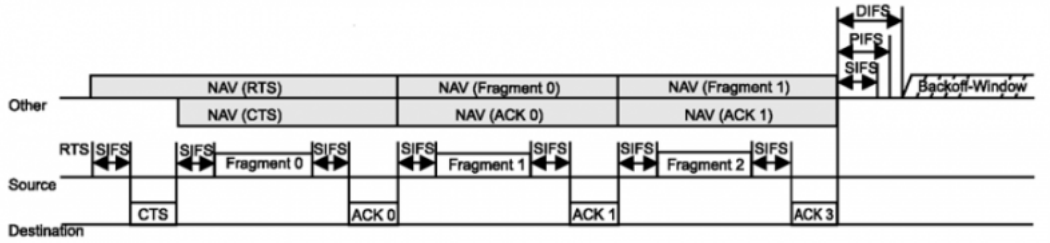


Figura 2.11: Frammentazione 802.11

2.2.3.3 Scanning

Lo standard definisce due differenti metodi a disposizione delle STA per venire a conoscenza dei BSS disponibili. Tale procedura è detta *scanning* e può essere svolta in modalità *active* o *passive.*

Passive : Le STA ascoltano su ogni canale delle porzioni di tempo per ricevere le trame **BEACON** inviate dagli AP. Queste trame sono inviate ad intervalli di tempo regolari a patto che il canale risulti libero, altrimenti viene post-posta. Le trame BEACON contengono le informazioni in base alle quali le STA decidono come e se associarsi. Va considerato che quando un AP appartiene a due BSS differenti, invia due beacon ed usa due BSSID differenti. E' la meno dispendiosa delle tecniche di scanning.

Active : Una STA "scansiona" tutti i possibili canali sequenzialmente e per ciascuno di essi:

1. si mette in ricezione, per imparare un NAV e trasmettere in modo ordinato, per un tempo massimo di ProbeDelay;
2. trasmette un Probe Request, con i BSSID e/o SSID desiderati, eventualmente una wildcard;
3. se nessuno risponde entro un tempo limite pari a *MinChannelTime*, il canale viene considerato vuoto;
4. se qualcuno risponde entro il *MinChannelTime*, attende un tempo *MaxChannelTime* e raccoglie tutte le Probe Response ricevute.

Una trama Probe Response è praticamente analoga ad un BEACON e viene memorizzata allo stesso modo dalle STA. E' inoltre soggetta ad ACK come una normale trama.

2.2.3.4 Trame MAC

La trama MAC è mostrata in figura 2.12 (tratta da [12]) e composta da:

1. Header;
   (a) **Frame Control**;
   (b) **Duration**;
   (c) **Addresses**;
   (d) **Sequence Control** (opzionale);
   (e) *QoS Control* (opzionale);
   (f) *HT Control Fields* (opzionale).
2. Body con informazioni specifiche per il tipo di trama, specificato nel Frame control;
3. FCS, un CRC a 32 bit.

Il campo frame control determina il contenuto del corpo del pacchetto ed è suddiviso in:

1. **Version** (posto sempre a 0);
2. **Type** (0 Management frame, Control Frame, Data Frame) - (Si vedano tabelle 2.1 e 2.2);
3. **Subtype** (Funzione del frame in base al tipo principale) - (Si vedano tabelle 2.1 e 2.2);

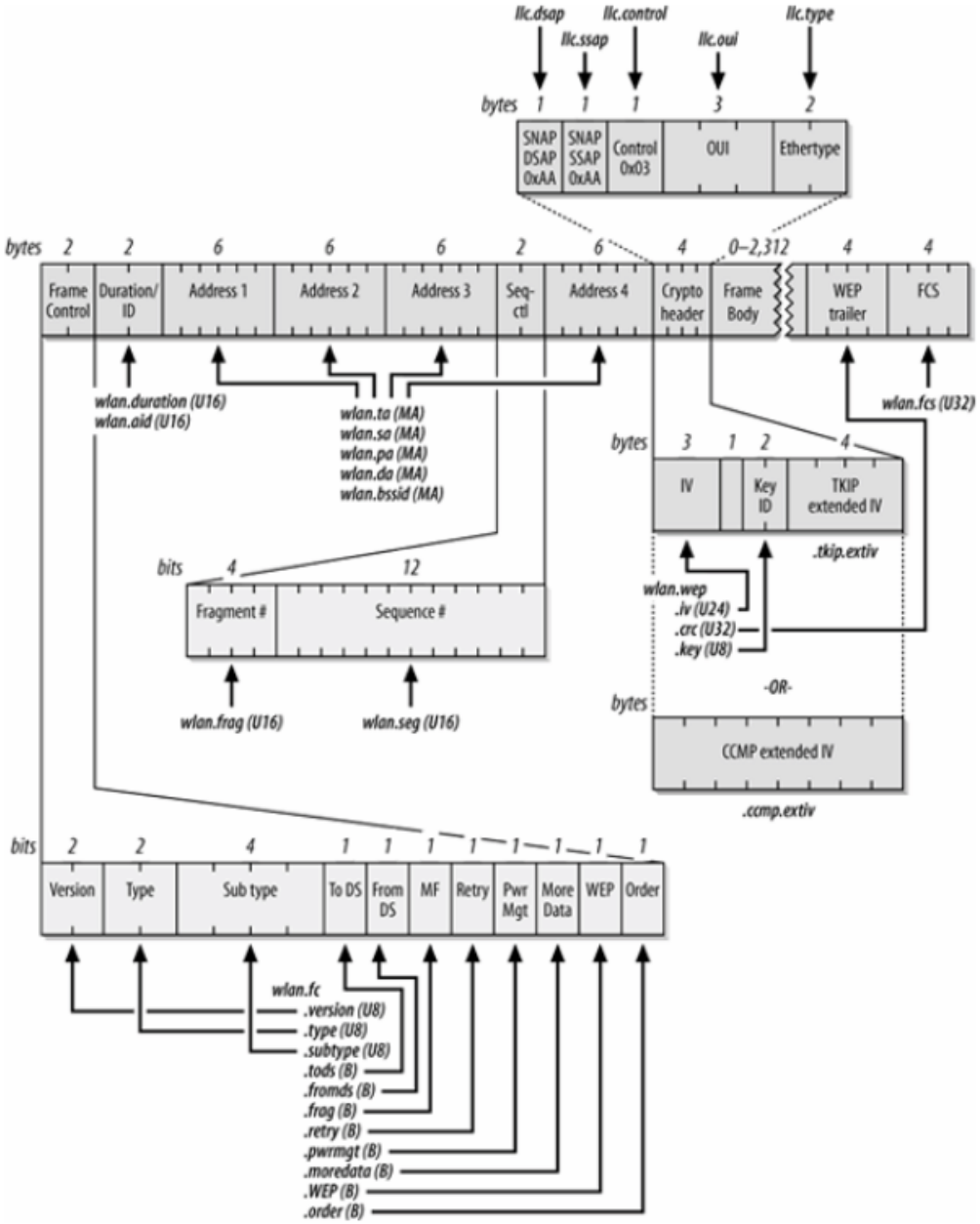


Figura 2.12: 802.11 Header

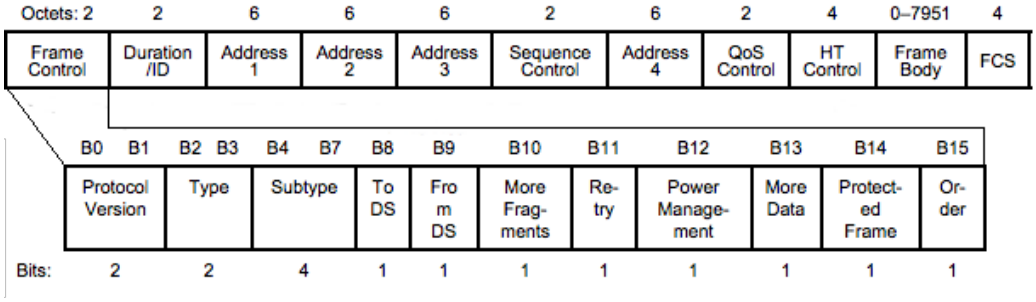


Figura 2.13: 802.11 Header - Dettaglio bit

4. **ToDS**

5. **FromDS** (Il significato di questo campo e del precedente è esplicato nelle tabelle 2.3 e 2.4)

6. **MoreFragment** (Se settato indica che restano altri frammenti della trama corrente da trasmettere)

7. **Retry** (Se settato indica che la trama è già stata trasmessa in precedenza e che si tratta quindi di una ritrasmissione);

8. **Power Management** (Se settato indica che la STA è entrata in modalità "basso consumo");

9. **More Data** (Se settato dall'AP indica alla STA che sono stati bufferizzati altri frame per questa);

10. **Protected Frame** (Indica l'encryption adottata)

11. **Order** (Se settato, indica che la STA ha richiesto la consegna ordinata dei frame inviati)

I campi ToDS e FromDS indicano la provenienza di una trama in un ESS (tabella 2.3) ed esplicano il significato dei campi Address 1 - 4 (tabella 2.4).

Si distinguono quattro tipologie di indirizzi:

1. *Source*: indirizzo del device che ha generato il pacchetto;

Type	Descrizione	**Subtype**	Descrizione
b3 - b2		b7 - b6 - b5 - b4	
00 - (0)	Management	0000	Association request
0	Management	0001	Association response
0	Management	0010	Reassociation request
0	Management	0011	Reassociation response
0	Management	0100	Probe reques
0	Management	0101	Probe response
0	Management	0110	Timing Advertisement
0	Management	0111	Reserved
0	Management	1000	Beacon
0	Management	1001	ATIM
0	Management	1010	Disassociation
0	Management	1011	Authentication
0	Management	1100	Deauthentication
0	Management	1101	Action
0	Management	1110	Action No Ack
0	Management	1111	Reserved
(01) - 1	Control	0000–0110	Reserved
1	Control	0111	Control Wrapper
1	Control	1000	Block Ack Request (BlockAckReq)
1	Control	1001	Block Ack (BlockAck)
1	Control	1010	PS-Poll
1	Control	1011	RTS
1	Control	1100	CTS
1	Control	1101	ACK
1	Control	1110	CF-End
1	Control	1111	CF-End + CF-Ack

Tabella 2.1: Type e Subtype - Control e Management

Type	Descrizione	**Subtype**	Descrizione
b3 - b2		b7 - b6 - b5 - b4	
10 - (2)	data	0000	Data
2	data	0001	Data + CF-Ack
2	data	0010	Data + CF-Poll
2	data	0011	Data + CF-Ack + CF-Poll
2	data	0100	Null (no data)
2	data	0101	CF-Ack (no data)
2	data	0110	CF-Poll (no data)
2	data	0111	CF-Ack + CF-Poll (no data)
2	data	1000	QoS Data
2	data	1001	QoS Data + CF-Ack
2	data	1010	QoS Data + CF-Poll
2	data	1011	QoS Data + CF-Ack + CF-Poll
2	data	1100	QoS Null (no data)
2	data	1101	Reserved
2	data	1110	QoS CF-Poll (no data)
2	data	1111	QoS CF-Ack + CF-Poll (no data)
11 - (3)	reserved	0000-1111	Reserved

Tabella 2.2: Type e Subtype - Data

	ToDS (Set)	ToDS (Clear)
FromDS (Set)	Mesh BSS (Wireless DS)	da DS a STA
FromDS (Clear)	da STA a DS	Traffico Intra-BSS

Tabella 2.3: ToDS / FromDS - 1

2. *Transmitter*: indirizzo del device che per ultimo ha inviato il pacchetto sul mezzo wireless;

3. *Receiver*: indirizzo del device che sarà incaricato di inoltrare il pacchetto verso la destinazione finale;

4. *Destination*: indirizzo del device a cui è destinato il pacchetto.

In particolare:

Address1: è sempre l'indirizzo della stazione ricevente. Se ToDS è settato, questo è l'indirizzo dell'AP, altrimenti è l'indirizzo della STA;

Address2: è sempre l'indirizzo della stazione trasmittente. Se FromDS è settato, l'indirizzo è quello dell'AP, altrimenti è quello della STA;

Address3: è utilizzato per assegnare l'indirizzo mancante:

- Nei frame in cui FromDS vale 1, è l'indirizzo sorgente;
- Nei frame in cui ToDS vale 1, è l'indirizzo di destinazione;

Address4: è usato nei casi in cui è presente un Wireless Distribution System e in cui FromDS e ToDS sono ambedue settati (i.e: frame trasmesso da un ap ad un altro). Contiene per cui l'indirizzo sorgente altrimenti mancante.

	ToDS (Set)	ToDS (Clear)
FromDS (Set)	Addr 1: Receiver, Addr 2 : Transmitter	Addr 1: Destination
	Addr 3 : Destination, Addr 4 : Source	Addr 2 : BSSID, Addr 3 : Source
FromDS (Clear)	Addr 1: BSSID, Addr 2 : Source	Addr 1: Destination
	Addr 3 : Destination	Addr 2 : Source, Addr 3 : BSSID

Tabella 2.4: ToDS / FromDS - 2

2.2.4 WLAN Controller, LWAPP, CAPWAPP

La gestione di un ESS che comprenda molteplici BSS (e quindi molteplici AP) può essere gravosa. In particolare, si pensi all'aggiunta di un SSID o alla modifica dei parametri relativi a sicurezza ed autenticazione: queste operazioni andrebbero effettuate manualmente su ogni singolo AP. In questo caso parliamo di *AP Standalone.*

Per agevolare la gestione di ESS complessi, è stato nel tempo sviluppato ed introdotto dai vendor il concetto di *Wireless Controller* o *WLAN Controller* o *Access Controller*. In pratica, sfruttando appositi protocolli, quali **LWAPP** (Lightweight Access Point Protocol), introdotto da Airspace che successivamente è stata acquisita da Cisco, e **CAPWAPP (***Control And Provisioning of Wireless Access Points***),** è possibile modificare i servizi erogati da ogni singolo AP in maniera centralizzata. Gli AP vengono chiamati in questo contesto anche *Wireless Termination Points.*

Avendo altresì una visione centralizzata dello stato degli AP, è possibile gestire automaticamente (i.e: algoritmicamente) compiti quali:

- Identificazione e prevenzione delle interferenze : in pratica, viene calcolato dinamicamente il miglior pattern di riuso dei canali possibile;

- Ottimizzazione della copertura : cioè viene gestita la potenza in trasmissione degli access point per ridurre le aree di intersezione di questi o per cercare di coprire l'area più vasta possibile;

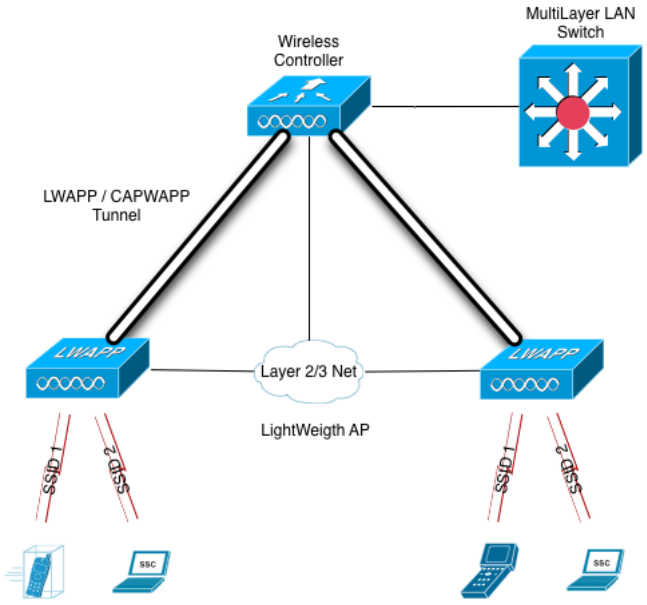


Figura 2.14: Architettura WLAN Controller

LWAPP è stato standardizzato, grazie anche ad una forte sponsorizzazione da parti di Cisco, dall'IETF nello standard RFC5412 [17]. Il protocollo prevede due tipologie di pacchetti che possono essere trasportati direttamente a livello DATA LINK o incapsulati in datagrammi UDP over IP (e quindi in Layer 3):

1. LWAPP DATA Message: Trame 802.11 incapsulate;
2. LWAPP CONTROL Message: messaggi di management scambiati tra AC E WTP.

Qualora WTP e AC comunichino in layer 2, viene utilizzato un bit nell'header di LWAPP, chiamato *C-Bit,* per distinguere DATA e CONTROL Message. Se invece comunicano in layer 3, ovvero le trame vengono incapsulate in datagrammi IP, vengono utilizzate porte UDP separate per le 2 tipologie di messaggio.

Il protocollo segue, a grandi linee, queste fasi:

Discovery

- Layer 2
  - Al boot i WTP inviano in broadcast una trama *Discovery Request* alla quale il WLAN Controller risponde con una *Discovery Response* "accettando" l'AP. Si dice che l'AP ha fatto *join* al WLC. La fase di join comprende una operazione di *MTU Discovery* per stabilire quale sia la dimensione massima delle trame supportata dal percorso fra l'AP e il WLC. I WTP in questo caso possono funzionare anche senza possedere un indirizzo ip.
- Layer 3
  - Esistono alternative differenti per effettuare il provisioning in rete Layer 3. Questi meccanismi sono necessari in quanto potenzialmente i WTP e l'AC possono trovarsi in domini di broadcast differenti e le trame Discovery Request inviate dai WTP non arriverebbero qualora AP e AC si trovassero su due sottoreti distinte.
    * DNS : Gli AP al boot, se non hanno risposte alle trame Discovery Request, tentano la risoluzione di alcuni nomi standard a DNS quali ad esempio, nel caso Cisco, *CISCO-LWAPP-CONTROLLER.localdomain*. La procedura viene portata a termine ovviamente dopo aver ricevuto un ip dinamico.
    * DHCP : In molti casi è possibile configurare il server DHCP che fornisce gli ip agli AP perché nelle lease includa l'opzione 43 (*vendor-encapsulated-options*). Tale opzione contiene informazioni utili agli ap (sottoforma di stringa esadecimale) per identificare il wireless controller e il suo formato non è standardizzato ma *vendor specific*.

```
subnet 10.1.0.0 netmask 255.255.255.0 {
        range 10.1.0.2 10.1.0.255;
```

```
        option routers 10.1.0.1;
        option domain-name-servers 192.168.0.1;
        option vendor-encapsulated-options
                80:0b:00:00:02:0a:46:01:f8:0a:46:01:f9;
}
```

L'esempio mostra le configurazioni per un controller *HP / H3C* con WTP *WA2620E-AGN*.
La stringa è nella forma: **aa:bb:cc:cc:dd:ee:ee:ee:ee:ff:ff:ff:ff** dove:

1. **aa :** tipo opzione (vendor specific)
2. **bb :** numero dei campi successivi
3. **cc** : campi fissi posti a 0
4. **dd** : numero di indirizzi ip (corrispondenti a quelli dei WLC) che seguiranno;
5. **ee**: primo indirizzo ip;
6. **ff** : secondo indirizzo ip.

L'*option router 10.1.0.1*, indica che queste configurazioni devono essere inviate solo alle richieste DHCP per cui è stato il router 10.1.0.1 a fare da *DHCP Relay*. Un esempio è fornito in figura 2.15.

Provisioning

Terminata la fase di discovery e join, avviene il *provisioning*. In particolare, l'AC invia al WTP appena "joinato" le configurazioni circa gli SSID da erogare, quindi:

1. SSID;
2. Parametri di sicurezza/autenticazione;
3. Data rate da annunciare nei Beacon;
4. Canale radio da adottare.

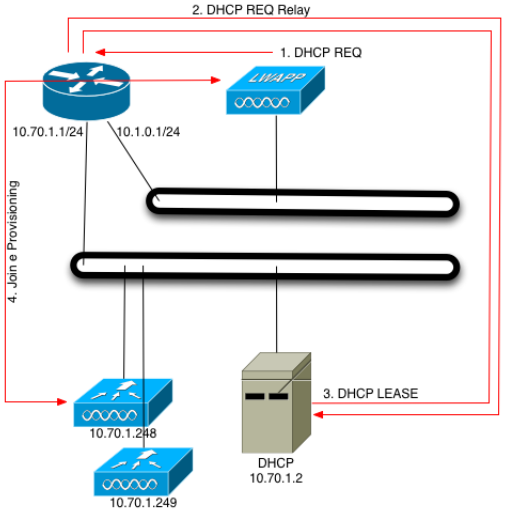


Figura 2.15: WLC DHCP Provisioning

A questo punto, gli AP sono pronti per erogare il servizio richiesto. Va notato che in fase di provisioning, il WLC potrebbe inviare all'AP un aggiornamento del firmware. In questo caso, gli AP sono tenuti ad installare l'aggiornamento, a riavviarsi e a rieseguire di conseguenza la procedura di join e provisioning. In sostanza, tale architettura consente di mantenere aggiornato il parco macchine wireless con il minimo sforzo rispetto ad una architettura con AP Standalone.

L'uso di un WLC facilita anche l'analisi di eventuali problematiche in quanto in un singolo punto di raccolta e consultazione è possibile estrapolare informazioni circa gli AP e il loro stato. Tali informazioni sono esportate anche via SNMP grazie al fatto che, tipicamente, i vendor definiscono le MIB in modo opportuno. Nell'esempio, in figura 2.16, si vedono i dati degli AP con nome AP0x, con x da 1 a 6, i relativi ip e mac address, e qualche statistica circa il numero di STA associate. Ovviamente questo è solo un estratto delle statistiche presenti che spaziano dal livello fisico alle caratteristiche dei meccanismi di autenticazione adottate da ogni singola STA.

```
snmpwalk -mALL -v2c -c public 10.70.1.248  hpnicfDot11APIPAddress
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APIPAddress."210235A42MB10B000007" = IpAddress: 10.1.0.35
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APIPAddress."210235A42MB10B000016" = IpAddress: 10.1.0.14
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APIPAddress."210235A42MB10B000130" = IpAddress: 10.1.0.236
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APIPAddress."210235A42MB10B000139" = IpAddress: 10.1.0.214
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APIPAddress."210235A42MB10B000143" = IpAddress: 10.1.0.2
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APIPAddress."210235A42MB10B000157" = IpAddress: 10.1.0.168

snmpwalk -mALL -v2c -c public 10.70.1.248  hpnicfDot11APMacAddress
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APMacAddress."210235A42MB10B000007" = STRING: 3c:d5:a6:8e:8:c0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APMacAddress."210235A42MB10B000016" = STRING: 3c:d5:a6:8e:9:e0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APMacAddress."210235A42MB10B000130" = STRING: 3c:d5:a6:c1:df:60
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APMacAddress."210235A42MB10B000139" = STRING: 3c:d5:a6:c1:e0:80
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APMacAddress."210235A42MB10B000143" = STRING: 3c:d5:a6:c1:e1:0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APMacAddress."210235A42MB10B000157" = STRING: 3c:d5:a6:c1:e2:c0

snmpwalk -mALL -v2c -c public 10.70.1.248  hpnicfDot11APTemplateNameOfAP
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APTemplateNameOfAP."210235A42MB10B000007" = STRING: "ap01"
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APTemplateNameOfAP."210235A42MB10B000016" = STRING: "ap02"
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APTemplateNameOfAP."210235A42MB10B000130" = STRING: "ap03"
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APTemplateNameOfAP."210235A42MB10B000139" = STRING: "ap04"
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APTemplateNameOfAP."210235A42MB10B000143" = STRING: "ap05"
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11APTemplateNameOfAP."210235A42MB10B000157" = STRING: "ap06"

snmpwalk -mALL -v2c -c public 10.70.1.248  hpnicfDot11ApStationAssocSum
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationAssocSum."210235A42MB10B000007" = Counter32: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationAssocSum."210235A42MB10B000016" = Counter32: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationAssocSum."210235A42MB10B000130" = Counter32: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationAssocSum."210235A42MB10B000139" = Counter32: 286
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationAssocSum."210235A42MB10B000143" = Counter32: 495
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationAssocSum."210235A42MB10B000157" = Counter32: 13

snmpwalk -mALL -v2c -c public 10.70.1.248  hpnicfDot11ApStationReassocSum
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationReassocSum."210235A42MB10B000007" = Counter32: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationReassocSum."210235A42MB10B000016" = Counter32: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationReassocSum."210235A42MB10B000130" = Counter32: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationReassocSum."210235A42MB10B000139" = Counter32: 8
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationReassocSum."210235A42MB10B000143" = Counter32: 129
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationReassocSum."210235A42MB10B000157" = Counter32: 17

snmpwalk -mALL -v2c -c public 20.70.1.248  hpnicfDot11ApStationCurAssocSum
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationCurAssocSum."210235A42MB10B000007" = INTEGER: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationCurAssocSum."210235A42MB10B000016" = INTEGER: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationCurAssocSum."210235A42MB10B000130" = INTEGER: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationCurAssocSum."210235A42MB10B000139" = INTEGER: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationCurAssocSum."210235A42MB10B000143" = INTEGER: 0
HPN-ICF-DOT11-APMT-MIB::hpnicfDot11ApStationCurAssocSum."210235A42MB10B000157" = INTEGER: 0
```

Figura 2.16: WLC SNMP Walk

Capitolo 3

Analisi Preliminare

Prima della rassegna di alcune soluzioni commerciali ed accademiche, vengono presentate alcune analisi effettuate su dispositivi mobili in commercio. Tali analisi mostrano alcune peculiarità dei dispositivi da intercettare utili per i fini del progetto.

In primis, verrà brevemente analizzata, tramite l'ausilio di dati empirici, l'attenuazione del segnale fra emettitore e ricevitore al variare della distanza.

Successivamente, sarà presentato un primo studio sulle caratteristiche dei pacchetti *probe request* con enfasi particolare sulla frequenza con cui questi vengono generati.

In seconda battuta, viene presentato uno studio sull'attenuazione che accusano i pacchetti catturati su canali adiacenti rispetto a quello su cui si effettua il dump.

Infine, verranno esposti i risultati relativi ad alcuni test relativi ai meccanismi di risparmio energetico adottati dai dispositivi client.

3.1 Attenuazione al variare della distanza

La prima analisi effettuata studia l'attenuazione del segnale al variare della distanza fra TX ed RX. Il grafico, in figura 3.1, mostra come varia la potenza ricevuta da RX. Ogni punto rappresenta 100 misurazioni mentre la

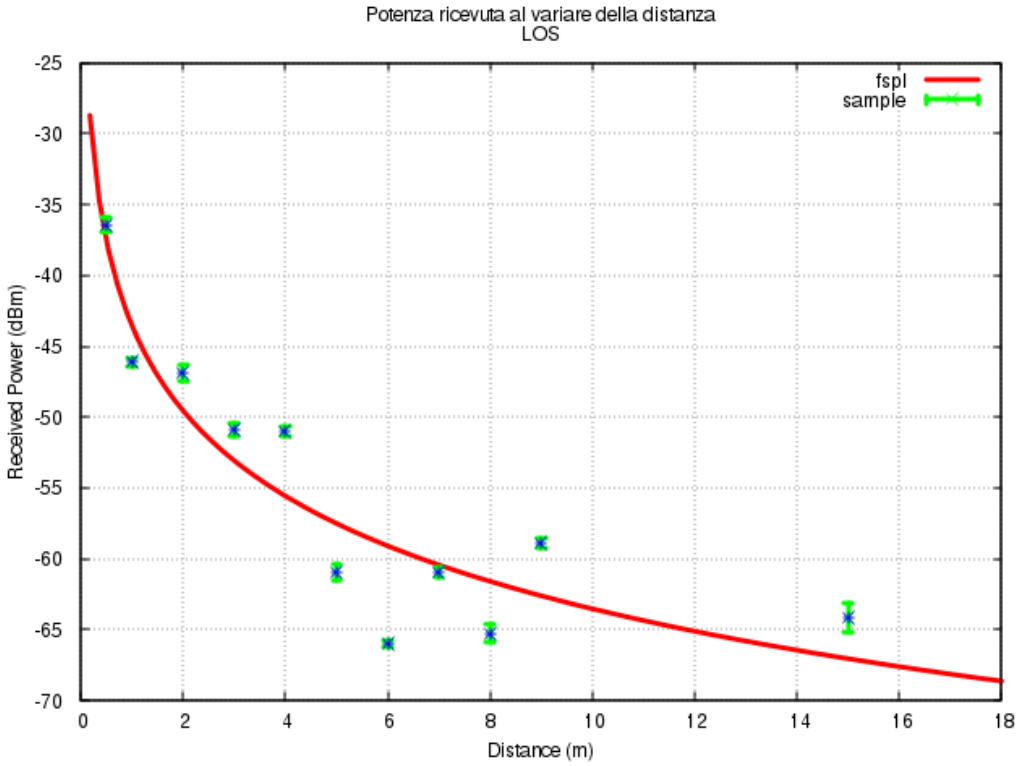


Figura 3.1: Potenza ricevuta al variare della distanza

curva FSPL è stata ottenuta interpolando i dati con il modello *Free Space Path Loss* presentato in 5.1. Si vede come l'attenuazione cresca all'incirca proporzionalmente al quadrato della distanza in quanto nei primi 5 metri si passa da una potenza ricevuta di circa *-30dBm* ad una di circa *-55dBm* per poi via via diminuire più lentamente. In questo caso, la sperimentazione è stata condotta con TX ed RX in visibilità ottica. La situazione può variare qualora questa venga a mancare in quanto le misurazioni sarebbero soggette ad una quantità di disturbi differente.

3.2 Probe

Il primo studio è relativo alla frequenza con cui vengono emessi pacchetti di tipo *Probe Request*. Scopo di questa analisi è stabilire la frequenza con cui

un dispositivo **in stand-by** emette questa tipologia di pacchetti per capire la mole di traffico minima utilizzabile per la geolocalizzazione del device.

In figura 3.3, si osserva un campione di circa due ore ottenuto da un dispositivo latente connesso ad una rete aperta (i.e: senza meccanismi di cifratura).

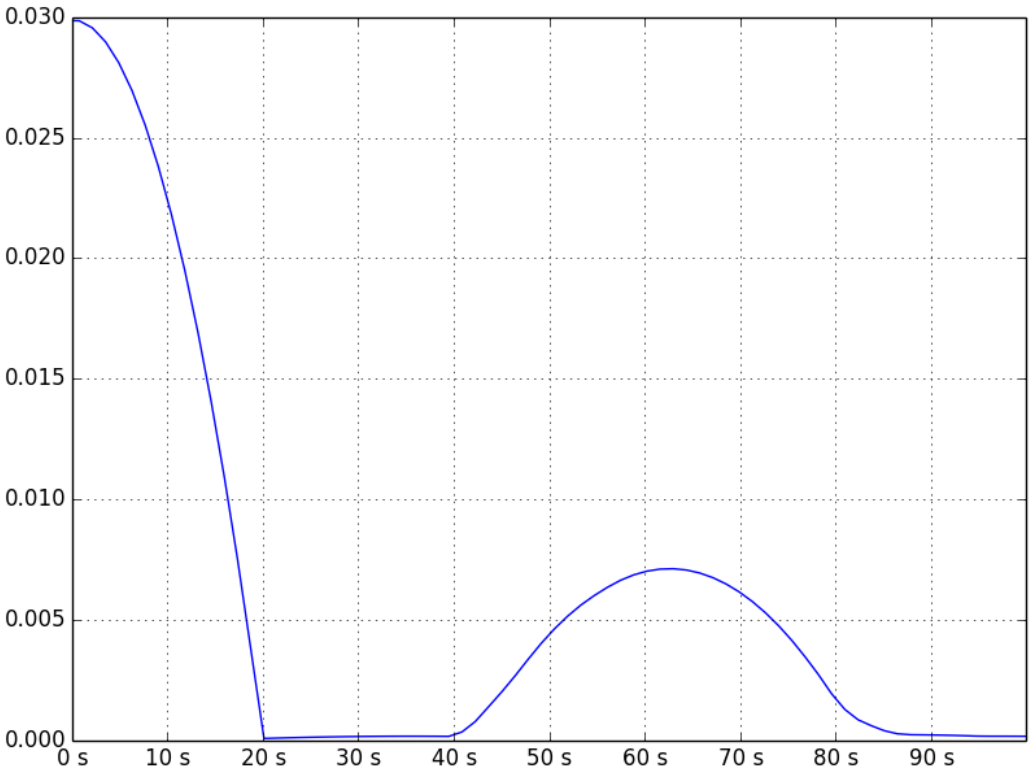


Figura 3.2: Distribuzione di probabilità - probe nel tempo

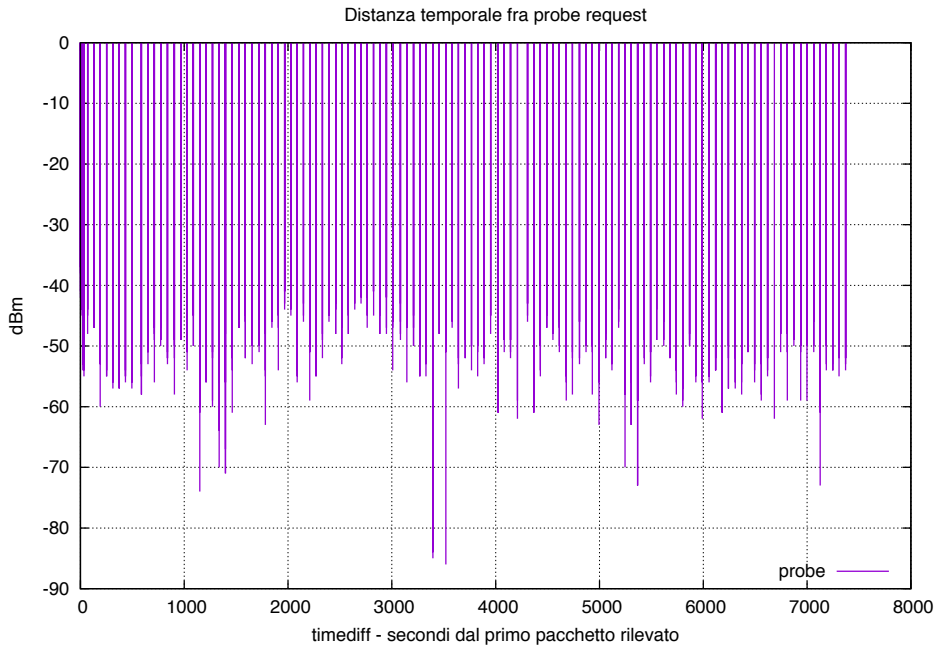


Figura 3.3: Probe nel tempo - campione

Osservando la distribuzione di probabilità in figura 3.2, si può intuire con più dettaglio il pattern del grafico mostrato precedentemente. La distribuzione è, infatti, una bimodale e mostra come la stragrande maggioranza dei probe inviati siano spediti sotto forma di *burst* con un delta temporale ridottissimo (1° moda). Tali burst sono inviati all'incirca una volta al minuto (2° moda) ed ognuno è composto mediamente da circa 4 pacchetti.

Nelle considerazioni future, tale valore verrà tenuto in considerazione come lower bound per il traffico generato dai device.

3.2.1 Servizi di geolocalizzazione Android e Probing

Il sistema operativo Android ha mostrato in fase di analisi una peculiarità: disattivando le interfacce WiFi ma lasciando operativi i servizi di

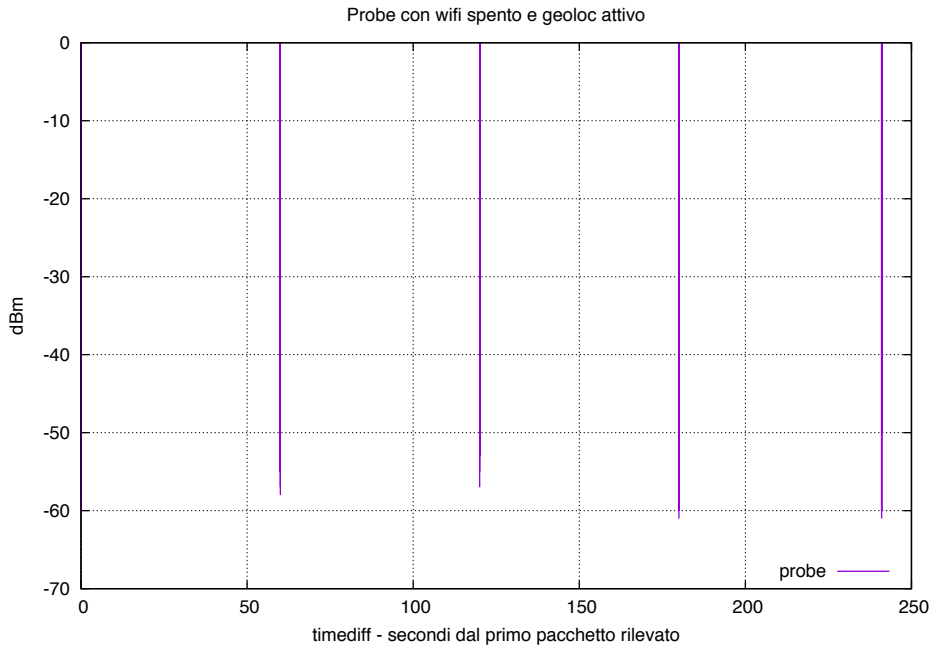


Figura 3.4: Probe - Wifi spento - Geoloc Service abilitato

Geolocalizzazione Google, periodicamente, vengono inviati probe sulla rete. Tale comportamento non è stato rilevato con altri dispositivi equipaggiati con Microsoft Windows Phone.

Anche in questo caso, vengono inviati burst composti ciascuno di circa 4 probe request ogni 60 secondi circa. Da notare che, anche in questo caso, il test è stato compiuto con il dispositivo in stand-by.

Questo risultato allarga il cerchio dei dispositivi rilevabili in quanto non è condizione necessaria che l'interfaccia wifi sia espressamente abilitata.

3.2.2 Canali adiacenti: Attenuazione

I canali 802.11 sulle frequeza 2.4GHz sono sovrapposti esattamente, come è possibile notare in figura 2.4. Volendo realizzare uno strumento di geolo-

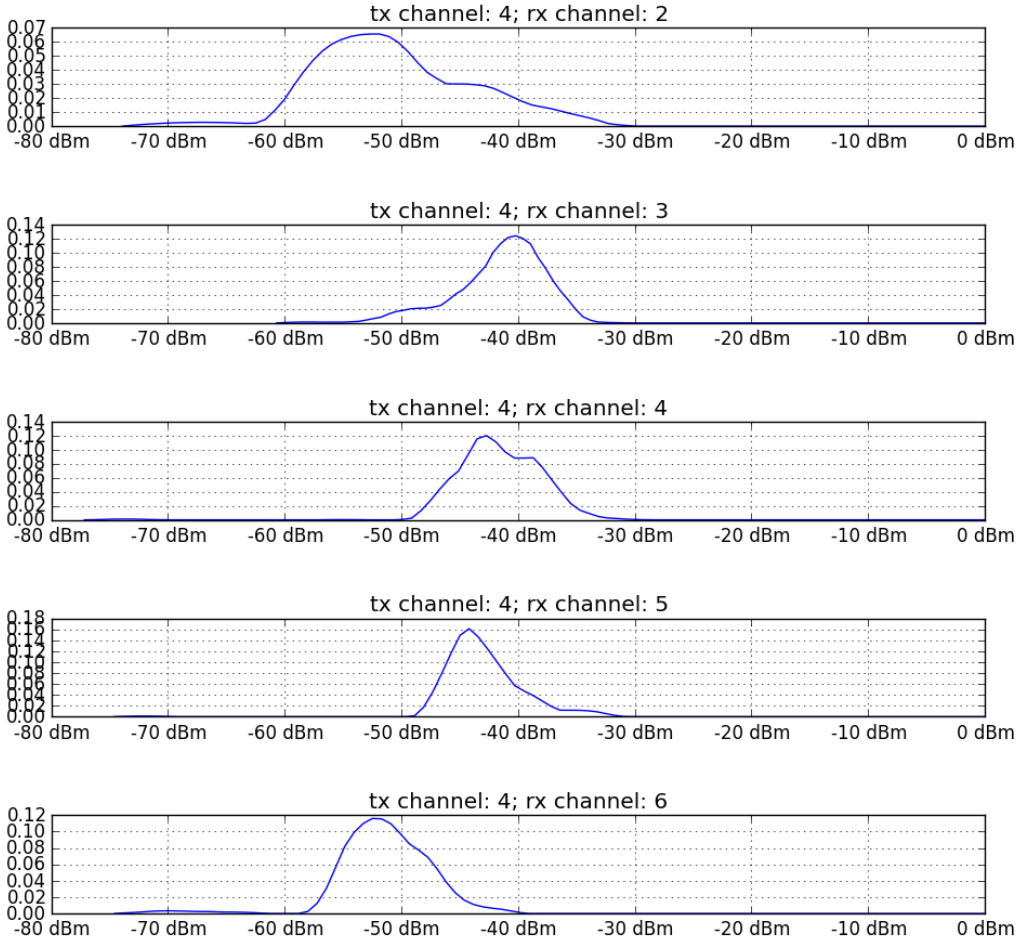


Figura 3.5: PDF - intensità segnale su canali adiacenti.

calizzazione basato sulla potenza del segnale ricevuto, è necessario tenere in considerazione cosa comporta questa configurazione.

L'energia irradiata da un trasmettitore dovrebbe seguire, secondo lo standard, gli schemi in figura 3.6 (per 802.11b) e 3.7 (per 802.11g/n) [18]. Prendendo ad esempio, considerandone la diffusione, 802.11g, a 11MHz dalla portante, il segnale emesso dovrebbe avere una attenuazione di -20dB, mentre a 22 Mhz, dovrebbe essere attenuato di -30dB rispetto al segnale di intensità massima, a 40MHz, il canale è attenuato di soli -40dB.

Occorrerà tenere conto di queste considerazioni qualora si preveda di

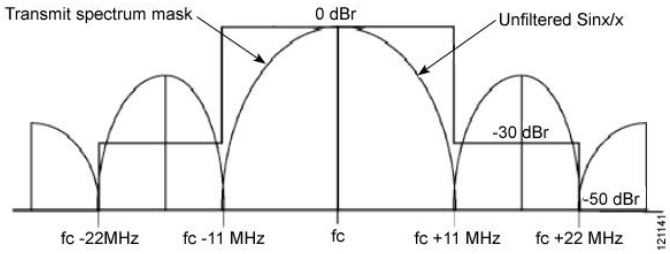


Figura 3.6: Spectral Mask (802.11b)

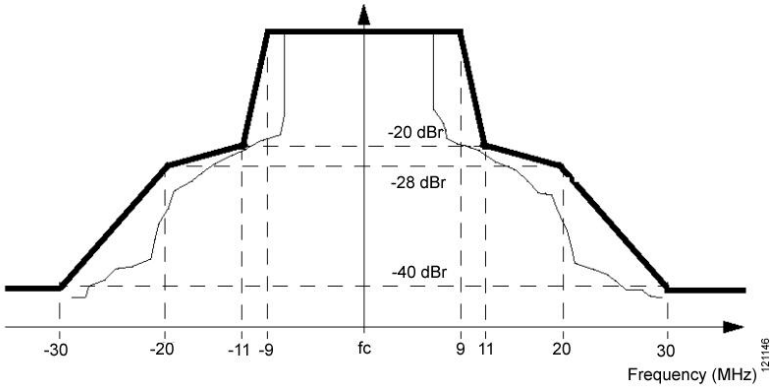


Figura 3.7: Spectral Mask (802.11 g)

considerare per la localizzazione anche pacchetti ricevuti da canali adiacenti.

La figura 3.5 mostra delle prove empiriche con un emittente sul canale 4. Si può osservare come le osservazioni effettuate, eseguendo un dump del traffico sui canali 3 e 5, non mostrino significative variazioni. Questo perché distando appena 5MHz dalla portante del canale 4 sono ancora nella fascia non attenuata della maschera spettrale. La distribuzione delle potenze con cui sono invece stati catturati i pacchetti sui canali 2 e 6, distanti 10MHz dalla portante dell'emittente, mostrano attenuazioni di circa 15 / 20 dB come da standard.

3.3 Power Management

Ulteriore punto da considerare come focale nei dispositivi mobili è la gestione della batteria. L'autonomia dei dispositivi è considerata, in fase di acquisto, praticamene alla pari con i servizi offerti dal dispositivo stesso. Si ha però un *trade-off* fra durata della batteria e servizi in *fore/back-ground*. Maggiore è la quantità di servizi erogati e di risorse utilizzate, minore è la durata della batteria. Tuttavia i vari vendor implementano delle soluzioni per tamponare questo fenomeno.

Incentrando l'attenzione sul wifi, gli aspetti impattanti sul prototipo da sviluppare e su cui si presuppone che parte dei sistemi di risparmio energetico facciano riferimento sono due: la gestione della potenza trasmissiva in base alla distanza dall'AP e la medesima in base alla batteria disponibile.

Nei paragrafi che seguono, vengono presentati test empirici per intuire possibili utilizzi in queste direzioni.

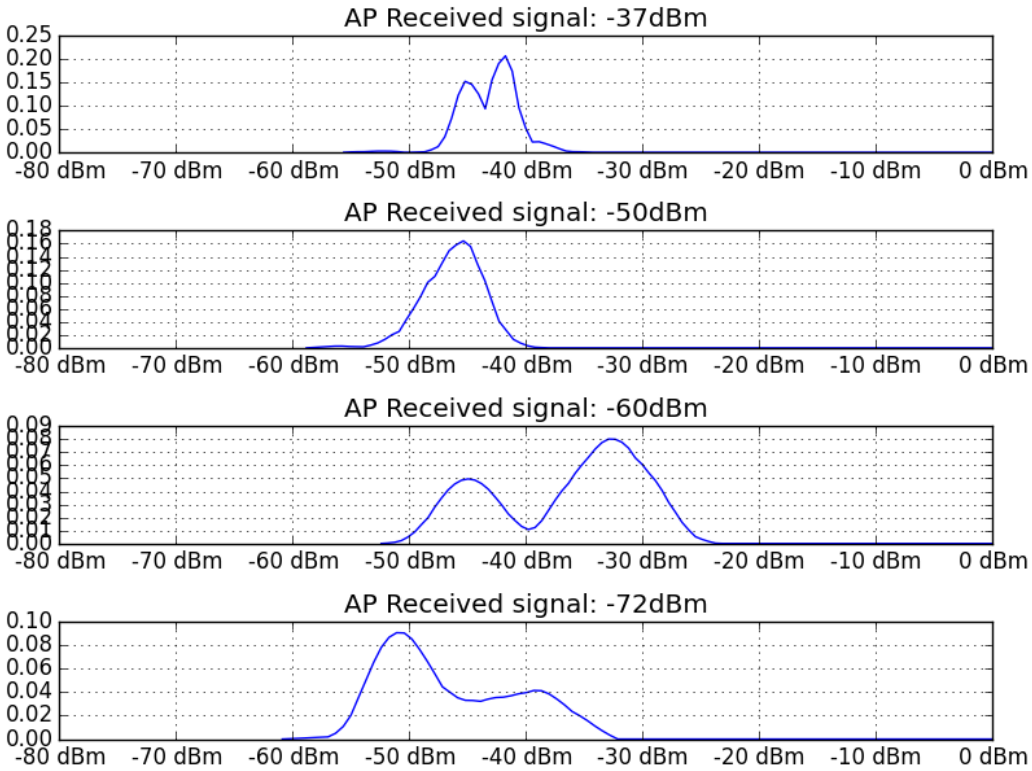
3.3.1 Potenza TX al variare del segnale RX

Il primo dei test presentati cerca di analizzare il comportamento di un emettitore al variare della distanza, misurata in termini di RSSI, dall'AP di riferimento.

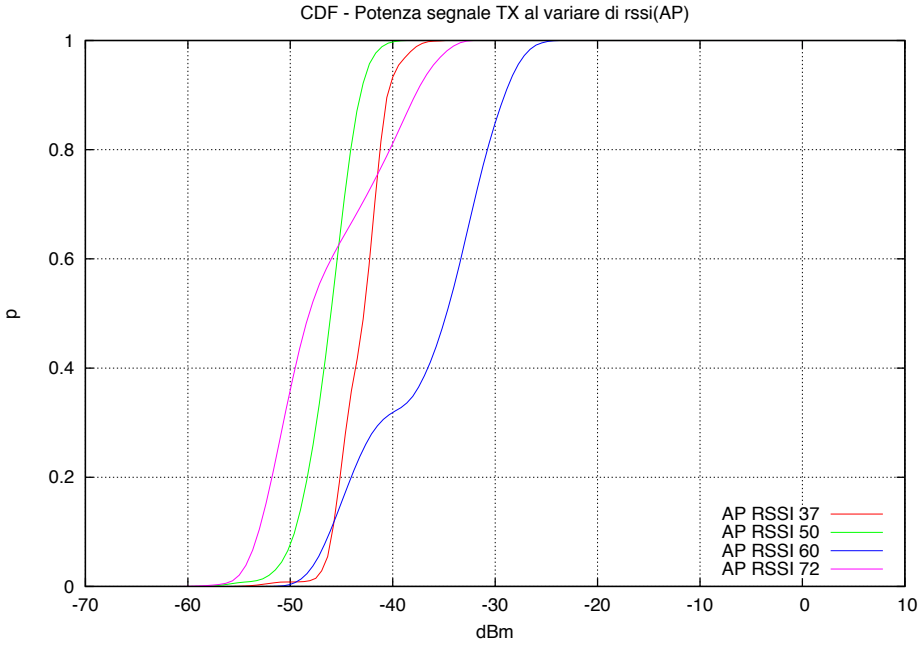
E' stato condotto posizionando una "sonda" (i.e: un dispositivo con interfaccia in monitor mode in grado di catturare il traffico) sempre alla medesima distanza dall'emettitore e variando la distanza di ambedue dalla BS. Sul dispositivo mobile è eseguita una applicazione che effettua 100 richieste icmp (*ping*) ad un altro nodo del DS ma esterno al BSS.

Per ogni sessione, è stata calcolata la distribuzione di probabilità (figura 3.8a) e la funzione di ripartizione (figura 3.8b) dei pacchetti catturati dalla sonda limitrofa.

Dai grafici, è possibile evincere come all'aumentare della distanza le medie si discostino al massimo di 10dB l'una dall'altra. Tuttavia, ad aumentare significativamente è la varianza. Le curve delle distribuzioni per i pacchetti



(a) PDF - Potenza TX al variare della potenza con cui si vede AP di riferimento



(b) CDF - Potenza TX al variare della potenza con cui si vede AP di riferimento

Figura 3.8: Potenza TX al variare del segnale RX da AP

inviati quando l'AP era visto con *-60* e *-72 dBm*, con una varianza pari a *30 - 35,* sono più "larghe" rispetto a quelle dei pacchetti inviati con un segnale RX dall'AP minore di *50 dBm*, dove la varianza è pari a circa *5 - 6.*

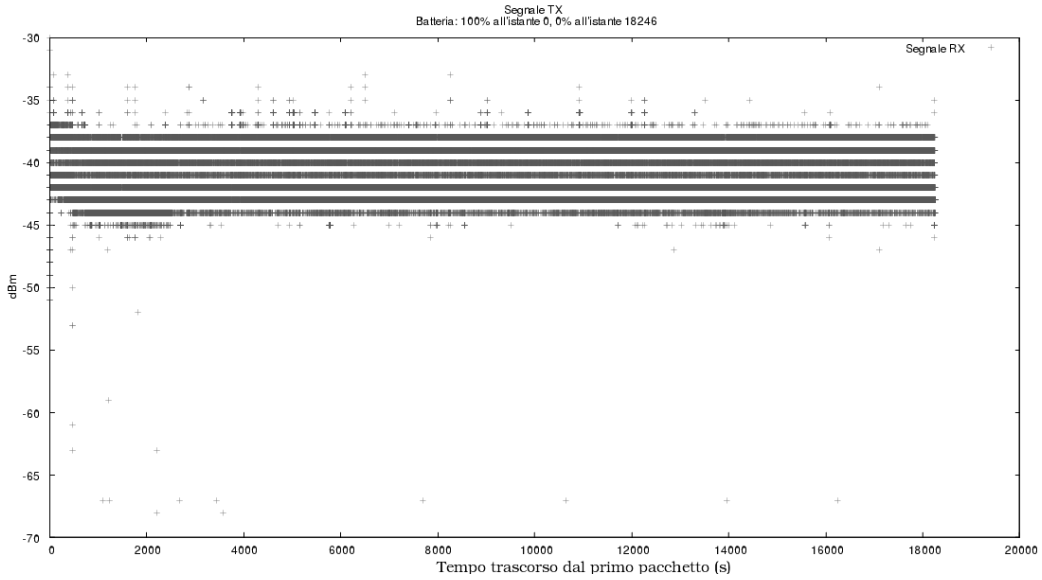
Questo comportamento può fornire un indice prezioso per dedurre la distanza di un device dalle sonde qualora questo produca abbastanza traffico in un tempo limitato. In questo modo potrebbe essere utile il calcolo di media e varianza del campione di traffico raccolto come aspetto caratterizzante.

3.3.2 Potenza TX al variare del tempo rimanente di carica

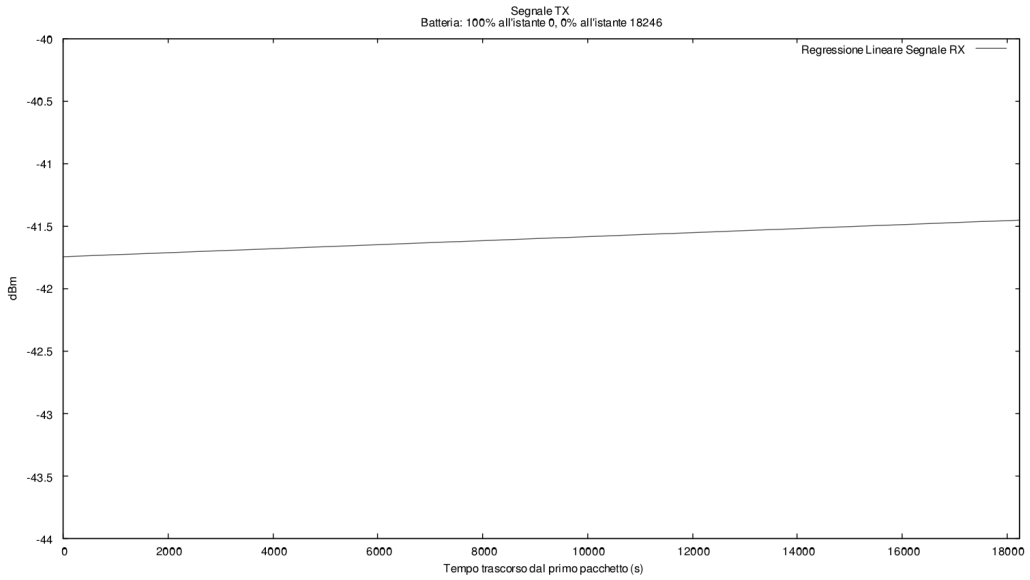
L'ultima delle analisi preliminari ha l'intento di verificare empiricamente se il software di diversi dispositivi mobili preveda una riduzione della potenza in trasmissione al diminuire della batteria residua. Il test effettuato prevede la presenza di una sonda e di un emettitore in posizioni prestabilite e fisse. Sull'emettitore gira un software che effettua le già viste richieste icmp ad un nodo raggiungibile attraverso il DS ma esterno al BSS. Sono state effettuate delle catture lunghe ciascuna un ciclo di vita delle batterie dei dispositivi.

L'immagine in figura 3.9a mostra come all'istante 0, dove la batteria era al 100% di carica, sia presente un pattern del tutto analogo a quello all'istante 18000 (5 ore dopo) con batteria prossima all'esaurimento. La seconda immagine, in figura 3.9b, rappresenta la regressione lineare dei dati già visti calcolata con Gnuplot (Listato: 3.1).

Grazie alla regressione è possibile notare che la pendenza m è prossima allo 0, per cui non è evidente che i meccanismi di risparmio energetico limitino la potenza dei dispositivi in trasmissione. Questo risultato gioca a favore della predicibilità in quanto diversamente sarebbe stato difficile, se non impossibile, distinguere un dispositivo con batteria scarica da uno effettivamente "lontano" esclusivamente grazie al suo RSSI. Questo perché un dispositivo con batteria scarica qualora dovesse trasmettere con una potenza minore, per questioni legate al risparmio energetico, sarebbe facilmente



(a) Dati RAW



(b) Regressione Lineare

Figura 3.9: Segnale RX in funzione della batteria residua

Algoritmo 3.1 Regressione Lineare

```
gnuplot> f(x) = m*x + b
gnuplot> fit f(x) 'onbattery.dat' using 1:2 via m,b

After 5 iterations the fit converged.

Final set of parameters
==========================

m               = 1.60575e-05
b               = -41.7452

gnuplot> set xrange[0:18246]
gnuplot> set yrange[-44:-40]
```

scambiabile con un device lontano considerando la bassa potenza ricevuta come indice di lontananza.

Capitolo 4

Stato dell'arte

Nel capitolo vengono introdotte le tipologie di soluzioni ad oggi presenti in letteratura per quanto riguarda la possibilità di geolocalizzare dei device mobili sfruttando le onde elettromagnetiche. Verranno dapprima elencate le varie tipologie di sistemi di geolocalizzazione elencando varie tipologie di classificazione in base agli aspetti osservati (e.g: sistema di coordinate, utilizzatore finale, ...). In secondo luogo saranno introdotte le principali tecniche algoritmiche di geolocalizzazione. Per concludere, saranno elencate le scelte adottate per l'implementazione del prototipo oggetto della tesi in base alle classificazioni presentate.

4.1 Sistemi di coordinate ed utilizzatore finale

I vari approcci esistenti in letteratura sono orientati a differenti tipologie di geolocalizzazione: *(i)* fisica (i.e: espressa in coordinate, e.g: "*degree / minutes / seconds*"), *(ii)* simbolica (i.e: espressa in linguaggio naturale, e.g: "in ufficio", "nell'ultima stanza a destra", ...), *(iii)* assoluta (e.g: sistema di riferimento a griglia condiviso da tutti gli oggetti) e *(iv)* relativa (i.e: espressa tramite distanza da un punto di riferimento noto) [14].

Un sistema di geolocalizzazione è composto da almeno due componenti separati, dove uno emette segnali elettromagnetici e l'altro ha il compi-

to di effettuare le misurazioni. Tipicamente, la maggior parte della logica applicativa è su questo secondo elencato.

I vari approcci sono classificabili tramite due macro-caratteristiche: *(i)* l'algoritmo di localizzazione utilizzato con le relative misurazioni necessarie e *(ii)* il livello fisico sul quale operano [23].

Un'ulteriore classificazione può essere fatta sull'architettura adottata e sull'utilizzatore finale dei dati ottenuti[10]:

remote positioning system: in questo caso l'emettitore è mobile e la logica applicativa è "remotizzata";

self-positioning system: l'unità di misurazione e, con lei, la parte algoritmica per la localizzazione sono residenti sul dispositivo mobile;

indirect remote system: questo tipo di architettura si ha nel momento in cui il dispositivo mobile "intelligente" invia ad un sistema remoto i dati inerenti alla classificazione che ha calcolato;

indirect self-positioning system: analogamente si ha questo tipo di soluzione qualora sia il sistema remoto, che ha calcolato le coordinate del dispositivo mobile, ad inviargliele.

Si noti che, per le ultime due soluzioni, è necessario che il sistema remoto e i dispositivi mobili possano dialogare.

Sono possibili altre distinzioni effettuate sulle tecniche algoritmiche per il calcolo della posizione. In questo caso, si hanno tre differenti possibili percorsi, visti nei paragrafi a seguire, a loro volta suddivisi in ulteriori correnti.

4.2 Geometrico

Vengono utilizzate proprietà geometriche in combinazione con altre proprietà dei mezzi trasmissivi proporzionali alla distanza fra punti.

4.2.1 Trilaterazione

Le tecniche descritte in seguito, misurano le distanze da punti di riferimento precisi. Tali distanze vengono dedotte da alcune grandezze proporzionali alle stesse. Il metodo di derivazione viene descritto nei punti seguenti di volta in volta.

4.2.1.1 TOA: Time of Arrival

La distanza fra il dispositivo mobile e i punti di riferimento noti è direttamente proporzionale al tempo di propagazione. Noto il tempo di propagazione, si risale per cui alla distanza grazie alla relazione tra spazio percorso e tempo impiegato. In un sistema self-positioning devono essere ricevuti tre segnali da altrettanti punti di riferimento distinti. Chiamando t_i il tempo di propagazione rispetto al beacon i, si ha: $t_i c = \sqrt{(x_i - x_D)^2 + (y_i - y_D)^2}, \forall i = 1, 2, 3, \ldots$ (posto c pari alla velocità della luce). Ponendo le equazioni a sistema, è possibile trovare i valori di x_D e y_D corrispondenti alle coordinate del device mobile. Questo sistema presenta quindi due peculiarità: *(i)* tutti i partecipanti devono essere fortemente sincronizzati e *(ii)* i pacchetti necessari per le misurazioni devono contenere un timestamp. Queste caratteristiche ne fanno un sistema utilizzabile principalmente per la tipologia self-positioning dove la logica è lato device mobile anche se ne è possibile l'adozione anche per sistemi di remote positioning.

4.2.1.2 TDOA: *Time Difference Over Arrival*

In questo caso, viene misurata la differenza, in termini di tempo, che uno stesso segnale impiega a giungere a 3 differenti misuratori. Vengono poi sfruttate le proprietà degli iperboli (nota: un'iperbole è definita come il luogo geometrico dei punti del piano aventi come costante la differenza delle distanze con i fuochi): in questo caso i fuochi sono i punti in cui sono posti i sensori, con N sensori si hanno per cui *N-1* iperboli.

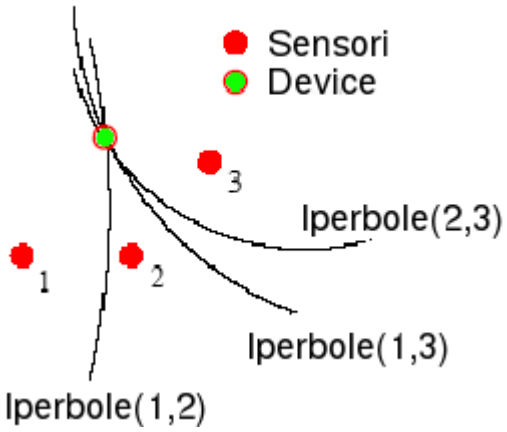


Figura 4.1: TDOA

Prendendo ad esempio un caso con 3 sensori posti in $\{x_1, y_1\}$, $\{x_2, y_2\}$ e $\{x_3, y_3\}$, come quello in figura 4.1, si ha che il tempo impiegato da una trasmissione iniziata da un device in $\{x, y\}$ per raggiungere ciascun sensore è dato da $T_s = \frac{1}{c}\sqrt{(x - x_s)^2 + (y - y_s)^2}$ con $S = \{1, 2, 3\}$ e c pari alla velocità della luce (da $v = \frac{s}{t}$). Assumendo ad esempio il sensore 2 come quello a cui per primo è giunto il pacchetto, lo si usa come origine del sistema (i.e: come punto di riferimento per la localizzazione) e si ottiene: $T_2 = \frac{1}{c}\sqrt{x^2 + y^2}$. Le differenze rispetto al punto di riferimento saranno quindi nella forma: $\tau_1 = T_1 - T_2 = \frac{1}{c}\left(\sqrt{(x - x_1)^2 + (y - y_1)^2} - \sqrt{x^2 + y^2}\right)$ e $\tau_3 = T_3 - T_2 = \frac{1}{c}\left(\sqrt{(x - x_3)^2 + (y - y_3)^2} - \sqrt{x^2 + y^2}\right)$. Mettendo queste equazioni a sistema, si ottiene un'unica soluzione che rappresenta il punto in cui si trova il device. Buona parte degli algoritmi basati su questa tecnica sono orientati all'ottimizzazione delle operazioni necessarie per approssimare la soluzione [23] o a rilassare il vincolo relativo alla sincronizzazione [22]. Rimane come in TOA il vincolo della sincronizzazione fra i beacon node ma non è più richiesto che i pacchetti contengano il timestamp di invio.

4.2.1.3 RSSB: *Received Signal Strength Based*

Le due soluzioni precedenti hanno un problema di fondo: sono molto sensibili al multipath fading, così come lo è anche la tecnica basata sull'angolo di arrivo vista successivamente. In questa tipologia di soluzione, si cerca di mettere in relazione l'attenuazione subita da un segnale ricevuto e la distanza che ha percorso. Per arrivare a questo risultato, in letteratura è possibile trovare modelli teorici ed empirici in parte visti in seguito. **La difficoltà, in questo caso, risiede nella modellazione del rumore dovuto al multipath fading e allo shadowing.** Alcuni approcci hanno adottato in simbiosi il *signal-to-noise-ratio* ed un meccanismo di calibrazione per l'implementazione dei propri algoritmi [33]. Una volta scelto il modello di attenuazione, vengono adottate le tecniche di multilaterazione viste per TOA. Infatti, ponendo $X = (X_r, Y_r)$ come le coordinate del device con locazione ignota, $X_i = (X_i, Y_i)$ come le coordinate del punto di riferimento i e r_i come la distanza fra il punto X e il punto X_i (calcolata tramite i modelli di attenuazione) si può impostare un sistema di equazioni con $i \in [1, n]$: $(X_i - X_r)^2 + (Y_i - Y_r)^2 = r^2$. Test dimostrano che la precisione in questo approccio aumenta, nel caso $n \geq 3$, escludendo i *beacon node* X_i per cui $r_i \gg r_j, \forall i \neq j$ [4].

I sistemi basati sullo studio dell'RSSI, sia per quanto riguarda gli approcci basati sulle proprietà geometriche che su altre proprietà, sono sicuramente fra i più gettonati in quanto non necessitano di hardware particolare.

4.2.1.4 RTOF: *Round-Trip Time of Flight*

Viene misurato il RTT di un segnale generato dai dispositivi mobili verso le sonde con relativo ritorno. Come per TOA è necessaria una compartecipazione alla logica applicativa da parte di device mobili e stazioni di misurazione. Inoltre è difficile escludere dall'RTT il tempo di processazione lato sonda. Questo tempo può essere trascurabile sulle lunghe distanze, ma non in situazioni indoor.

4.2.1.5 RSPM: *Received Signal Phase Method*

Conosciuto anche come *phase-of-arrival* necessita di stazioni in punti conosciuti che emettano segnali sinusoidali ad una frequenza nota f con offset pari a 0. La logica a questo punto è del tutto simile a quella di TOA/TDOA (a seconda che si stia implementando un sistema self-positioning o remote) con la differenza che il delay è calcolato in termini di frazioni di lunghezza d'onda. Necessita di hardware ad-hoc oltre che di una forte sincronizzazione e, necessariamente, di visibilità *line of sight*.

4.2.2 Triangolazione

4.2.2.1 AOA: *Angle of arrival*

Tramite hardware specializzato, si ricava l'angolo con cui un segnale è giunto ad un dispositivo mobile dai nodi beacon. Uno dei vantaggi è che in un sistema di coordinate 2D sono sufficienti 2 soli nodi emettitori, mentre ne occorrono 3 per un sistema 3D. Inoltre non è necessaria alcuna sincronizzazione. Di contro, l'accuratezza diminuisce all'aumentare della distanza ed è una tecnica molto sensibile al rumore. Infine l'hardware necessario è complesso e quindi costoso.

4.3 Analisi della scena / Fingerprinting

Nelle soluzioni basate sul fingerprinting, vengono studiate in una fase detta *offline* le caratteristiche dell'attenuazione introdotta dall'ambiente e subita dai segnali elettromagnetici. Questi dati vengono successivamente usati nella fase *online* per dedurre la posizione dei dispositivi confrontando, tramite opportune metriche/tecniche, i dati ottenuti a *runtime* con i dati raccolti in precedenza. Una buona parte delle soluzioni basate sull'analisi della scena operano sulla potenza con cui i segnali vengono ricevuti. Oltre alle tecniche presentate, basate su modelli probabilistici e machine learning,

ne Esitono altre come ad esempio quelle basate su programmazione genetica [15].

4.3.1 Modelli Probabilistici

Questi metodi cercano di inferire la posizione di un device sfruttando le leggi della probabilità. A titolo esemplificativo, il sistema *Horus* [38, 37, 39] è un sistema di self-positioning il cui funzionamento è diviso in due fasi. Anche in questo caso si hanno dei nodi beacon in punti noti $B = \{B_1, ..., B_n\}$.

In un primo momento, detto *offline phase*, vengono costruite campionando lo spazio le radio-map per ogni beacon node, che altro non sono che le distribuzioni di probabilità discrete (tramite istogrammi normalizzati) della potenza con cui in un determinato punto si ricevono pacchetti da ciascun beacon.

Nella fase *online*, viene stimata la posizione di un device grazie ad un vettore $S = \{s_1, s_2, ..., s_i, ..., s_m\}, m \leq n$ dove s_i è la potenza ricevuta dal beacon B_i. Determinare il punto in cui si trova il target, consiste nel trovare il punto in cui $P(B_1 = s_1, ..., B_m = s_m)$ è massimizzato. Considerato che ogni beacon node è indipendente dagli altri, tale probabilità è calcolabile con:

$$P(B_1 = s_1, ..., B_m = s_m) = P(B_1 = s_1) \dots P(B_m = s_m)$$

La locazione scelta per il target che ha generato il vettore s è quella che massimizza tale probabilità. Per ottimizzare le performance dal punto di vista computazionale, l'intera area viene divisa in settori chiamati *cluster* e vengono analizzati i soli punti e beacon all'interno del cluster d'interesse.

In questo approccio, si hanno tre peculiarità rispetto ai predenti: *(i)* è presente un insieme di dati collezionati in una fase di "addestramento", *(ii)* i vincoli relativi alla sincronizzazione si fanno molto più laschi e *(iii)* infine si introducono dei meccanismi di sampling e media[25, 26] tenendo in considerazione che misure singole si espongono troppo agli errori dovuti a fattori ambientali.

4.3.2 Machine Learning

In contrapposizione ai metodi probabilistici, alla stessa classe appartengono gli approcci basati su machine learning. Anche in questo caso, si adottano algoritmi che sono meno vincolati a meccanismi di sincronizzazione e che prevedono un set di dati collezionati *offline* per tarare l'intero sistema.
Degni di nota in quanto fra i più utilizzati, sono le soluzioni basate su *KNN* (*k-nearest neighbors*). Questi sono approcci che puntano a dedurre la posizione dei device imitando la maggioranza degli elementi a loro vicini una volta fissato uno spazio di coordinate. Uno dei più noti è RADAR [2]. L'applicazione, prevede la costruzione di un training-set tramite un'applicazione installata sui device mobili che invia in broadcast quattro datagrammi UDP al secondo, ciascuno contenente le coordinate *x,y*, l'orientazione *d* (Nord,Sud,Ovest,Est) ed un identificativo (n.c.d: per fare *merge* delle informazioni raccolte dai vari nodi misuratori). L'orientazione viene considerata in quanto definisce se un dispositivo ha visibilità ottica con i beacon. La presenza o meno della visibilità ottica varia le misurazioni di circa 5dBm. Con queste informazioni sono costruite una serie di tuple nella forma (x, y, d, ss_i) dove ss_i indica la potenza con cui il beacon i ha ricevuto il pacchetto. Più precisamente, ogni tupla è formata mediandone una ventina. Da questi dati si ricava il training-set finale: ad ogni punto *x,y* è associato un vettore di proprietà $S = \{ss_1, ss_2, .., ss_i, ..., ss_n\}$con n pari al numero dei beacon.
Nella fase *online*, la localizzazione dei device viene effettuata restituendo le coordinate *x,y* associate al vettore S che ha distanza euclidea minima con il vettore *S'* dove *S'* è l'equivalente nella fase *online* di *S*. L'algoritmo corrisponde a quello presentato in sezione 5.2.2 con K=1. Va considerato che nella fase offline della versione originale, viene utilizzata la stessa applicazione adottata per la costruzione del training set.
Fra gli altri metodi basati su *machine learning*, in [3] è evidenziato come, con l'approccio presentato e basato su reti neurali, siano sufficienti 5 soli campioni per ottenere una accuratezza nell'ordine dei *3m*. Mettendo a confronto i risultati ottenuti con i vari metodi basati sull'analisi della scena, si può

osservare che gli approcci probabilistici spesso sono più resistenti al rumore [29].

4.4 Test di Prossimità

Questa categoria consente di implementare soluzioni di tipo *remote* e *self-positioning* con una discreta facilità. Sono tecniche sostanzialmente basate su griglie dense di nodi misuratori in posizioni fisse. Quando un dispositivo mobile è rilevato da uno di questi, viene geolocalizzato come nel suo intorno con un margine di errore dipendente dalle scelte effettuate (i.e: distanza fra le sonde, orientamento delle stesse, politiche in caso un device sia evidenziato da più sonde, ...). Tipicamente, queste soluzioni hanno device terminali completamente privi di logica applicativa.

4.5 Soluzioni commerciali

Alcune fra le soluzioni commerciali più adottate sono quelle proposte da *Aruba* e *Cisco*. Entrambe queste soluzioni presentano una architettura analoga in quanto elaborano i dati ottenuti dal wireless controller usato per gestire l'intera infrastruttura (architettura presentata in 2.2.4). Essendo soluzioni commerciali, ma soprattutto *proprietarie,* non si ha evidenza degli algoritmi utilizzati ma solo dell'architettura dal punto di vista sistemistico. Viene per cui presentata la sola soluzione proposta da *Cisco*.

4.5.1 Cisco MSE/PRIME

Una delle soluzioni commerciali disponibili per realizzare applicazioni *Context Aware* con l'ausilio della geolocalizzazione è dovuta a Cisco [19]. Viene sfruttata l'architettura vista, in 2.2.4, basata sull'uso di una infrastruttura gestita tramite Wireless Controller. Il meccanismo di geolocalizzazione può essere rinforzato con l'ausilio di RFID che il software è in grado di gestire.

I componenti principali sono:

1. Gli AP;
2. Il/I Wireless Controller;
3. Cisco MSE (*Mobility Service Engine*);
4. Cisco PRIME.

MSE è una appliance software basata su Linux che offre diversi servizi orientati alla "gestione delle posizioni", in particolare:

- Esegue algoritmi di geolocalizzazione;
- Mantiene informazioni di calibrazione;
- Invia notifiche in stile "push" relativamente ai posizionamenti o ad aree di interesse;
- Effettua statistiche e mantiene dati storici.

PRIME, è un punto di accesso all'intera infrastruttura. Infatti consente il management dei wireless controller e la possibilità di visualizzare i dati estrapolati da MSE. Per la gestione dei Wireless controller è utilizzato, in modalità lettura/scrittura, il protocollo SNMP visto in 2.1 .

Per la comunicazione fra MSE ed i Wireless Controller, è stato, invece, implementato un protocollo proprietario chiamato NMSP (*Network Mobility Service Protocol*) che a sua volta viene cifrato con SSL. In pratica, non permette la visibilità dei dati scambiati e del loro formato ed impedisce a terzi di creare applicazioni analoghe interfacciandosi con i WLC senza acquistare le licenze MSE.

MSE, tuttavia, non è interrogabile solo ed esclusivamente tramite PRIME, è infatti possibile sfruttare delle API SOAP/XML su HTTP con cui interfacciarsi ed ottenere i dati desiderati.

Per quanto concerne la parte algoritmica, essendo una soluzione proprietaria, non si ha una visibilità completa. Le macro-tecniche adottate sono:

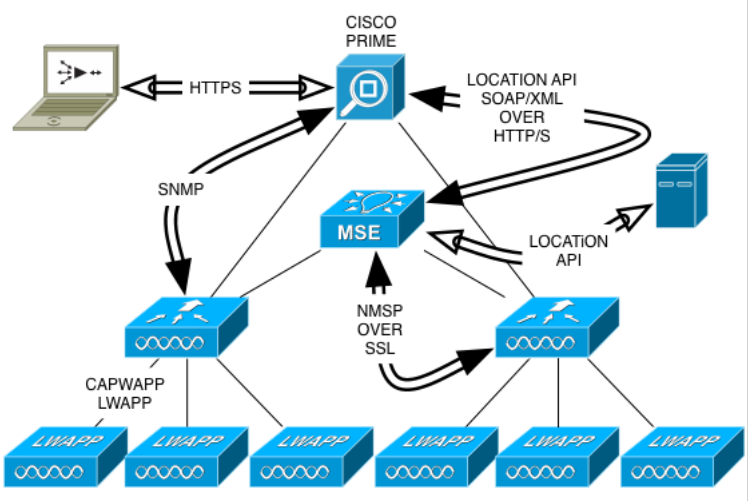


Figura 4.2: Architettura MSE

1. Triangolazione tramite RSSI in ambienti indoor;
2. TDOA (*Time Difference over Arrival*), in ambienti outdoor;
3. Una delle precedenti in combinazione con l'uso di tag in stile RFID.

Per la prima tecnica basata su RSSI, è necessario avere un minimo di 3 AP. In ogni caso sono consigliate coperture realizzate seguendo quanto segue:

1. BSA sovrapposte per almeno il 10% dell'area totale con le celle adiacenti;
2. AP non distanti più di 17-20 m;
3. Distanza fra un AP ed un altro costante fra tutte le coppie di AP;
4. AP posizionati in modo da non essere collineari;
5. Le forme geometriche formate dagli AP influenzano le prestazioni degli algoritmi di posizionamento, ad esempio:
   (a) Un posizionamento a triangoli equilateri è più efficiente di uno ad ottusangoli;

	Costo	Note
WLC	9000€	Modulo HA
WLC	10000€	con 50 licenze per gli AP
50 AP a/g/n	15000€	
MSE	2000€	con 50 licenze per gli AP
MSE	2000€	Appliance Virtuale
PRIME	15000€	con 500 licenze per i device
tot.	53000€	

Tabella 4.1: Stima costo CISCO WLC / MSE / PRIME

(b) Un posizionamento a quadrati è più performante di uno a rettangoli.

TAG Di comune accordo con altri vendor, è in fase di sviluppo uno speciale tag (in stile RFID) funzionante su 802.11 che invii dati telemetrici sulla rete e che consenta di fornire queste informazioni ad MSE praticamente *on-demand.*

ASPETTI COMMERCIALI Un aspetto negativo di questo tipo di soluzione è sicuramente quello economico. Una infrastruttura di questo tipo infatti ha costi notevoli sia sotto il punto di vista del materiale che delle relative licenze. Nella tabella 4.1 viene mostrata una stima di massima per una rete medio-piccola.

Capitolo 5

Scelte progettuali e Simulazioni

Sulla base di quanto visto, la soluzione, di cui verrà discussa la progettazione e l'implementazione, è basata su un sistema di remote positioning con test di prossimità dove questo non è un semplice test binario (rilevato/non rilevato da un nodo misuratore) ma è guidato da algoritmi basati su machine learning. Tale soluzione punta a migliorare la definizione di "nodo vicino" rispetto ai classici test di prossimità: in questo caso infatti, si vuole imporre che un device venga riconosciuto da una sonda solo se entro 5 metri di distanza dalla sonda stessa. Per capire se tale strada potesse essere percorribile, sono stati simulati dei test di classificazione sulla base di dati iniziali ottenuti tramite modelli di propagazione teorici che tenessero conto anche del rumore, o di parte di esso. Sempre sulla base dei dati ottenuti, sono state tratte indicazioni per la definizione dei parametri algoritmici utili anche per il prototipo finale.

Tale scelta è stata operata in quanto è l'unica che, ingegneristicamente, è meno soggetta a constraint legati all'uso di hardware specifico, alla necessità di una sincronizzazione stretta fra i vari attori del sistema o alla presenza di logica applicativa lato device.

Nei paragrafi che seguono, verranno presentati i modelli di propagazione analizzati, la definizione delle classi di vicinanza, l'algoritmo, con la relativa messa a punto dei principali parametri, per la classificazione dei device e i

risultati delle simulazioni.

5.1 Modelli di propagazione

Considerando alcune leggi che determinano l'attenuazione del segnale elettromagnetico nello spazio libero e il fatto che i dispositivi mobili, all'allontanarsi dagli AP di riferimento, aumentano, solo per certe trasmissioni, la potenza, come visto in 3.3.1, è stato effettuato un primo studio simulativo per individuare eventuali pattern riconoscibili da un sistema di *Machine Learning*. Tali modelli evidenziano, teoricamente e empiricamente, come la potenza di un segnale decresca proporzionalmente all'aumentare della distanza.

In letteratura esistono una serie di modelli recenti quali quelli basati su *ray tracing [30]* e su programmazione genetica [13]. Oltre a questi, sono presenti modelli più canonici (e.g: *multi-wall*) dove viene tenuta in considerazione l'attenuazione introdotta da *ogni* muro attraversato. Le problematiche di quest'ultimi sono legate al fatto che il fattore introdotto da ogni ostacolo presente deve essere conosciuto e quindi calcolato sul campo.

Le leggi prese in considerazione sono 2, rispettivamente: (5.1) l'*Equazione di Friis* [11], e (5.2) il *modello Log-Distance Path Model con Log-Normal Shadowing* (di seguito *LDPM*)[28].

$$\frac{P_r}{P_t} = G_t G_r \left(\frac{\lambda}{4\pi d}\right)^2 \tag{5.1}$$

Dall'eq. 5.1, è calcolata, assumento G_t e G_r pari ad 1, l'attenuazione in dB (nota anche come *Free Space Path Loss*) come:

$$A_{dB} = -10log\left(\frac{P_t}{P_r}\right) = -10log_{10}\left(\frac{4\pi d}{\lambda}\right)^2 = -10log_{10}\left(\frac{4\pi df}{c}\right)^2 = -10log_{10}\left(\frac{4\pi df}{c}\right)^2$$

$$P_r(d) = P_r(d_0) - 10n_p log\left(\frac{d}{d_0}\right) + \chi_{(0,\sigma)} \tag{5.2}$$

Dove, per l'equazione di Friis:

- P_r è la potenza ricevuta;
- P_t è la potenza trasmessa;
- G_t e G_r sono rispettivamente i guadagni dati dalle antenne di TX e RX;
- $\lambda = \frac{c}{f}$ è la lunghezza d'onda, dove c è la velocità della luce nel mezzo trasmissivo e f la frequenza a cui si sta trasmettendo;
- Si specificano f in Hz , d in m e c in m/s.

Dove, per il modello Log-Distance Path Model con Log-Normal Shadowing:

- d_0 è la distanza da P_t a cui è stata effettuata la misurazione empirica $P_r(d_0)$;
- n_p è il *path loss exponent*, cioè il fattore tipico di attenuazione (misurato empiricamente) dell'ambiente che circonda trasmettitore e ricevitore;
- $\chi_{(0,\sigma)}$ è una variabile casuale Gaussiana di media 0 e deviazione standard σ rappresentante il disturbo (espresso in dB) dovuto allo shadowing (scelto da una tabella calcolata in base a misurazioni). La modellazione del rumore tramite una variabile gaussiana avente media 0 e varianza espressa in dB, fa sì che la distribuzione risultante della potenza ricevuta sia una log-normale con valor medio nell'intorno di un valore dipendente dalla distanza.

$P_r(d_0)$ può essere calcolata con l'eq. di Friis (posto $d <= 1$) o empiricamente mediando più misurazioni.

Per quanto riguarda il modello LDPM, sulla base di statistiche effettuate su misurazioni empiriche, sono state compilate delle tabelle contenenti i valori del path loss exponent e del disturbo gaussiano. E' possibile osservare alcuni di questi valori in 5.1.

	Path Loss Exponent	Deviazione Standard
Vuoto Assoluto	2	0
Negozio medio / piccolo	2.2	8.7
Grande Magazzino	1.8	5.2
Uffici in muratura	3	7
Uffici open space	2.5	11.5
Fabbrica / LoS	2	3
Fabbrica / TX ed RX oscurati	3.3	6.8
Strada	3	7

Tabella 5.1: Modello LDPM: Path Loss Exponent Class

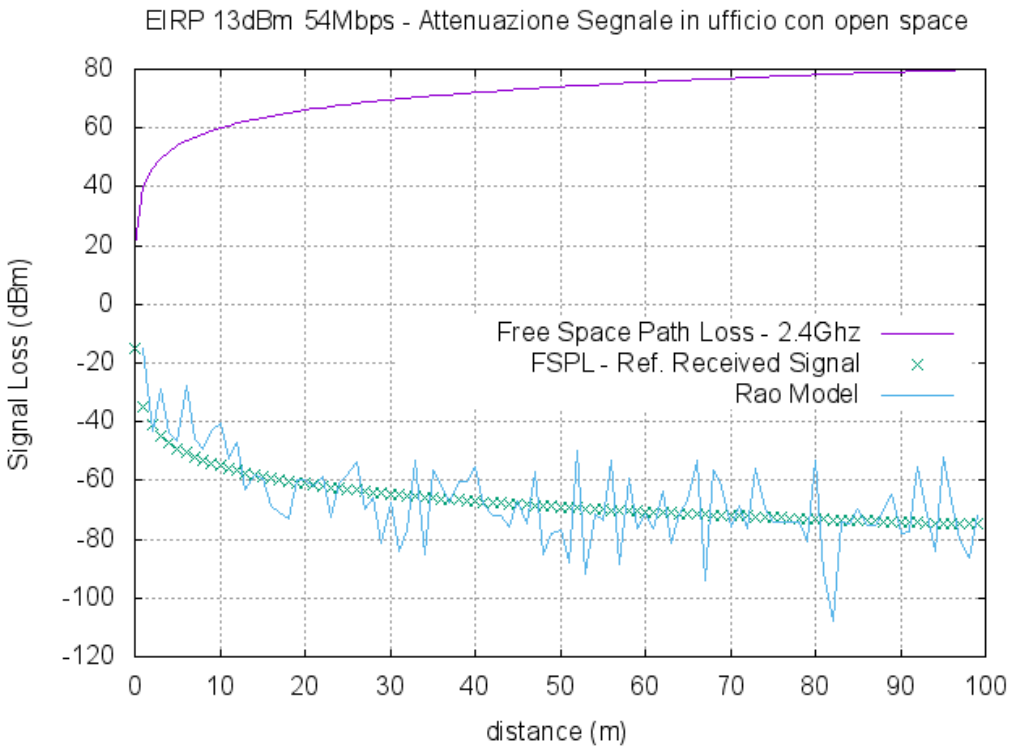


Figura 5.1: Friis e LDPM a a confronto - 1.

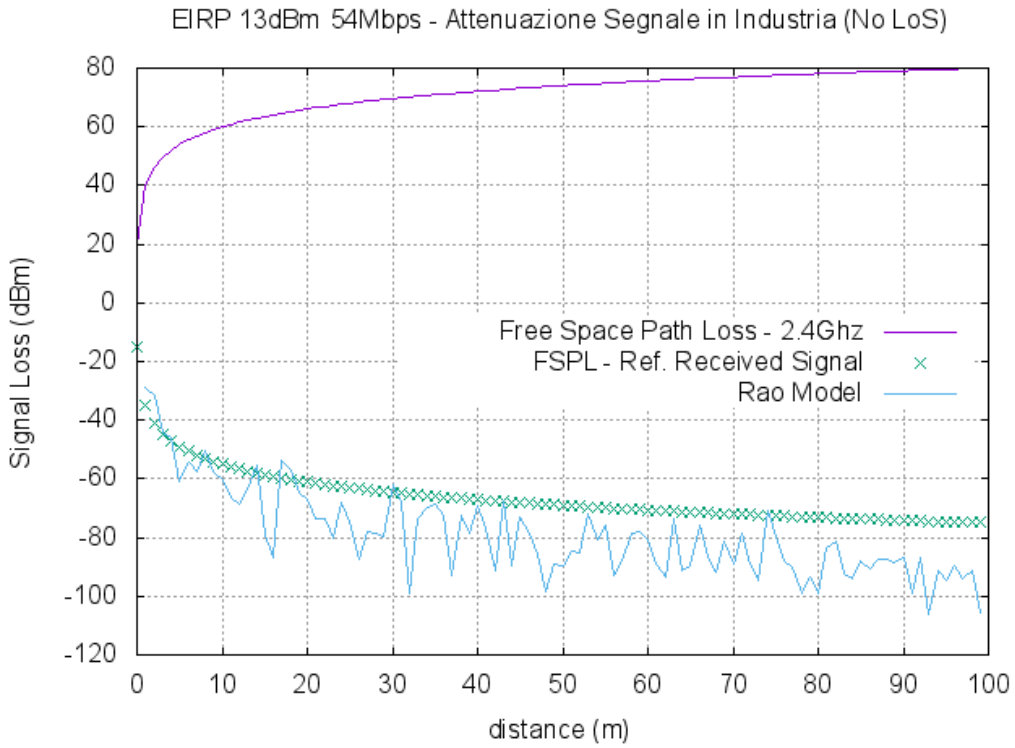


Figura 5.2: Friis e LDPM a confronto - 2.

Il terzo modello adottato sfrutta le analisi viste in 3.3.1. Supponendo, come elemento semplificativo, che la varianza cresca linearmente in modo direttamente proporzionale alla distanza dall'ap di riferimento, si ottiene:

$$\sigma(d) = m * d + \sigma(d_0) \tag{5.3}$$

Dove m e $\sigma(d_0)$ sono ottenuti tramite misurazioni empiriche. Negli esempi che seguono vengono considerati $m = 1.96$ e $\sigma(d_0) = 6$ ottenuti considerando i dati in 3.3.1.

Da cui:

$$P_r(d) = P_r(d_0) - 10n_p log\left(\frac{d}{d_0}\right) + \chi_{(0,\sigma)} + \chi_{(0,\sigma(d))} \tag{5.4}$$

L'idea di questo modello, è legata al fatto che all'aumentare della distanza aumentano le possibili fonti di rumore come vetri, muri, spigoli, ...

In realtà è possibile osservare dai grafici 5.3 e 5.4 che le fluttuazioni, dovute al rumore gaussiano aggiunto in funzione della distanza, iniziano a diventare significative solo intorno ai *-80dBm*, cioè vicino al limite in cui TX ed RX riescono a comunicare.

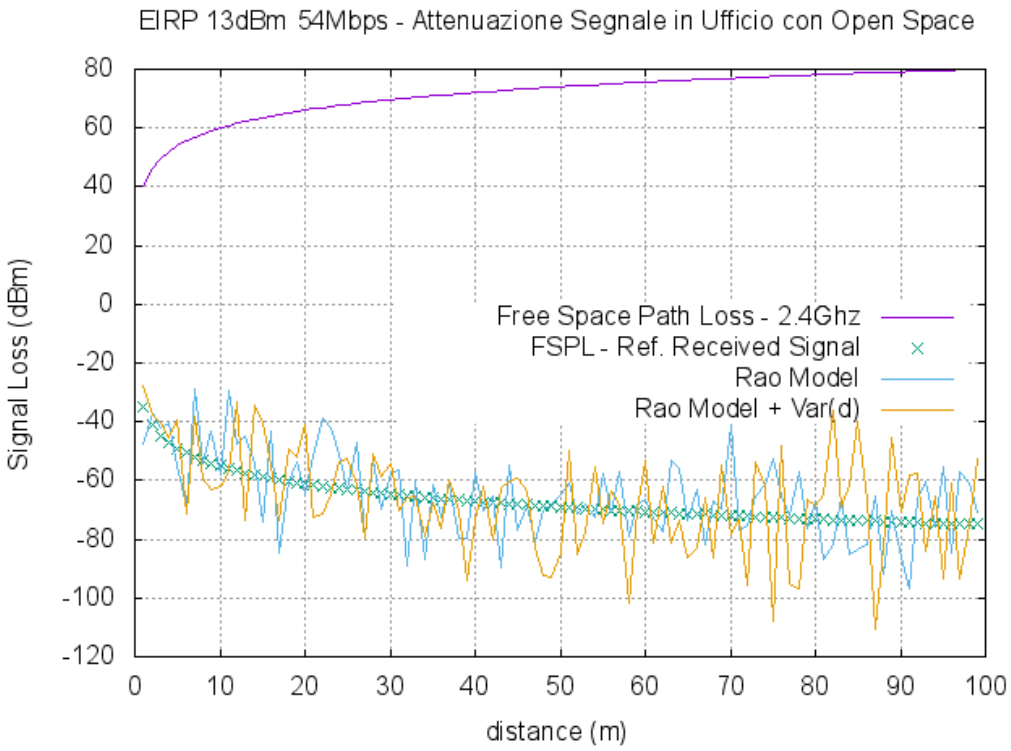


Figura 5.3: Friis, LDPM e varianza in funzione della distanza - 1

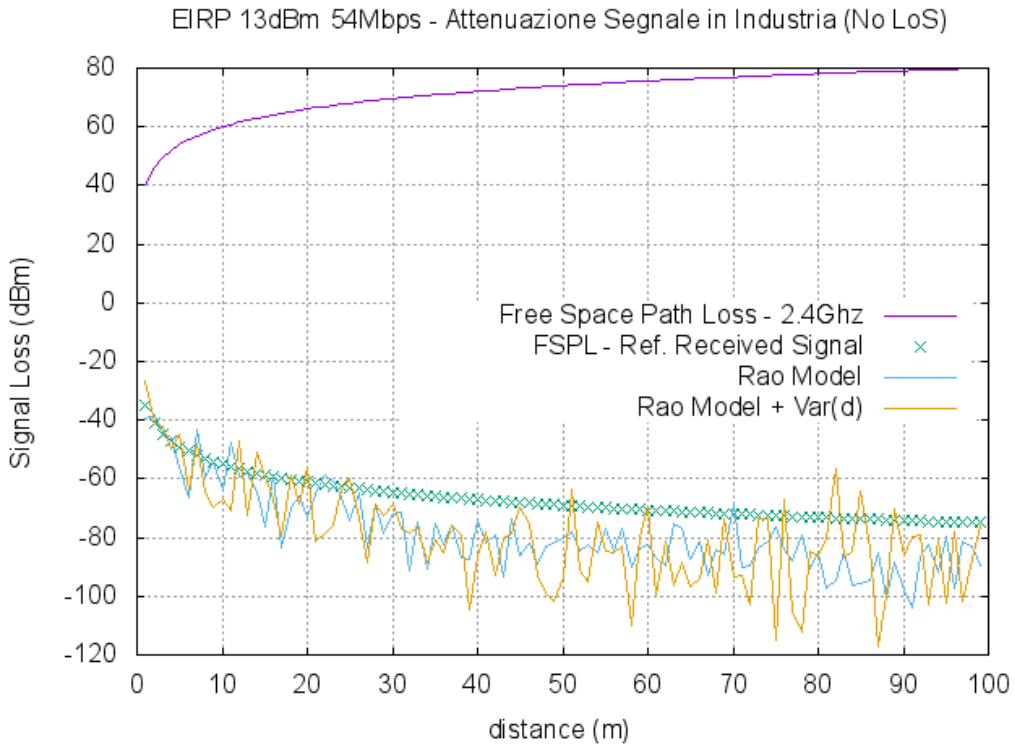
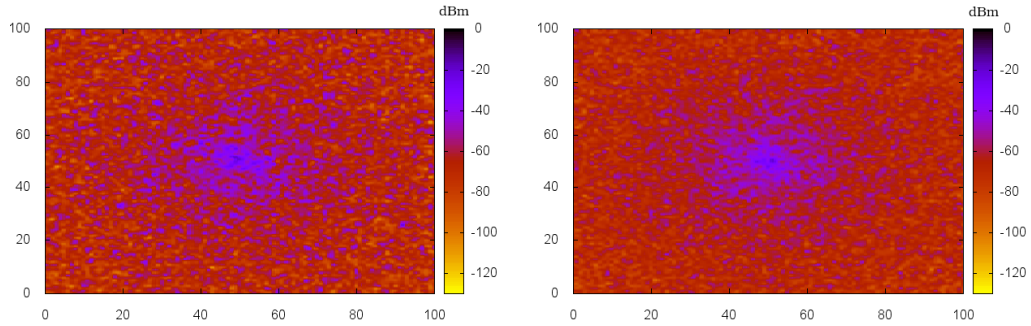


Figura 5.4: Friis, LDPM e varianza in funzione della distanza - 2

In figura 5.5 è possibile osservare il risultato prodotto dall'ultimo modello di attenuazione nello spazio. La distanza è misurata in metri e la sonda RX è fissata nel punto *<50,50>*. Ogni punto rappresenta il valore che è ricevuto dalla sonda posta al centro dal punto stesso. Le varie immagini sono state calcolate mediando un numero di campioni per punto da 1 a 6. Si nota come, utilizzando più campioni, l'effetto del rumore "emulato" dai termini gaussiani venga mitigato [26, 37]. Questo comportamento può aiutare la separazione in classi di vicinanza rendendo i confini più netti.

5.2 Classificazione con Machine Learning

Il Machine Learning è una corrente dell'intelligenza artificiale che sfrutta fatti noti a priori per dedurre caratteristiche specifiche di nuovi dati,



(a) 1 Campione

(b) 2 Campioni

(c) 3 Campioni

(d) 4 Campioni

(e) 5 Campioni

(f) 6 Campioni

Figura 5.5: Segnale ricevuto in *<50,50>* da ogni punto del piano *x,y* - Modello "LDPM + σ(d)" al variare dei campioni per punto.

osservando le relazioni fra le variabili analizzate.

Esistono differenti tipologie di Machine Learning che vengono introdotte sinteticamente di seguito:

Apprendimento Supervisionato Nelle soluzioni ad apprendimento supervisionato, il training set è composto da un insieme di coppie di esempi determinate da un oggetto di input e un valore di output desiderato. Gli oggetti in input sono tipicamente rappresentati con un set di variabili statistiche che ne descrivono le caratteristiche. Gli oggetti di output sono tipicamente valori appartenenti ad una enumerazione delle possibili classi.

Un algoritmo di apprendimento supervisionato genera un classificatore che è in grado di predire il valore di output giusto per ogni input valido.

Apprendimento non supervisionato In questo caso, il problema è quello di trovare strutture nascoste nei dati in input, in assenza di training set, da cui inferire una soluzione.

Apprendimento con rinforzo L'apprendimento per rinforzo si basa su stimoli dati dall'esterno a seconda delle scelte effettuate dell'algoritmo. Questa tipologia di algoritmi tenta di sfruttare questi stimoli forniti dall'agente esterno per migliorare le proprie decisioni alle esecuzioni successive. Pur sembrando simile alla categoria di apprendimento supervisionato, ne differisce in quanto non è previsto un training set iniziale. Questa tipologia può essere a sua volta suddivisa in due ulteriori categorie.

1. **Esperienza con apprendimento continuo**: Sono quegli algoritmi che dispongono di un meccanismo di valutazione di ogni scelta effettuata dall'algoritmo stesso. Sono, quindi, in grado di fornire un ACK positivo o negativo alla soluzione con cui poi verranno effettuate le opportune tarature per le scelte successive. Si considerino, ad esempio, i programmi di riconoscimento vocale o i programmi di OCR;

2. **Esperienza con addestramento preventivo**: Qualora la valutazione delle soluzioni proposte sia computazionalmente onerosa o impossibile per limiti fisici dell'implementazione (vedasi reti neurali su circuiti VLSI), si applica una prima fase di istruzione dell'algoritmo. Una volta ottenuto il grado di affidabilità richiesto viene cristallizzato e reso non più modificabile.

A questo punto, sulle basi dei modelli di attenuazione visti precedentemente sono state definite 3 classi da utilizzare con algoritmi ad apprendimento supervisionato per capirne le prestazioni. Le variabili che vengono utilizzate per descrivere gli elementi in input sono media e varianza di un set di 5 campioni.

5.2.1 Definizione Classi:

Le classi di vicinanza a cui appartengono i dispositivi, vengono definite in funzione della distanza dalla sonda. In immagine 5.6 è possibile osservare tale suddivisione presupponendo che la sonda sia posizionata in *<50,50>*.

Near(0): punti con una distanza euclidea al massimo pari a 5 dal punto *<50,50>* ;

Far(1): punti con una distanza euclidea compresa fra 5 e 7 dal punto *<50,50>*. La classe non ha un particolare interesse: è introdotta con il solo fine di "mediare" il passaggio fra la classe *near* e la classe *farther,* limitando di fatto il *flapping* fra queste classi nel processo di classificazione grazie al fatto che non risultano più direttamente confinanti. Gli elementi identificati in questa classe, nell'implementazione finale, non vengono classificati come *prossimi* alla sonda per cui la classe di fatto è adottata solo per emulare un meccanismo di isteresi;

Farther(2): punti con una distanza euclidea superiore a 7 dal punto *<50,50>*.

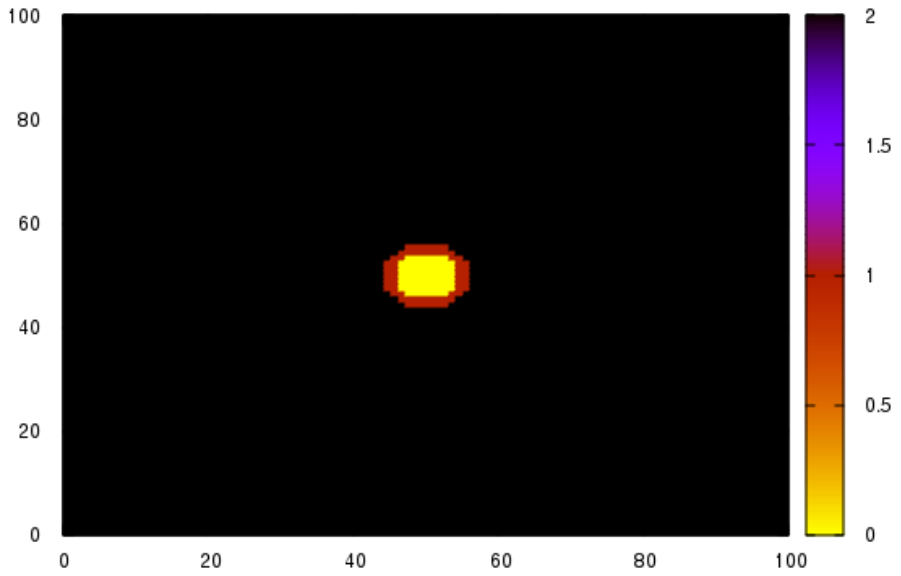


Figura 5.6: Classi di vicinanza

5.2.2 KNN

L'algoritmo adottato è il KNN (*K-Nearest Neighbors)* [9]. Tale algoritmo fa parte della classe degli algoritmi supervisionati ed è l'algoritmo più semplice fra quelli di classificazione. Scelto un parametro k, il suo comportamento è riassumibile per punti:

1. **Addestramento:** Gli oggetti del training set vengono posti sul piano cartesiano in base alle loro caratteristiche (nel caso in esame, media e varianza dei campioni);

2. **Classificazione:** I nuovi oggetti in input vengono a loro volta posizionati sul piano. La classe di questi ultimi viene decisa in base a quella che è la classe più frequente far i k elementi del training set più vicini. In pratica, nel processo di classificazione si ha lo stesso procedimento di

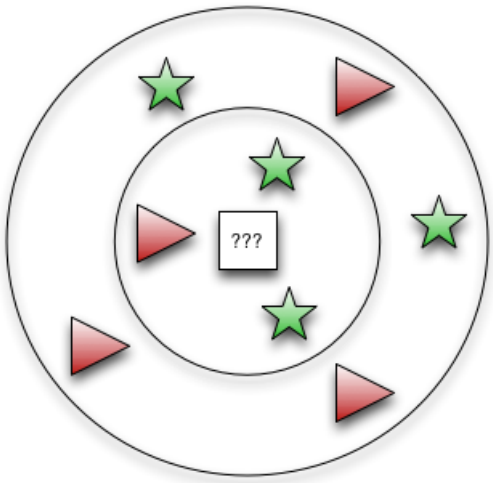


Figura 5.7: Esempio classificazione con KNN

una votazione: hanno diritto di voto per decidere quale classe assegnare all'elemento in input i k elementi vicini più vicini a questo ed ognuno vota sempre per la propria classe di appartenenza. Il peso di ogni voto può essere o uniforme o in funzione della distanza dell'elettore. Vince la maggioranza.

Prendendo ad esempio la classificazione con voto uniforme dell'elemento quadrato in figura 5.7 con k=*3* (area dentro il cerchio più piccolo), a questo verrà assegnata la classe stella perché fra i suoi 3 vicini più vicini 2 sono stelle ed uno è un triangolo.

Il parametro più importante dell'algoritmo è sicuramente k. Al crescere di k si riduce il rumore nella classificazione, ma, al tempo stesso, il criterio di scelta diviene più labile. La scelta di k avviene spesso tramite la tecnica di *cross-validation* del training set.

Una variante abbastanza comune consiste nel considerare come elettori non i k vicini più vicini, bensì tutti i vicini entro un raggio r dall'oggetto da classificare.

5.2.3 Training Set, tuning e testing KNN

Per l'implementazione dell'algoritmo di classificazione sono state utilizzate le librerie *Python Sklearn* [27]. Nai paragrafi che seguono, vengono esposte le considerazioni effettuate per la creazione del training set, presentate le operazioni intraprese per ottimizzare l'algoritmo ed i risultati ottenuti.

5.2.3.1 Costruzione del training-set

Per la costruzione del training set sono stati usati 60 elementi per classe. Gli elementi per le classi *Near* e *Farther* sono presi da sottoclassi realizzate in base alla distanza massima dal punto centrale. Ad esempio, per la classe *Near* sono presenti 30 elementi lontani al massimo 2.5m e 30 elementi con una distanza compresa fra 2.5m e 5m. Discorso analogo per la classe *Farther* dove sono scelti randomicamente 20 elementi con distanza rispettivamente compresa fra 7 e 30 m, 30 e 50 m, più di 50m. Questa suddivisione è stata effettuata per assicurare una presenza sufficiente di elementi di confine fra le classi.

Ogni punto compreso nel training set, è caratterizzato da media e varianza. Questa scelta è stata operata partendo dal presupposto che all'aumentare della distanza cresca anche la varianza dei campioni utilizzati per calcolare la entry. Nel caso del modello teorico "LDPM + σ(d)", questo comportamento è forzato.

5.2.3.2 Parametri simulazione - Cross Validazione

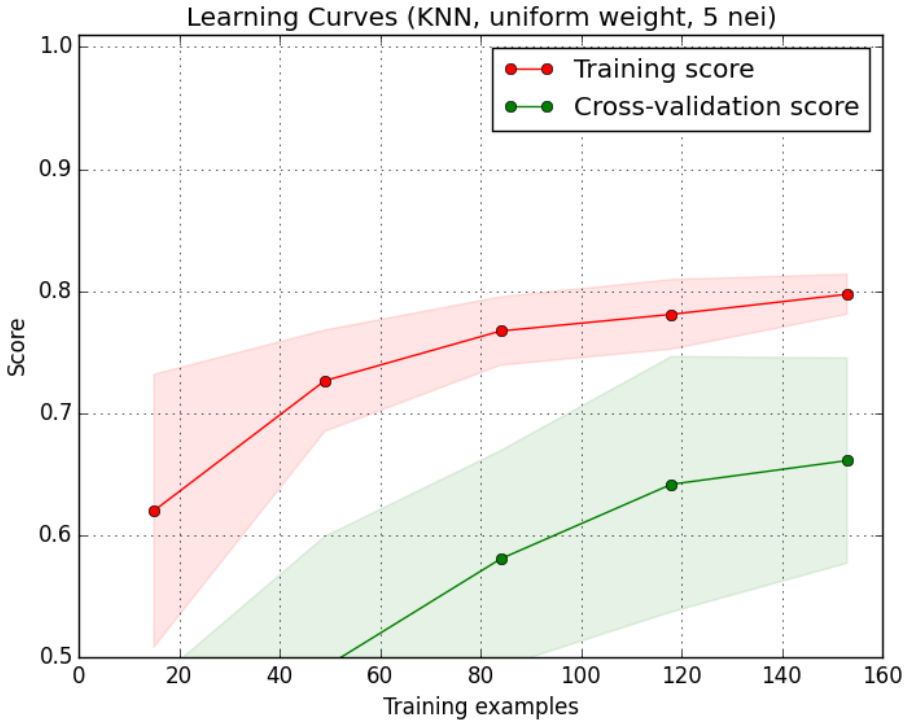


Figura 5.8: Cross-Validazione per scelta K - Pesi uniformi

Si vuole studiare l'errore nella classificazione al variare della varianza relativa all'ambiente (disturbo gaussiano dovuto allo shadowing). Per ogni valore attribuito alla varianza, vengono classificati 60 elementi (20 per classe) generati variando il *seed* del generatore di numeri casuali. Per ogni metrica definita in seguito, in ogni punto sono calcolati gli intervalli di confidenza al 95%.

In figura 5.9, si possono osservare i risultati della cross-validazione, al variare di *k,* per l'agoritmo con pesi proporzionali alla distanza. Si nota come al crescere di k non siano presenti miglioramenti e come la dimensione del training set sia opportuna. La curva di cross-validazione infatti, all'aumentare del numero di campioni, tende ad appiattirsi (i.e: il numero

di campioni è sufficiente) senza degenerare in termini di score (i.e: non è presente *overfitting*).

Nel grafico in 5.8 è, invece, mostrata la cross-validazione per l'algoritmo con pesi uniformi. In tal caso, è possibile osservare come non sia la scelta ideale per il tipo di classificazione da effettuare in quanto non solo si hanno risultati più bassi in termini di score rispetto al caso dell'algoritmo precedente per la curva di cross-validazione, ma anche per quella di training (validazione dei punti utilizzati per l'addestramento).

In via definitiva, vengono utilizzati i parametri:

- *k=5;*
- pesi dei singoli voti proporzionali alla distanza.

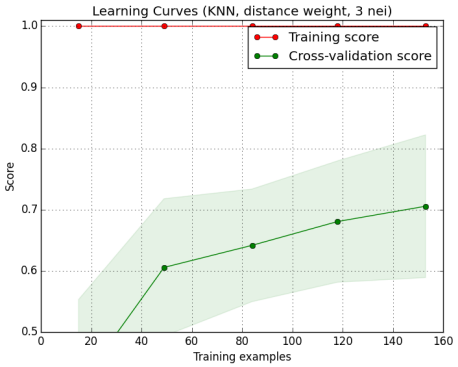
5.2.3.3 Metriche di misura e analisi risultati

Definiamo tramite lo pseudo-codice in listato 5.1 i concetti di vero positivo, vero negativo, falso positivo e falso negativo. Tramite la tabella 5.2, è definito il test binario (si distingue solo in base al fatto che un device sia *near* o no) su cui poi vengono definite le metriche di misura:

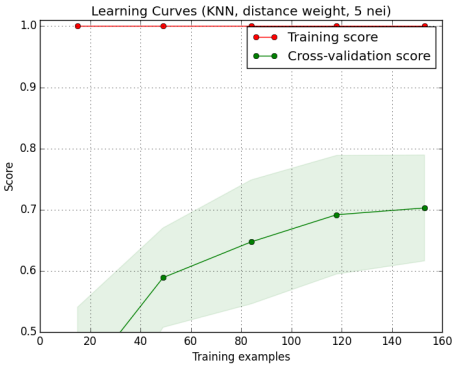
Out Class ↓	**In Class →**	**Near**	**Far/Farther**	
Near		TP	FP (errore di primo tipo)	*predittività positiva*
Far/Farther		FN (Errore di secondo tipo)	TN	*predittività negativa*
		Sensitività	*Specificità*	

Tabella 5.2: Tipologia errori per classe *near*

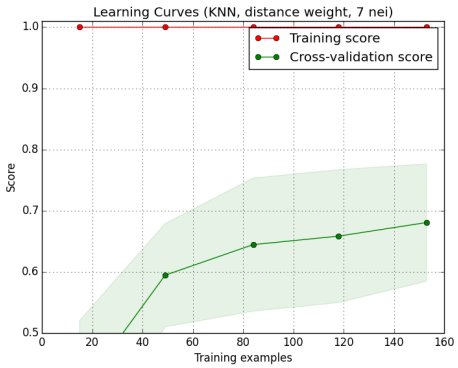
TPR (*True Positive Rate* $TPR = TP/(TP + FN)$) Indica la probabilità con cui il classificatore è in grado di assegnare una classe ad un oggetto effettivamente facente parte di tale classe. Tale parametro è indicato anche come *sensibilità*/*sensitività*;



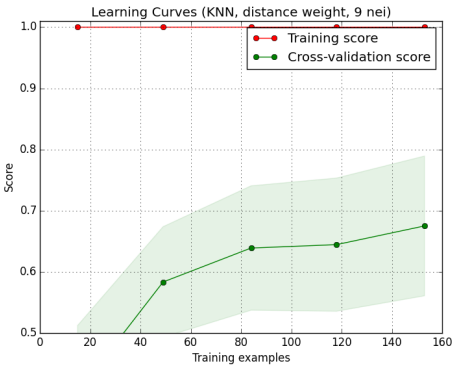
(a) K=3



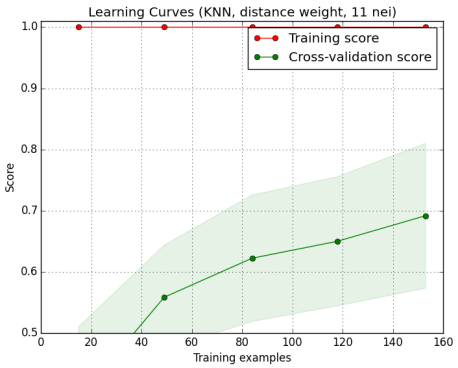
(b) K=5



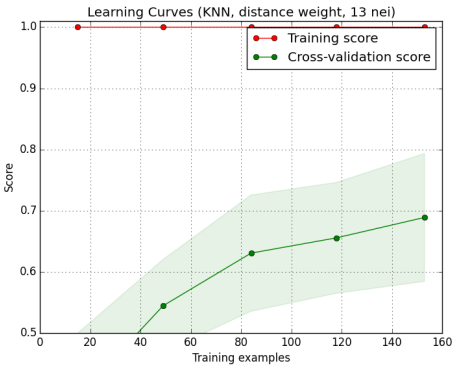
(c) K=7



(d) K=9



(e) K=11



(f) K=13

Figura 5.9: Cross-Validazione per scelta K - Pesi proporzionali alla distanza

Algoritmo 5.1 Veri/Falsi - Positivi/Negativi

```
outclass = classifier.classify([inputmean,inputvariance])

if inclass == outclass
        for class in classes
                if outclass == class
                        stats[class,'TP'] += 1
                else
                        stats[class,'TN'] += 1
else
        for class in classes
                if outclass == class
                        stats[class,'FP'] += 1
                elif inputclass == class
                        stats[class,'FN'] += 1
                else
                        stats[class,'TN'] += 1
```

TNR (*True Negative Rate* $TNR = TN/(FP + TN)$) Indica la probabilità con cui il classificatore è in grado di escludere l'appartenenza ad una classe per un oggetto che effettivamente non vi appartiene. Tale parametro è anche noto come *specificità*;

PPV (*Positive Predictive Value* $PPV = TP/(TP + FP)$) Indica la probabilità con cui un oggetto classificato come appartenente ad una data classe lo sia effettivamente. Tale parametro è anche indicato come *predittività positiva* o *valore predittivo positivo*;

NPV (*Negative Predictive Value* $NPV = TN/(TN+FN)$) Indica la probabilità che un device non classificato come appartenente ad una determinata classe non lo sia effettivamente. Tale parametro è anche indicato come *predittività negativa* o *valore predittivo negativo*;

FPR (*False Positive Rate* $FPR = FP/(FP+TN) = 1-TNR$) Probabilità di commettere errori di primo tipo;

FNR (False Negative Rate $FNR = FN/(FN+TP) = 1-TPR$) Probabilità di commettere errori di secondo tipo;

Accuratezza ($ACC = TP+TN/(TP+TN+FP+FN)$) Rappresenta la capacità dell'algoritmo di includere od escludere correttamente un punto come appartenente alla classe di vicinanza oggetto di studio.

Buoni classificatori hanno sensibilità (TPR) e specificità (TNR) elevate poiché l'obiettivo è quello di determinare tutti coloro che appartengono, non appartengono, ad una determinata classe. Queste proprietà infatti sono intrinseche del classificatore e non dipendono dai device esaminati. Contrariamente, i valori di predittività dipendono dalla frequenza con cui i device appartengono alle varie classi (i.e: più è rara l'appartenenza ad una determinata classe, minore sarà il valore predittivo positivo e maggiore la predittività negativa; *vice-versa,* più è comune l'appartenenza ad una determinata classe, maggiore sarà il suo valore predittivo positivo e minore la predittività negativa).

In figura 5.10, vengono indicati i risultati per la classificazione della classe *near* tramite le metriche definite. Nell'elenco seguente si traggono le conclusioni:

1. L'accuratezza (*acc*) passa dall'88% con varianza 1 al 75% con varianza 10, tuttavia si mantiene su valori superiori all'80% fino a quando la varianza non diventa maggiore di 8;

2. La curva relativa al *True Positive Rate*, o sensibilità, perde all'aumentare della varianza un 10-15%. Questo denota che aumentano i device *near* che non vengono classificati come tali. Anche tale curva rimane sopra, o intorno, al 70% fino ad una varianza pari ad 8;

3. Attenuazione analoga alla precedente anche per la curva relativa alla specificità, o *True Negative Rate*: anche in questo caso, si è nell'ordine

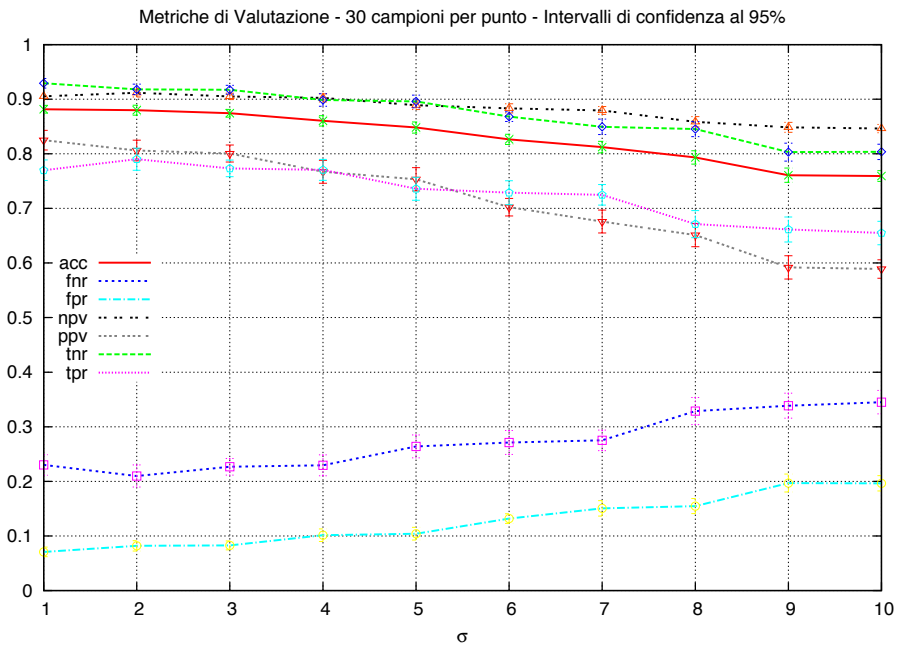


Figura 5.10: Valutazione simulazioni per la classe near

del 10-15%. Ciò testimonia un aumento dei falsi positivi nel processo di classificazione;

4. Tuttavia, le due curve precedenti partono da valori iniziali differenti, per cui l'incidenza dei falsi negativi rimane comunque maggiore rispetto a quella dei falsi positivi. Questo fatto è sottolineato anche dalle curve per il *false negative* e il *false positive rate*;

5. La curva del *positive predictive value*, o precisione, per all'aumentare della varianza il 20-25% del suo valore iniziale attestandosi nell'intorno del 60% con varianza pari a 10. Questo testimonia come la probabilità che un nodo classificato come *near* sia effettivamente tale diminuisca teoricamente con l'aumentare del rumore dovuto allo shadowing;

6. La curva che subisce la minor attenuazione, nell'ordine del *5%*, è quella legata al *negative predictive value*, o predittività negativa. Questo andamento è indice del fatto che la varianza non incide molto sulla classificazione degli elementi non *near* come tali.

Nel capitolo, sono stati visti alcuni semplici modelli di propagazione del segnale elettromagnetico. Sulla base di questi sono state definite delle classi di vicinanza per capire se l'algoritmo di classificazione supervisionata KNN fosse stato in grado di operare correttamente.

Sulla base delle metriche definite e dei risultati presentati, KNN è parzialmente in grado di distinguere, sfruttando media e varianza di campioni composti da 5 rilevazioni, i device entro i 5 metri da una sonda da tutti gli altri. Le metriche presentate nella sezione precedente, hanno evidenziato come l'algoritmo abbia una buona resistenza ai falsi positivi (sempre superiore all'80%) anche in presenza di rumore di tipo shadowing. Il *true positive rate*, cioè la capacità dell'algoritmo di individuare device *near*, non fornisce risultati altrettanto brillanti arrivando a poco più del 65% con varianza pari a 10 (pur restando in realtà quasi sempre sopra il 70%). Quest'ultimo dato è basso perché influenzato dai falsi negativi, cioè dai dispositivi "prossimi" non

rilevati. L'unico parametro non del tutto soddisfacente è il *positive predictive value*, tuttavia, essendo buoni i risultati per sensitività e specificità, che sono proprietà indipendenti dalla popolazione esaminata, si è proceduto alla realizzazione del prototipo reale.

Capitolo 6

Implementazione

Sulla base degli studi presentati nei capitoli precedenti, l'obiettivo è la realizzazione, basata su tecnologie *Open Source*, di un prototipo applicativo in grado di identificare i device rilevati come "vicini" a determinate sonde e che consenta la consultazione centralizzata dei dati raccolti.

Nei paragrafi che seguono, verranno introdotte le tecnologie adottate per la realizzazione delle sonde, l'architettura del prototipo, i test effettuati e i risultati ottenuti.

6.1 La sonda

6.1.1 Tecnologie Adottate

6.1.1.1 Raspberry B+

Come base per il progetto è stato adottato un *Raspberry B+*. Raspberry B+ è un *single-board computer* molto in voga, economica ed inizialmente concepita per attività didattiche. Basata su architettura ARMv11 (Set di istruzioni *RISC* a 32bit), supporta una serie di distribuzioni linux derivate da distribuzioni quali *Debian*, *Arch Linux* e *Fedora*, oltre a una serie di distribuzioni open non basate su Linux (e.g: *Risc OS*) .

Componente	
CPU	BMC2835 700 MHz ARM1176JZF-S core (ARMv11)
GPU	Broadcom VideoCore IV
Memory (SDRAM)	512MB
USB	4x2.0
Network	Ethernet 10/100 (Rj-45)
Potenza Assorbita	3W
Voltaggio in input	5V

Tabella 6.1: Raspberry B+

Nel caso specifico, è stata utilizzata la distribuzione *Raspbian 7*, derivata da Debian.

Raspberry B+ non implementa la tecnologia *PoE (Power Over Ethernet - 802.3af)* per cui, per facilitarne l'installazione dove tale tecnologia è presente, è stata adottato lo splitter TL-POE10R in grado di fornire i 5V a 2A richiesti dal Raspberry. L'alimentazione delle sonde tramite PoE, consente di poter spegnere e riavviare remotamente sonde in stati di *freeze* raggiungendo tramite ip di management gli switch a cui queste sono collegate e de-alimentando le relative porte.

6.1.1.2 Modulo WiFi TL-WN722N, monitor mode e radiotap header

Come modulo WiFi è stato adottata la scheda WiFi USB Tp-Link TL-WN722N dotata di antenna removibile con connettore *RP-SMA*. Il modulo, equipaggiato con un chipset Atheros AR9271, supporta gli standard 802.11b/g/n con un rate massimo di 150Mbps ed è in grado di operare sulle frequenze nell'intorno dei 2.4GHz. La scheda è equipaggiata con un *SoC* (*System-On-Chip*) dotato di RAM(160Kb), ROM(24Kb), Chipset WiFi e CPU (*Tensilica core (Xtensa LX2.1.0) 117 MHz*) in un singolo chip. La logica con cui è stato realizzato il chipset wifi è derivata da quella adottata

per il chipset *AR9285* [36][7]. Può trasmettere ad una potenza massima di 19dBm ed ha una sensitività pari a -96dBm.

Dal kernel Linux 2.06.35 è supportata tramite il modulo *ath9k_htc*. Il firmware adottato è anch'esso *open* e distribuito con più licenze in base alle sotto componenti (MIT License, CleraBSD, GPLv2). Inoltre, il driver supporta le *Linux Wireless Extensions* [34] e *nl80211/cfg80211* suoi successori per cui è in grado di esportare in *user-space* diverse statistiche fisiche e alcuni parametri di configurazioni. Fra le caratteristiche interessanti di queste API, v'è la possibilità di cambiare anche i parametri fisici (quale può essere ad esempio il canale adottato) senza dover far ripartire il driver.

La scheda wireless utilizzata supporta ovviamente la modalità *monitor mode* o *RFMON (Radio Frequency Monitor)*. In questa modalità, differentemente da quanto avviene mettendo la scheda in modalità promiscua, è possibile *catturare* tutti i pacchetti senza essere associati a nessun AP. Inoltre, in questa modalità vengono riportati dai tool di dump del traffico, tutti i pacchetti 802.11 che in modalità promiscua verrebbero mascherati come trame ethernet dal livello LLC.

E' possibile impostare l'interfaccia in monitor mode sfruttando *iw* (CLI per nl80211/cfg80211):

```
iw <devname> interface add <name> type <type>
```

In maniera del tutto analoga è possibile impostare il canale di ascolto:

```
iw dev <devname> set channel <channel>
```

Per impostare lo stato dell'interfaccia è possibile invece adottare i tool forniti da *iproute2*:

```
ip link set dev <devname> up
```

La WNIC in questione è di tipo *SoftMAC* per cui lo strato MLME (*Media access control Sublayer Management Entity*), in sostanza quello che mantiene l'automa della macchina a stati finiti di 802.11, è implementato in software tramite le API fornite da *mac80211*. Il vantaggio dell'uso di mac80211 risiede

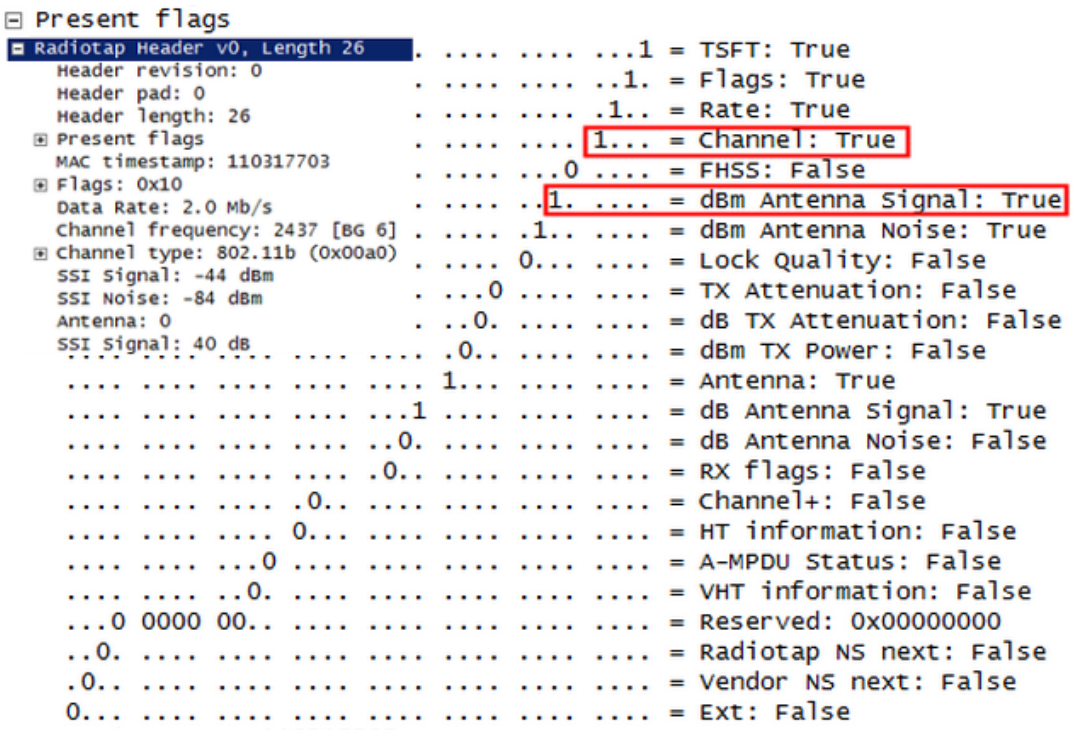


Figura 6.1: Radiotap

nella possibilità, una volta posta l'interfaccia in monitor mode e se supportato, come in questo caso, dall'hardware/driver, di fare *injection* nei frame 802.11 degli header *radiotap*.

Gli header radiotap sono un particolare tipo di header, non standardizzato, che consentono al driver di fornire in user space informazioni aggiuntive circa le caratteristiche del mezzo fisico nel momento in cui è stato ricevuto il pacchetto. Nel nostro caso è possibile osservare, in figura 6.1, come siano contenuti due elementi fondamentali: la potenza del segnale con cui viene ricevuto ogni singolo pacchetto ed il canale su cui è stato ricevuto. La WNIC esporta questi valori sono tramite i campi *radiotap.dbm_antsignal* e *radiotap.channel*.

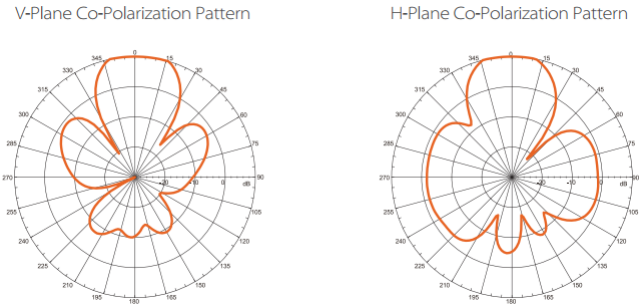


Figura 6.2: TL-ANT2414A Radiation Pattern

6.1.1.3 Antenna

L'antenna utilizzata è la TP-Link TL-ANT2414A [35]. L'antenna opera in modalità direzionale secondo il radiation patter in figura 6.2 e offre un guadagno di 14dBi a fronte dell'attenuazione introdotta dal cavo CFD-200 (*0.9db/m @ 2.5GHZ*) per il collegamento alla scheda wifi. Il guadagno alla scheda è quindi effettivamente di 9.5dBi. La scelta di una antenna direzionale è stata fatta per restingere il campo di azione della sonda e diminuire così ulteriormente il numero di falsi positivi per quanto concerne la classe *near*.

6.1.1.4 Valutazione Economica

L'hardware elencato (raspberry, splitter poe, antenna e relativo cavo, dongle wifi) ha un costo totale di circa 90€ iva esclusa e, contrariamente alle soluzioni proprietarie, si tratta di sole spese in beni materiali e non sono previsti ulteriori costi di licenza.

6.2 Architettura

L'architettura del prototipo prevede siano presenti sonde dislocate nell'ambiente con il compito di catturare ed effettuare le prime statistiche e infine in grado di comunicare con un server centrale incaricato di effettuare le classificazioni degli elementi ricevuti.

La classificazione è demandata ad uno o più server centralizzati in quanto è computazionalmente la parte più onerosa e le risorse a disposizione del Raspberry non sarebbero sufficienti per una elaborazione performante. Lo schema in figura 6.3 mostra, ad altro livello, il flusso seguito, prima all'interno della sonda, poi sul server di calcolo, per arrivare alla classificazione dei dispositivi. Ogni componente verrà analizzato nel dettaglio nelle sezioni successive partendo dai componenti software necessari per la cattura ed una prima elaborazione dei dati effettuata dalle sonde fino alla classificazione svolta lato elaboratore/i centrali.

6.2.1 Architettura Sonda

Le operazioni nella sonda seguono una logica *pipe and filter*. Prima di analizzare tale architettura è utile analizzare le fasi di boot e alcune note. Una sonda è identificata univocamente tramite il mac address della propria interfaccia ethernet. Le sonde al boot dell'applicativo di sniffing eseguono nell'ordine:

1. Risoluzione a dns del nome "*wiglprovisioner.*<localdomain>" (n.b: il server wiglprovisioner è uno dei server *collector*);
2. provisioning configurazioni (esempio in Listato 6.1) tramite richiesta snmp all'ip ottenuto al punto 1;
   (a) Server *"collector"* a cui inviare i dati (IPv4|"node a dns");
   (b) Lista dei canali su cui effettuare *channel-hopping* (interi separati da virgola)*;*
   (c) *dwell time* (ms): tempo di attessa fra un salto di canale ed il successivo.
   (d) **formato configurazioni**:

<id sonda;IPv4|NAME;channel,channel,channel,...;dwell>

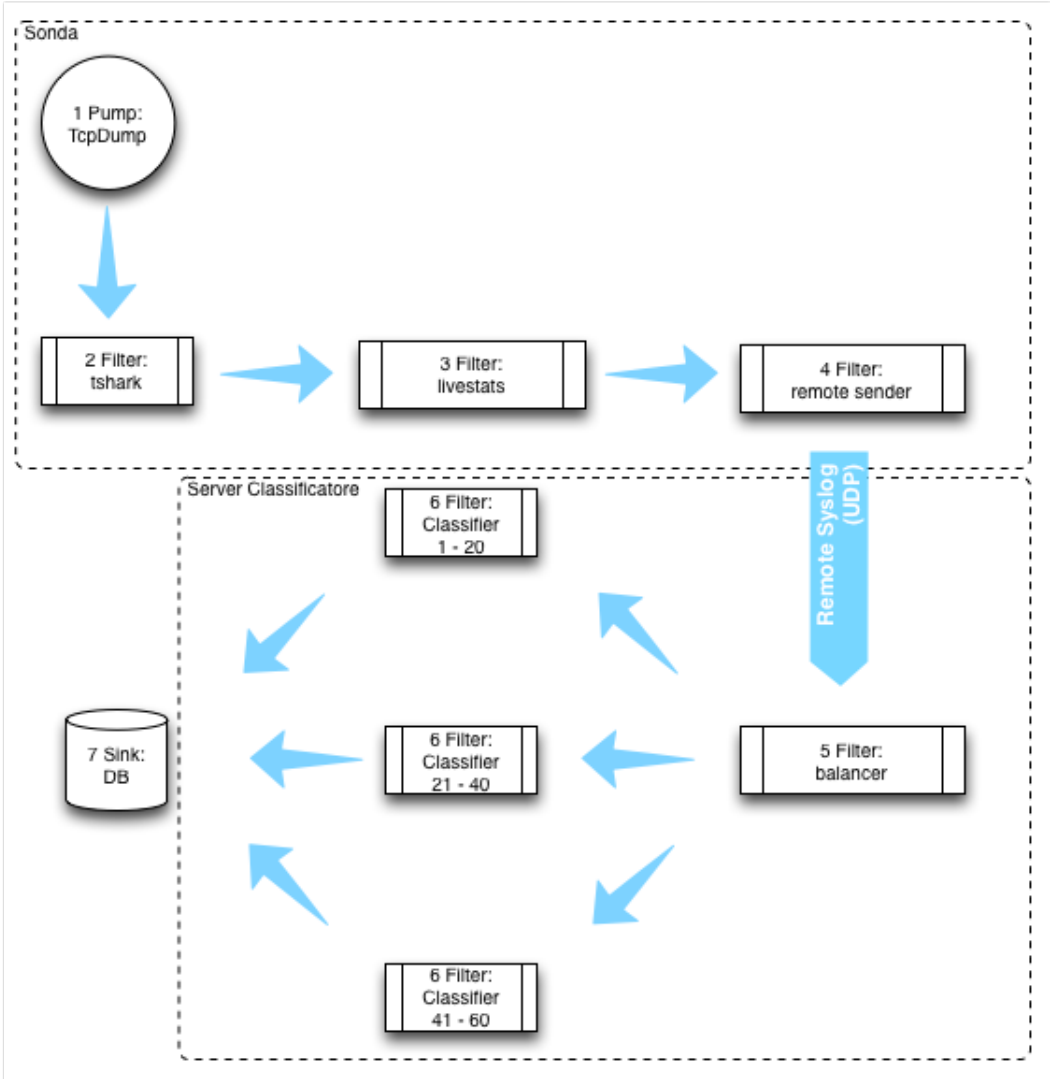


Figura 6.3: Architettura

Algoritmo 6.1 Provisioning sonde via SNMP

```
sondauno:~# snmpwalk -v2c -c public wiglprovisioner \
        'NET-SNMP-EXTEND-MIB::nsExtendOutLine."getwiglsentinelconf"'

NET-SNMP-EXTEND-MIB::nsExtendOutLine."getwiglsentinelconf".1
        = STRING: b8:27:eb:a9:77:eb;wiglservera;4,6,8;100
NET-SNMP-EXTEND-MIB::nsExtendOutLine."getwiglsentinelconf".2
        = STRING: b8:27:eb:b8:64:fa;wiglserverb;4,6,8;100
```

3. configurazione interfaccia per dump traffico;

   (a) imposta interfaccia in monitor mode tramite *iw;*

   (b) esegue script per channel hopping.

4. Avvio tool di dump/analisi del traffico .

Il tool di analisi del traffico è composto da una sorgente e quattro filtri in pipe, analizzati nell'ordine ordine:

6.2.1.1 tcpdump

E' il classico traffic dumper Linux. Viene istanziato per catturare sulla sola interfaccia wireless in monitor mode, per leggere i primi 68 byte di ogni pacchetto e per scrivere i dati su stdout. Vengono letti solo i primi 68 byte in quanto sufficienti per le informazioni d'interesse, cioè gli header, e in modo da ottimizzare la fase di cattura. L'opzione *-n* ha il solo compito di evitare la risoluzione degli ip individuati. Quella che segue il comando utilizzato lato sonda.

```
tcpdump -s 68 -ni moni0 -w -
```

6.2.1.2 tshark

Oltre ad essere un *traffic dumper*, è anche un network traffic analyzer e in questo caso viene usato solo a tale scopo facendo le veci di tcpdump in quanto più avanzato. Sulle basi di quanto visto in 2.2.3.4, viene impostato un

filtro sui valori dei bit del campo frame control fromds e tods così da tenere in considerazione solo il traffico dal BSS al DS o intra-BSS (i.e: generato da STA e diretto al DS, o da STA a STA). L'opzione *-l* ha il solo compito di effettuare il flush dello standard output dopo aver stampato le informazioni relative ad ogni pacchetto. Le opzioni *-T* e *-e* vengono usate congiuntamente per selezionare le informazioni da stampare per ogni pacchetto. In questo caso, timestamp in cui il pacchetto è stato ricevuto, mac address della STA che ha generato il pacchetto e potenza con cui il pacchetto è stato ricevuto.

```
tshark -r - -n -l -T fields \
        -e frame.time_epoch -e wlan.sa -e radiotap.dbm_antsignal \
        -Y "(wlan.fc.fromds == 0  && wlan.fc.tods == 1) || ((wlan.fc.fromds == 0
&& wlan.fc.tods == 0) && wlan.ta != wlan.bssid)"

...
1423334271.144762000      a6:c0:e1:a5:76:f5      -87
1423334271.189329000      cc:fa:00:b5:46:88      -46
1423334271.671614000      a6:c0:e1:a5:76:f5      -87
...
```

6.2.1.3 livestats

E' una componente che legge l'output di **tshark** e mantiene opportunamente delle strutture per il calcolo delle tuple necessarie ai collector per la classificazione dei device. Le informazioni sui device rilevati vengono memorizzate in una struttura come quella in figura 6.4. Ad ogni inserimento, o al più ogni secondo qualora non ci siano dati in input, la lista dei device per cui sono presenti dati, viene scansionata linearmente. I controlli eseguiti durante la scansione per ogni device sono nell'ordine:

1. Verifica numero di pacchetti raccolti: se sono almeno cinque, vengono calcolate media, mediana, terzile, moda e varianza, stampati i valori su standard output e rimossa la coda del device dalla struttura;
   formato output:

```
sentinel mac,device mac,mean,median,tertile,mode,variance
```

2. Se non sono stati raccolti 5 pacchetti, viene verificato che la distanza temporale dal primo all'ultimo pacchetto non sia superiore ad un tempo t_1 fissato e parametrico. Se tale distanza è superiore a t_1, i pacchetti vengono scartati e la coda del device ripulita;

3. Arrivati a questo punto, viene verificato che il tempo trascorso dall'ultima cattura al tempo in cui sono effettuati i controlli non sia superiore ad un tempo t_2 fissato e parametrico. Se tale distanza è superiore a t_2, i pacchetti vengono scartati e la coda del device ripulita;

La verifica al punto 2 impone che non passi più di un tempo $t_1$fra la cattura del primo pacchetto e dell'ultimo. In questo modo, si cerca di aumentare le probabilità che tutti i pacchetti catturati in una sessione non siano stati raccolti in un momento di transizione da una classe di vicinanza ad un altra.

La verifica al punto 3 impone invece che non passi più di un tempo t_2 fra la cattura dell'ultimo pacchetto da un determinato device e l'inserimento del successivo. Questo fa sì che pacchetti processati in ritardo a causa delle latenze di processazione (tempo di sosta nei buffer di sistema + tempo per il dissecting necessario a tcpdump/tshark) non vengano considerati in quanto ormai troppo vecchi e non più di interesse per una visualizzazione realtime. Inoltre, evita il consumo eccessivo di memoria in quanto la struttura che mantiene i dati per device viene regolarmente ripulita dei dati non più interessanti.

Le verifiche al punto 3 sono rilassabili se non si è interessati alla sola visualizzazione realtime ma ad una vista statistica sul lungo periodo.

6.2.1.4 Remote Sender

L'ultimo filtro sulla sonda si occupa di inoltrare lo standard output di livestats verso il server collector. Per farlo viene utilizzato l'utility syslog *logger*. L'output di ogni sonda viene così inviato al server tramite il comando seguente:

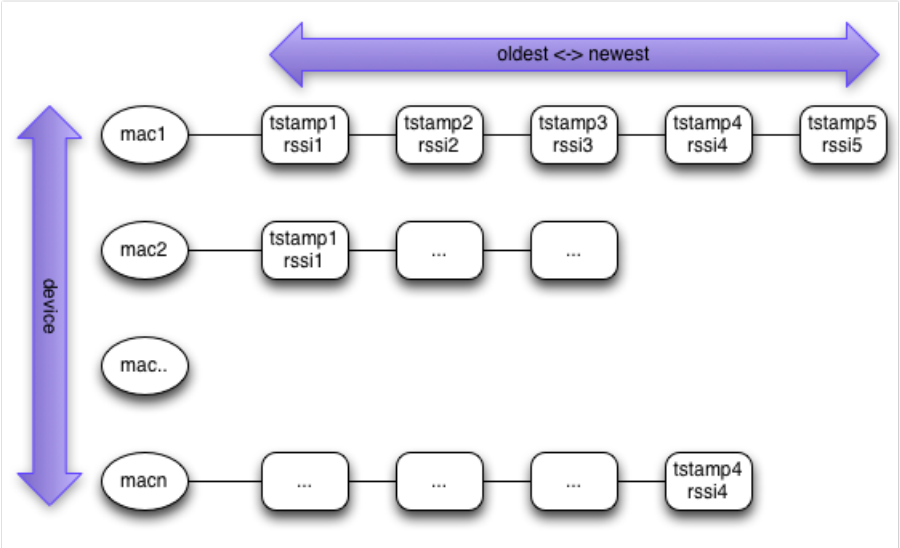


Figura 6.4: livestats - struct

```
logger -n $collector -p local4.info --tag "wigl"
```

Sostanzialmente, il trasporto delle informazioni avviene su UDP e non è previsto ACK per cui l'architettura utilizza un protocollo client-server, per astrarre l'architettura pipe-and-filter anche nel passaggio fra sonde e collector.

6.2.2 Architettura Collector

I server collector sono i server che classificano i device visti dalle sonde grazie alle tuple inviate da quest'ultime tramite protocollo syslog. Fra i collector ne è presente uno che svolge anche funzione di provisioner sia per le sonde che per i restanti collector. Tale server è chiamato provisioner, i restanti vengono in seguito chiamati collector. Il provisioner esporta, esattamente come visto per le sonde, le configurazioni per gli altri collector.

6.2.2.1 Boot e configurazione

Al boot ogni server recupera, attraverso snmp, la lista dei dataset che deve precaricare dal provisioner, i dataset stessi e la lista delle code cui assegnare

Algoritmo 6.2 Provisioning collector via SNMP

```
wiglservera:~# snmpwalk -v2c -c public wiglprovisioner \
        'NET-SNMP-EXTEND-MIB::nsExtendOutLine."getwiglcollectorconf"'

NET-SNMP-EXTEND-MIB::nsExtendOutLine."getwiglcollectorconf".1
        = STRING: 192.168.20.2;1#b8:27:eb:a9:77:eb
NET-SNMP-EXTEND-MIB::nsExtendOutLine."getwiglcollectorconf".2
        = STRING: 192.168.20.2;2#b8:27:eb:b8:64:fa
```

ciascun dataset.

Come è possibile vedere nel listato 6.2, le configurazioni sono rappresentate da stringhe nel formato:

```
IPv4 server;queue id#sentinel mac 1,sentinel mac 2,... .
```

Una volta ottenute le configurazioni, via HTTP dal server provisioner vengono prima scaricati i training set alla URL *"http://wiglprovisioner/ts/*<mac sentinel x>", ed infine fatti partire i processi per la classificazione. Ad ogni processo è indicato parametricamente quali dataset caricare e da quale coda leggere.

6.2.2.2 Balancing

Il meccanismo di balancing è stato introdotto in quanto l'algoritmo KNN, introdotto precedentemente e usato per la classificazione, precalcola le distanza fra tutti i punti al momento del caricamento del training set. Nel meccanismo di configurazione appena descritto è chiaro che, su ogni server girano più processi classificatori e che ciascuno legge da una coda. Resta da chiarire come vengano implementate e riempite le code.

Le configurazioni, lette precedentemente, vengono utilizzate sia dai processi classificatori che dal processo di bilanciamento. Il processo di bilanciamento viene richiamato da *syslog-ng* al momento della ricezione di una entry generata da una sentinella tramite *livestats* sulla facility syslog *local4* e livello di priorità *info*. E' possibile vedere le configurazioni necessarie nel listato 6.3.

Algoritmo 6.3 Configurazione Syslog-ng

```
...
source s_net {
                udp();
};
...
destination d_wigl { program("/opt/wigl-balancer.py"); };
...
filter f_wigl { facility(local4) and level(info); };
...
log {source(s_net); filter(f_wigl); destination(d_wigl); };
...
```

Il processo balancer, ogni volta che viene richiamato da syslog, verifica la tupla in input e la inserisce sulla coda corretta incrociando il campo "sentinel mac" con le configurazioni ottenute dal provisioner. Una volta accodata la entry, l'applicativo esce.

Ad esempio, tutte le entry della sonda identificata dal mac *b8:27:eb:a9:77:eb*, vengono inviate al collector *wiglservera*, dove a sua volta sia il balancer che il processo classificatore sono a conoscenza del fatto che la coda designata per tale sonda è la coda 1.

Le code sono implementate tramite *Redis* ed una classe Python che ne "wrappa" i metodi. *Redis* è un *key-value cache and store db* rilasciato con licenza BSD. Viene spesso definito anche come un "*Data Structure Server*" considerato che ogni chiave può contenere valori organizzati in stringhe, tabelle hash, insiemi, liste, ... Una delle sue peculiarità è che l'intero database è mantenuto in memoria principale garantendo quindi una latenza di accesso ai dati molto bassa.

Nel caso del prototipo oggetto di studio, viene utilizzato un solo database per tutte le code (namespace redis: *wiglqueue*), dove ogni coda indicata nelle

Algoritmo 6.4 Pseudo-codice processo classificatore

```
while true
        tuple = queue.pop(qid)
        sentinel = tuple[sentinelmac]

        outclass = classifiers[sentinel].classify([tuple[mean],tuple[var]])

        store(tuple[devicemac],outclass)
```

configurazioni è una chiave del db con associata una lista. Le operazioni fondamentali, che vengono wrappate dalla classe python usata da balancer e classifier, sono la rpush (inserimento in coda, richiamata dal balancer) e la blpop (lettura e rimozione in testa, bloccante se la coda è vuota, richiamata in loop dai classificatori). Ciascuna di queste operazioni da documentazione Redis ha complessità *O(1)*.

6.2.2.3 Processi di classificazione

Ogni processo di classificazione mantiene una serie di oggetti *KNeighborsClassifier* (uno per sonda assegnatagli), legge dalla coda a lui assegnata le entry aggiunte dal processo balancer ed, in base al campo "sentinel mac", decide quale training-set usare per la classificazione. Il workflow del processo può essere riassunto nello pseudocodice in 6.4.

Le chiamate a *pop* sono bloccanti per cui se la coda è vuota il flusso si arresta in attesa che il balancer inserisca dati sulla coda *qid*. Il processo di stoccaggio dei dati in output e visualizzazione degli stessi non è oggetto di studio di questa tesi.

Considerati i risultati in 5.2.3.2, come parametri di KNN vengono adottati:

- *k=5;*
- Pesi votazioni proporzionali alla distanza.

6.3 Training, Test e Risultati

6.3.1 Training

La fase di training viene effettuata per ogni sonda in quanto il training set deve tenere conto dello shadowing e degli altri disturbi introdotti dall'ambiente in cui queste vengono posizionate.

Per il training sono stati sviluppate due applicazioni in simbiosi: la prima è una app Android che manda in broadcast ogni *100ms* un datagramma UDP su porta 2562 contenente un timestamp ed un numero identificativo della classe di vicinanza. La seconda applicazione è uno script Python che effettua il dissecting dei pacchetti UDP dal device di test e degli header radiotap a questi associati.

I pacchetti generati in aree controllate e delimitate, di cui quindi si conosce la classe di vicinanza, vengono catturati dall'applicazione python che ne effettua il dissecting; infine l'output è fornito in input a livestats che a questo punto fornisce le tuple per il training set.

Un esempio di training set realmente calcolato per una sonda è visibile in figura 6.5. Le curve di cross-validation sono visibili nelle figure 6.6 e 6.7 .

Nella figura 6.5, è possibile osservare anche i *decision boundary* e si può quindi notare il motivo per cui è stata inserita la classe *far* fra la classi *near* e *farther*. Tale classe, infatti, essendo inserita fra le due citate, emula il comportamento che si avrebbe con delle soglie di isteresi limitando di fatto il *flapping* nella classificazione fra oggetti *farther* e oggetti *near*. La classe *near* è rappresentate dai punti e dalla zona blu. Alla classe *far* è associato il verde mentre alla *farther* le due sfumaure di rosso.

Dai grafici delle curve di apprendimento (fig.: 6.6 e 6.7), si nota come, nel caso pratico, la differenza prestazionale fra pesi uniformi e proporzionali alla distanza sia in realtà molto meno netta rispetto al modello teorico. Rimangono del tutto invariate, invece, le considerazioni fatte sulla variabile k dell'algoritmo che può tranquillamente essere scelta pari a 5.

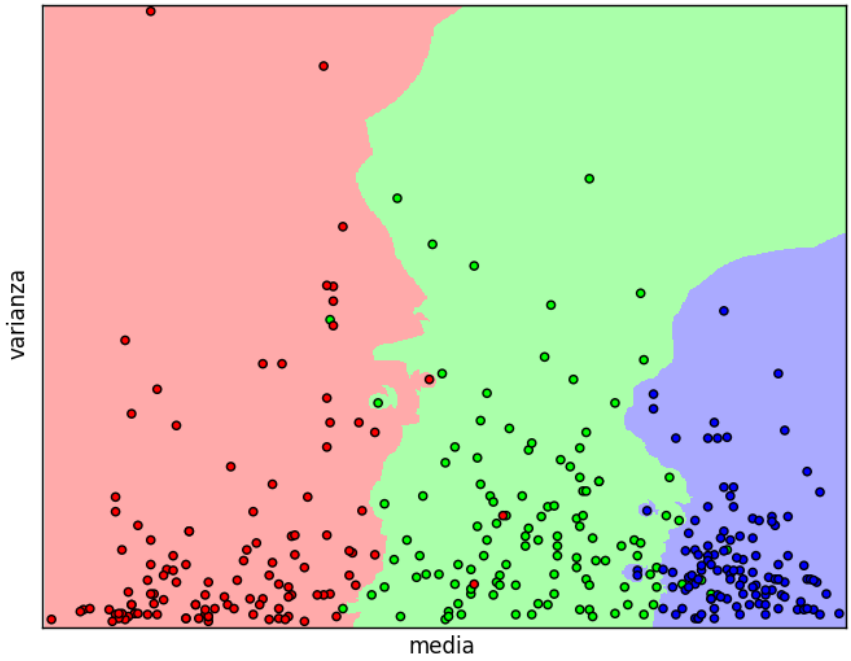
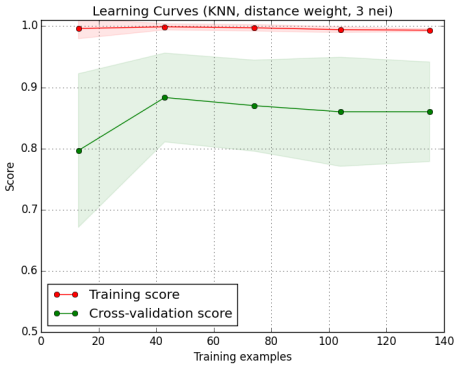


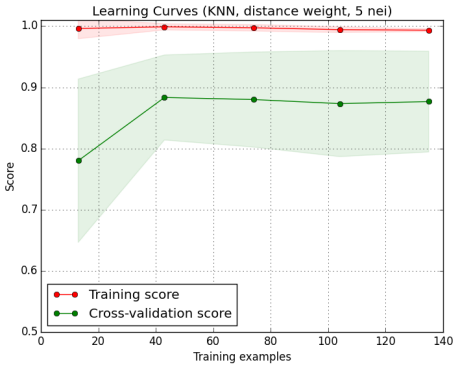
Figura 6.5: Training-Set: decision boundary e training point

6.3.2 Parametri e contesto test

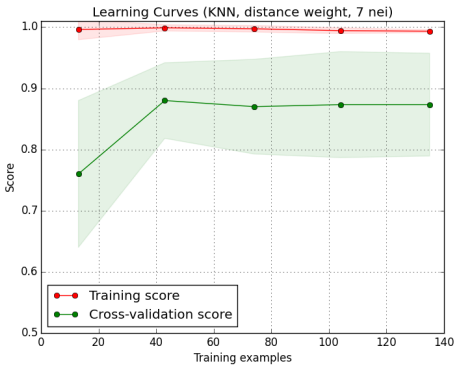
1. KNN:
   (a) *k=5;*
   (b) distance weigth.
2. Livestats:
   (a) 5 elementi per campione;
   (b) $t_1 = 3s$;
   (c) $t_2 = 3 * t_1 = 9s$.
3. Configurazioni run:



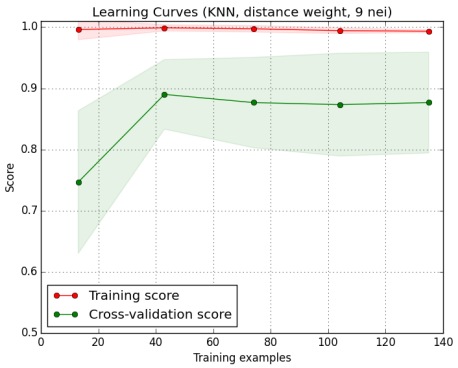
(a) K=3



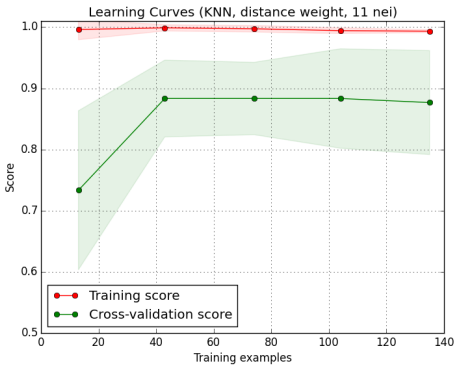
(b) K=5



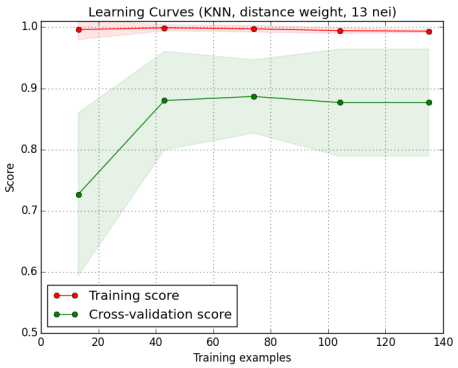
(c) K=7



(d) K=9

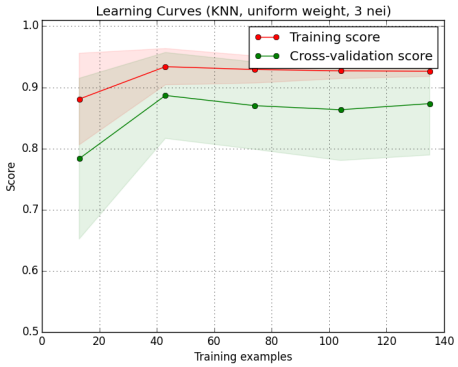


(e) K=11

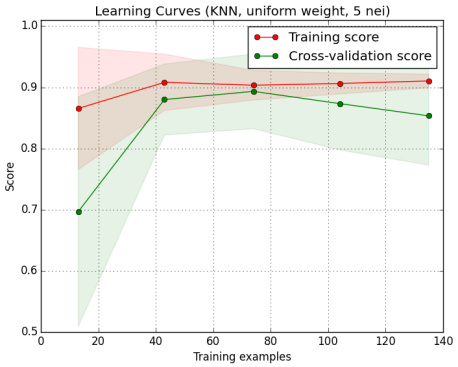


(f) K=13

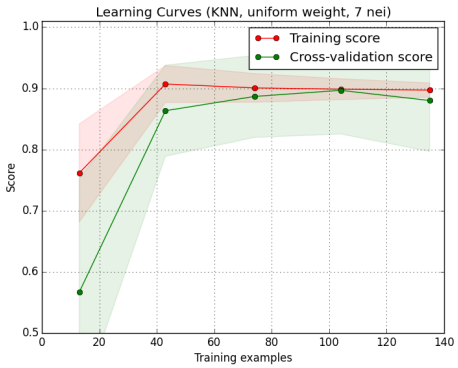
Figura 6.6: Cross-Validazione TS reale - Pesi proporzionali alla distanza



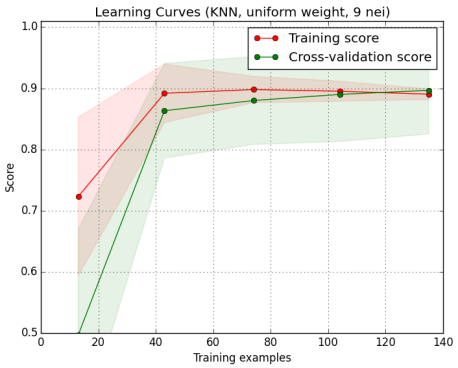
(a) K=3



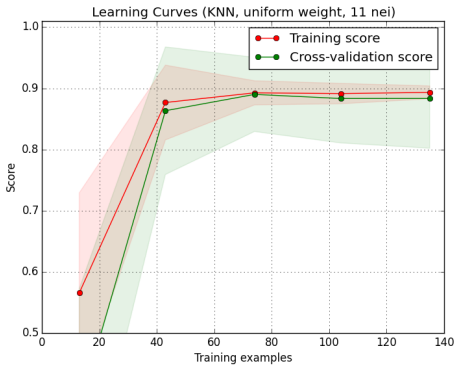
(b) K=5



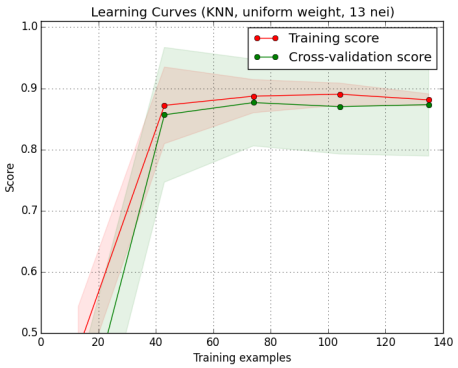
(c) K=7



(d) K=9



(e) K=11



(f) K=13

Figura 6.7: Cross-Validazione TS reale - Pesi uniformi

(a) 3 Scenari:

i. Device da classificare in un punto fisso dell'area di vicinanza;

ii. Device da classificare in movimento all'interno dell'area di vicinanza ed in uso attivo (i.e: tenuto in mano dal possessore);

iii. Device da classificare in movimento all'interno dell'area di vicinanza e non in uso (i.e: tenuto in tasca o in borsa) (n.b: chiamato in seguito "uso passivo").

(b) test per scenario: 10 test per ogni classe di vicinanza;

(c) pacchetti per ogni test: 300 pacchetti UDP in broadcast per ogni classe di vicinanza.

Con i parametri elencati in 1. e 2., sono stati approntati dei test per stabilire quanto l'implementazione reale degli algoritmi di classificazione differisca dall'idea teorica vista in 5.2.

Per lo studio della classificazione è stata utilizzata la stessa applicazione Android usata per la costruzione del training-set. Si ricorda che tale applicazione invia in broadcast pacchetti udp contenenti 2 campi: *(i)* timestamp e *(ii)* id della classe di vicinanza. Per la classificazione è stata invece adottata una versione modificata di livestats che, contrariamente alla versione standard, accetta una tupla in input comprensiva di identificativo della classe di vicinanza in cui è stato effettivamente generato il pacchetto e accoda alla tupla in output lo stesso valore. Le statistiche per burst di pacchetti non tutti segnalati come appartenenti alla stessa classe di vicinanza non vengono prodotte.

A questo punto, è stato possibile adottare lo stesso tool di classificazione utilizzato nell'approccio simulativo che viene però addestrato con il training set reale.

I risultati dei test vengono esposti in 6.3.3.

6.3.3 Risultati

Vengono ora presentati i risultati dei test introdotti nella sezione precedente. Le metriche di valutazione per la classificazione sono le stesse adottate per le simulazioni e di seguito riportate:

TPR (*True Positive Rate* $TPR = TP/(TP + FN)$) Indica la probabilità con cui il classificatore è in grado di assegnare una classe ad un oggetto effettivamente facente parte di essa. Tale parametro è indicato anche come *sensibilità*/*sensitività*;

TNR (*True Negative Rate* $TNR = TN/(FP + TN)$) Indica la probabilità con cui il classificatore è in grado di escludere l'appartenenza ad una classe per un oggetto che effettivamente non vi appartiene. Tale parametro è anche noto come *specificità*;

PPV (*Positive Predictive Value* $PPV = TP/(TP + FP)$) Indica la probabilità con cui un oggetto classificato come appartenente ad una data classe lo sia effettivamente. Tale parametro è anche indicato come *predittività positiva* o *valore predittivo positivo*;

NPV (*Negative Predictive Value* $NPV = TN/(TN+FN)$) Indica la probabilità che un device non classificato come appartenente ad una determinata classe non lo sia effettivamente. Tale parametro è anche indicato come *predittività negativa* o *valore predittivo negativo*;

FPR (*False Positive Rate* $FPR = FP/(FP+TN) = 1-TNR$) Probabilità di commettere errori di primo tipo;

FNR (False Negative Rate $FNR = FN/(FN + TP) = 1 - TPR$) Probabilità di commettere errori di secondo tipo;

Accuratezza ($ACC = TP + TN/(TP + TN + FP + FN)$) Rappresenta la capacità dell'algoritmo di includere od escludere correttamente un punto come appartenente alla classe di vicinanza oggetto di studio.

In figura 6.8, è possibile osservare i risultati per il primo scenario, cioè quello in cui vengono classificati dei dispositivi fermi in un punto di ogni area di classificazione. In questo caso è possibile osservare come, sopratutto per la classe *near,* i parametri statistici siano ottimali. Si hanno alti livelli di accuratezza e *positive predictive value* (predittività positiva) oltre che di *true positive rate* (sensibilità), *true negative rate* (specificità) e *negative predictive value* (predittività negativa). Per le altre classi le valutazioni, seppur leggermente inferiori, sono in linea.

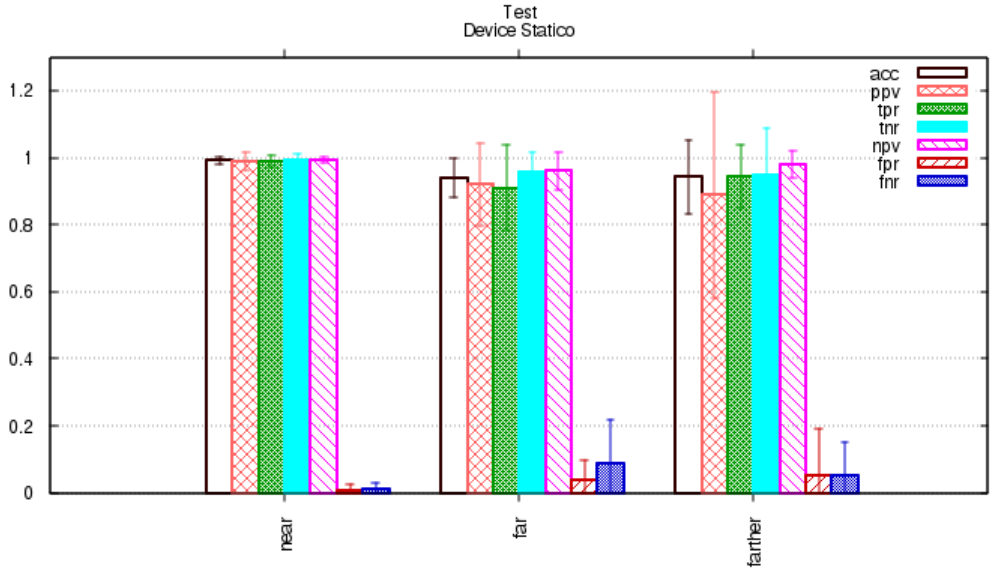


Figura 6.8: Scenario 1 - Device statico

Nel caso del secondo scenario, con dati osservabili in fig.: 6.9, si considerano dispositivi in movimento ed utilizzati attivamente dai possessori. In questo caso, il livello di accuratezza scende del *15%* circa. Questo degrado è in buona parte spiegato da una diminuzione più netta, rispetto agli altri quantificatori, del *tpr*, in quanto aumenta il numero di elementi non classificati come *near* ma realmente tali, cioè dei falsi negativi. Si può dedurre che il classificatore genera un numero ridotto di falsi positivi. Le misure relative al *tnr*, indice della frequenza con cui vengono attribuiti falsi positivi, del *ppv*

e del *npv,* infatti subiscono attenuazioni minori. Le variabili per le classi *far* e *farther*, subiscono attenuazioni più uniformi.

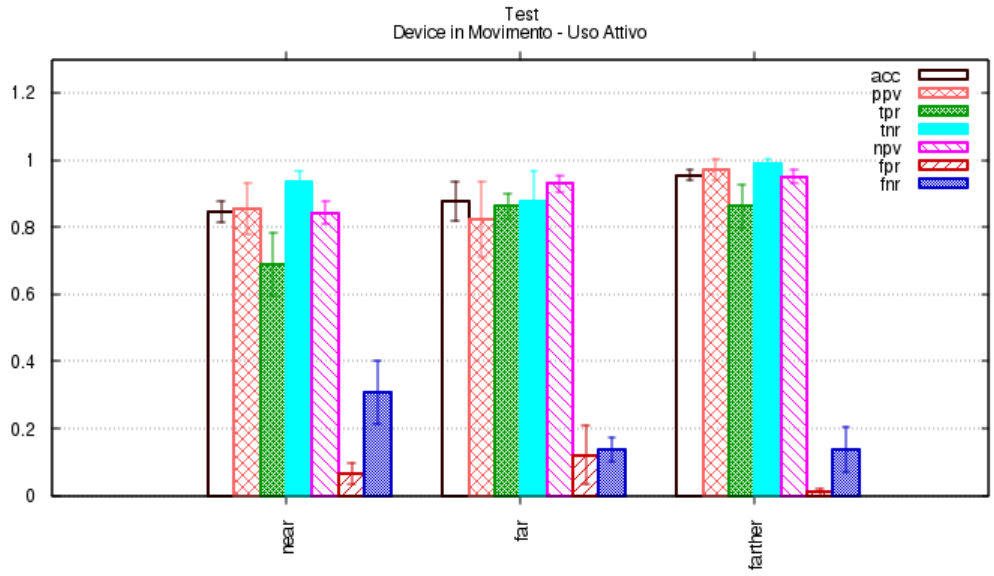


Figura 6.9: Scenario 2 - Utente in movimento - Uso attivo del device

Analizzando il terzo ed ultimo scenario (grafico 6.10), in cui vengono valutate le classificazioni di device in movimento ma non usati attivamente (i.e: sono locati in tasche, borse, zaini, etc. etc.), si nota un calo repentino dell'accuratezza nell'ordine di un ulteriore *20-25%*. Questo calo è dovuto in ottima parte ad un crollo del *tpr* per cui ad un aumento dei falsi negativi. La parallela diminuzione dei valori di *ppv,* per le classi *far* e *farther* denota come aumentino i casi di device classificati come lontani pur essendo in realtà vicini. **Gli alti valori di *ppv* e di *tnr* per la classe *near* mostrano invece come non aumentino i falsi positivi.** Anche in questo caso, quindi, si può notare come il classificatore, pur riducendo la propria accuratezza, non segnali device come vicini quando in realtà essi sono lontani dalle sonde. La diminuzione delle performance è in buona parte dovuta alle attenuazioni meno deterministiche introdotte dalla vicinanza dei dispositivi con il corpo. Questo scenario può essere molto frequente, si considerino ad esempio i punti di transito in una *hall* di un aeroporto o di una stazione della metropolitana.

Il fatto che probabilisticamente siano presenti molti fasi negativi e pochissimi falsi positivi, lascia aperta la possibilità di investigare su possibili tecniche di sovrastima del numero di device effettivamente vicini ad una sonda in base al numero dei soli individuati.

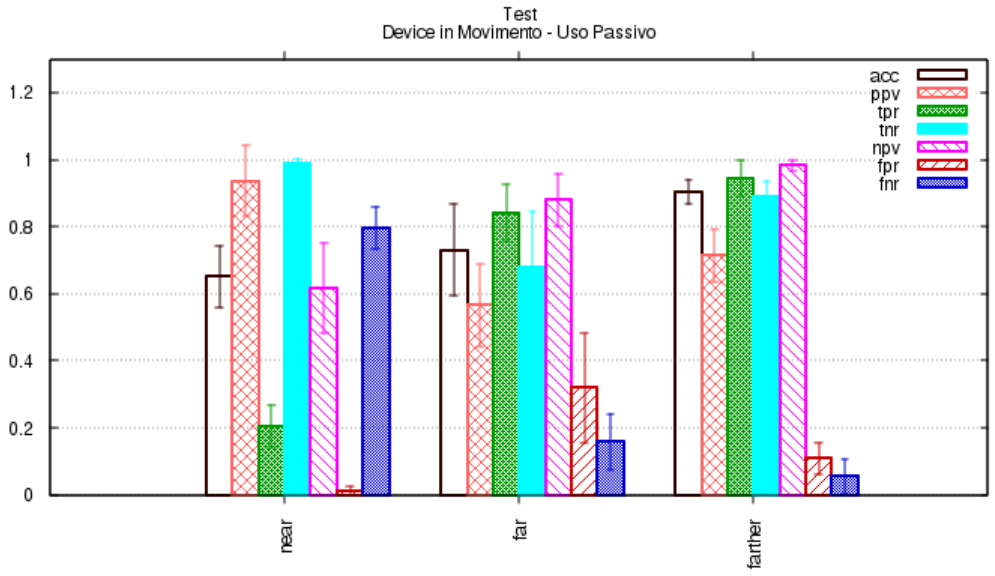


Figura 6.10: Scenario 3 - Utente in movimento - Device riposto

Confrontando questi dati con quelli ottenuti dalle simulazioni precedenti, vedi capitolo 5.2.3, si nota che la sensitività/sensibilità diminuisce dapprima con l'introduzione del movimento ed in seguito con mancanza di visibilità ottica con le sonde, cioè con l'aggiunta di un ulteriore elemento causa di shadowing. Questo comportamento è in linea con quanto visto nelle simulazioni. In linea, anche se leggermente migliore, è anche l'andamento del *true negative rate* che permane sempre sopra il 90%. Contrariamente, il *positive predictive value* nel caso reale viaggia, contrariamente alle simulazioni dove arrivava ad attestarsi intorno al 60%, su valori molto alti (80%) ed è indice del fatto che gli elementi classificati come *near*, lo sono con buone probabilità realmente. Infine, il *negative predictive value* è peggiore rispetto alle simulazioni nel solo scenario 3 in quanto aumentano in maniera più decisa i falsi negativi.

Capitolo 7

Conclusioni e Sviluppi Futuri

7.1 Conclusioni

Nel corso della tesi sono state analizzate le motivazioni che hanno indirizzato le scelte architetturali relative ad un sistema per la localizzazione dei dispositivi mobili. In questo caso, il concetto di localizzazione è diverso da quello che oggi si è automaticamente portati a pensare. Infatti, nella soluzione proposta, i dati resi disponibili dal prototipo non vengono utilizzati dagli utilizzatori finali, come potrebbe essere in una applicazione *location aware*, ma ad un'ente di sorveglianza in grado così di sapere quante e quali persone si stanno muovendo in un determinato luogo. In 1.1, si è visto come il crescente trend relativo al possesso e all'uso di smartphone sia stato determinante nello scegliere di realizzare un sistema di geolocalizzazione basato sulla tecnologia wifi. In molti paesi sviluppati si è avuto, dal 2011 al 2013, un incremento annuo del 10-15% dei detentori di smartphone e mediamente nel 2013, in questi stessi paesi, ci si attesta intorno al 50% della popolazione. A sua volta, circa un quarto dei detentori di device mobili fa uso costante del WiFi *out-of-home* quando lo ha a disposizione. Sono poi state effettuate alcune considerazioni su come una soluzione, basata su questa tecnologia, consenta non solo la localizzazione dei device, ma fornisca anche, in certe condizioni, dati importanti per poter risalire all'identità dei possessori. Questo aspetto

è importante in quei contesti pubblici quali possono essere ad esempio gli aeroporti, stazioni e più in generale luoghi pubblici, dove queste soluzioni possono essere adottate per incrementare il livello di sicurezza.

A tali considerazioni, ha fatto seguito l'introduzione di alcune tecnologie utilizzate, appunto 802.11 (in 2.2) per quanto concerne gli aspetti utili allo scopo della tesi, e SNMP (in 2.1) per il provisioning delle sonde.

In particolare, per quanto riguarda gli aspetti di 802.11, ci si è concentrati sul formato dei dati contenuti nella trama MAC. In 2.2.3.4 , è evidenziato come sia, teoricamente, possibile distinguere il traffico generato da dispositivi mobili, mentre in 6.2.1.2 è indicato il filtro, utilizzato dalle sonde per l'estrazione reale di tale traffico. Oltre agli aspetti tecnici *low-level* del wifi, è stata presentata anche l'architettura per la gestione di una infrastruttura wireless complessa e orientata al business. Tale architettura, è basata su wireless controller e access point "senza logica", configurati per stabilire una connessione con il wlc, scaricare le configurazioni da adottare e mantenere aperta la connessione per consentire il management degli ap stessi. Su questa infrastruttura, sono costruite la maggior parte delle soluzioni commerciali viste in 4.5, in quanto mette a disposizione un punto centrale dove reperire una grande quantità di informazioni.

In 3, sono state presentate alcune analisi sui pattern con cui i dispostitivi mobili generano traffico e su come modulano la potenza in trasmissione in base alle politiche di risparmio energetico. Si è potuto osservare la frequenza con cui vengono inviati pacchetti *probe-request*, utili per l'identificazione dei dispositivi, e che questi vengono inviati anche quando l'interfaccia WiFi di un dispositivo Android viene "spenta", ma vengono lasciati attivi i servizi di geolocalizzazione Google. Questa scelta operata da Google, fa sì che il numero di device rintracciabili aumenti.

Nel paragrafo 4, sono state presentate alcune caratteristiche dei sistemi di geolocalizzazione ed una loro classificazione. Riassumendo, si distinguono in prima battuta sistemi in cui la localizzazione è utile ad un sistema remoto e non al geolocalizzato, detti sistemi di remote-positioning, mentre *vice-versa*

si è alle prese con un sistema di self-positioning. E' stato scelto di dirigersi su un sistema di tipo remote-positioning perché uno dei vincoli posti era quello legato alla totale assenza di logica applicativa lato device ed in questo caso è stata per cui l'unica strada percorribile. In secondo luogo, la distinzione viene effettuata sulle tecniche adottate per l'individuazione dei device e sono state analizzate tre tipologie distinte: *(i)* tecniche basate su proprietà geometriche come triangolazione e trilaterazione, *(ii)* tecniche basate sull'analisi degli ambienti tramite *fingerprinting* (i.e: addestramento) quali quelle che adottano modelli probabilistici o basati su machine learning e *(iii)* algoritmi basati su test di prossimità. Le tecniche elencate in *(i)* e *(ii)* presentano delle problematiche relative alla necessità, talvolta, di una forte sincronizzazione fra le sonde installate, alla necessità di hardware particolare, o infine di logica applicativa lato device mobile. Le soluzioni proposte in *(iii)* possono invece essere più imprecise rispetto alle precedenti, ma godono del fatto che necessitano di vincoli meno stringenti rispetto a sincronizzazione e hardware specifico.

Nel caso specifico del prototipo presentato, la soluzione scelta è stata quella del test di prossimità, rafforzato con tecniche basate su machine learning. L'approccio basato su machine learning è stato quindi adottato per distinguere, sulla base di un dataset d'addestramento, tutti quei device entro un raggio di azione di 5 metri dalle sonde. In sostanza, tramite il fingerprinting a 3 classi, proposto in 5.2.1, si vanno a calcolare, tenendo conto dello shadowing in una determinata area, le soglie di isteresi con cui considerare un device come *prossimo* o non alla sonda.

Prima di procedere con una implementazione reale, in 5.1 sono state approntate delle simulazioni sulla base di alcuni modelli di propagazione che tengano conto, almeno in parte, dello shadowing. Nelle simulazioni, si è modellato il rumore dovuto allo shadowing, tramite una variabile random con distribuzione gaussiana di varianza direttamente proporzionale alla distanza. In queste simulazioni, si è potuto osservare come l'algoritmo di classificazione KNN, sia in grado di distinguere le caratteristiche di prossimità con

un buon grado di accuratezza generale (i.e: sempre superiore al *75%* per la classe *near*). Allo stesso modo, è stato evidenziato come l'aumentare della varianza, e quindi dello shadowing, introduca ,in quantità simili, errori di classificazione di primo tipo (i.e: falsi positivi appartenenti alla classe *near*) e di secondo tipo (i.e: elementi *near* non individuati).

Una volta ottenuti questi risultati, si è proceduto con la realizzazione del prototipo applicativo. In sostanza, sono stati presentati nell'ordine le parti fisiche utilizzate (sezione 6.1), le caratteristiche delle componenti software (sezione 6.2) e infine, l'architettura per il provisioning delle configurazioni e per il bilanciamento del carico lato server decisionale.

E' stata sviluppata una applicazione simile a quella adottata in [2] per generare il traffico necessario alla costruzione del training set e, su questo, sono state ripetute le analisi relative al *tuning* dell'algoritmo di classificazione scelto (i.e: *KNN*), fatte anche per le simulazioni. I risultati sono risultati essere in linea: KNN viene utilizzato con pesi proporzionali alla distanza dei vicini e con K (numero di punti elettori) pari a *5*. Tuttavia, contrariamente alle simulazioni, l'algoritmo con pesi uniformi ha fornito risultati equiparabili.

Infine, sono state riprodotte le statistiche analizzate durante le simulazioni. In questo caso, le analisi sono state incentrate soprattutto sullo studio della classificazione dei dispositivi in base all'uso fattone dal possessore distinguendo tre casi: *(i)* uso statico (e.g: navigazione web in una sala d'attesa), *(ii)* uso attivo del dispositivo e soggetto in movimento (e.g: consultazione di audio guide tramite apposite *app* in percorsi museali) e *(iii)* dispositivo in stand-by e posto a riposo all'interno di tasche, zaini, borse, ... **Si è potuto osservare come l'accuratezza della classificazione abbia una** prima **diminuzione all'aumentare della dinamicità dell'utente**, tuttavia questa rimane superiore all'*80%*. **Nel caso in cui invece un dispositivo sia riposto ed in *stand-by,* l'accuratezza subisce una ulteriore e netta diminuzione.** Dalle statistiche presentate in 6.3.3, si nota che questa netta diminuzione dell'accuratezza è dovuta principalmente ad una diminuzione del *true positive rate,* cioè ad un repentino aumento dei device *near*

non individuati come tali. Nonostante questa diminuzione dell'accuratezza, è però lodabile il fatto che, in realtà, **non si ha una diminuzione del *positive predictive value* e del *true negative rate* per cui permane molto bassa l'incidenza di elementi erroneamente categorizzati come prossimi alle sonde. Questo comportamento lascia aperta la possibilità di dedurre la quantità reale di elementi prossimi ad una sonda sulla base di quelli rilevati** in quanto si ha una stima della quantità di elementi non identificati e che il classificatore commette pochi errori di primo tipo.

In conclusione, la tecnica presentata è in grado di rafforzare correttamente una soluzione di geolocalizzazione basata su test di prossimità pur presentando alcune lacune nell'individuazione dei dispositivi risposti ed in possesso di utenti in movimento. Infatti, si è visto che tali device vengono spesso classificati come lontani dalla sonda. L'aspetto comunque positivo è legato al fatto che raramente dispositivi realmente lontani vengono classificati come prossimi. L'elaborazione dei dati ottenuti dalle singole sonde per dedurre il numero di persone, mettendo in relazione i dati di più sonde, non era obiettivo di questa tesi.

7.2 Sviluppi Futuri

La tesi non ha trattato alcuni elementi circa l'elaborazione dei dati ottenuti dalle sonde. In primo luogo, sarebbe quindi opportuno definire algoritmi in grado di elaborare quali siano i percorsi più gettonati, ed i flussi più frequenti nonché di evidenziare, sulla base di questi dati, eventuali anomalie ad esempio dovute a problematiche di sicurezza in atto.

Altresì, buona parte del prototipo applicativo, soprattutto per quanto riguarda le sonde, è stato realizzato sfruttando componenti software già esistenti collegati tra loro tramite un approccio *pipe and filter*. Per ottimizzare le prestazioni sarebbe auspicabile l'implementazione di software dedicati, in grado di gestire solo ed esclusivamente la logica applicativa necessaria. Sem-

pre a livello architetturale, ad ora la soluzione non prevede che una sonda a *run-time* possa cambiare dinamicamente il collector a cui inviare i dati. Per aumentare la scalabilità, potrebbe essere utile pensare a meccanismi di *live-migration* delle sonde per cui all'aumentare del carico del server a cui sono associate, queste possano, su sollecitazione del collector stesso, cambiare server a cui inviare le tuple da classificare. Fra le metriche di stima del carico dei server, potrebbe essere possibile analizzare il traffico *inbound* sulle schede di rete che è indice del numero di tuple classificate. Tale valore, è tipicamente esportato dai sistemi operativi via SNMP e quindi potrebbe essere sfruttato per implementare il meccanismo di *migration* in modo centralizzato.

Per quanto riguarda l'algoritmo basato su test di prossimità, andrebbe rafforzato ulteriormente il riconoscimento dei dispositivi appartenenti a soggetti in movimento. Inoltre, per arrivare ad una precisione maggiore e ad una approssimazione della posizione di un device anche quando non è in prossimità di una sonda, sempre rispettando i vincoli che impongono che non sia presente logica applicativa lato dispositivo mobile, si potrebbero adattare alcune soluzioni, come ad esempio RADAR [2] o HORUS [37, 38]. Questo potrebbe essere realizzabile introducendo meccanismi di sincronizzazione fra le sonde attui ad approssimare quali pacchetti siano stati rilevati contemporaneamente su più nodi rilevatori. Tale modifica, consentirebbe di realizzare un sistema di remote-positioning adattando sistemi di self-positioning senza necessità applicazioni lato device.

Bibliografia

[1] Norman Abramson. Development of the alohanet. *Information Theory, IEEE Transactions on*, 31(2):119–123, 1985.

[2] Paramvir Bahl and Venkata N Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784. Ieee, 2000.

[3] Roberto Battiti, Nhat Thang Le, and Alessandro Villani. Location-aware computing: a neural network model for determining location in wireless lans. Technical report, DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY, University of Trento, 2002.

[4] Sujittra Boonsriwai and Anya Apavatjrut. Indoor wifi localization on mobile devices. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2013 10th International Conference on*, pages 1–5. IEEE, 2013.

[5] Jeffrey Case, Mark Fedor, M Schoffstall, and James Davin. Rfc 1157: Simple network management protocol (snmp). *IETF*, RFC 1157:0 to end, April, 1990.

[6] Jeffrey Case, Keith McCloghrie, M Rose, and Steven Waldbusser. Introduction to version 2 of the internet-standard network management framework. *IETF*, IETF:0, 1993.

[7] Adrian Chadd. The firmware for qca ar7010/ar9271 802.11n usb nics. URL: https://github.com/qca/open-ath9k-htc-firmware.

[8] LAN/MAN Standards Committee. Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. *IEEE Standard for Information technology*, Telecommunications and information exchange between systems Local and metropolitan area networks — Specific requirements:0, 2012.

[9] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[10] Christopher Drane, Malcolm Macnaughtan, and Craig Scott. Positioning gsm telephones. *Communications Magazine, IEEE*, 36(4):46–54, 1998.

[11] H. T. Friis. Proceedings of the ieee, 1946.

[12] Matthew Gast. *802.11 wireless networks: the definitive guide*. OReilly Media, Inc., 2005.

[13] David Gualda, Jesus Urena, Juan C Garcia, Enrique Garcia, and Daniel Ruiz. Rssi distance estimation based on genetic programming. In *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*, pages 1–8. IEEE, 2013.

[14] Jeffrey Hightower and Gaetano Borriello. Location systems for ubiquitous computing. *Computer*, 34(8):57–66, 2001.

[15] Jia Huanxiang, Wang Yong, and Tao Xiaoling. Localization algorithm for mobile anchor node based on genetic algorithm in wireless sensor network. In *Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on*, pages 40–44. IEEE, 2010.

[16] International Data Corporation (IDC). Always connected, how smartphones and social keep us engaged, 2013.

[17] IETF. *Lightweight Access Point Protocol*, number RFC5412, 2010.

[18] Cisco Inc. Channel deployment issues for 2.4-ghz 802.11 wlans. URL: http://www.cisco.com/c/en/us/td/docs/wireless /technology/-channel/deployment/guide/Channel.pdf, 2004.

[19] Cisco Inc. Cisco motion: Design and deployment of contex-aware mobility services, 2012.

[20] Google Inc. The google maps geolocation api. URL: https://developers.google.com/maps/documentation/business/geolocation.

[21] Google Inc. Our mobile planet. URL: http://think.withgoogle.com/mobileplanet/en, 2013.

[22] Xinrong Li, Kaveh Pahlavan, Matti Latva-aho, and Mika Ylianttila. Comparison of indoor geolocation methods in dsss and ofdm wireless lan systems. In *Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000. 52nd*, volume 6, pages 3015–3020. IEEE, 2000.

[23] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, 2007.

[24] Amitangshu Pal. Localization algorithms in wireless sensor networks: Current approaches and future challenges. *Network Protocols and Algorithms*, 2(1):45–73, 2010.

[25] Charalampos Papamanthou, Franco P Preparata, and Roberto Tamassia. Efficient localization for wireless sensor networks using power measurements sampling. In *Proc. of the 3rd ACM workshop on Wireless security*, page 21. Citeseer, 2004.

[26] Charalampos Papamanthou, Franco P Preparata, and Roberto Tamassia. Algorithms for location estimation based on rssi sampling. In *Al-*

gorithmic Aspects of Wireless Sensor Networks, pages 72–86. Springer, 2008.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[28] Shreharsha Rao. Estimating the zigbee transmission-range ism band-designers of short-range wireless devices in the 900-mhz and 2.4-ghz band need to understand what and how parameters affect the transmission range. *EDN*, 52(11):67–74, 2007.

[29] Nattakorn Saengwongwanich, Chundi XIU, Jingnong WENG, and Pisit Boonsrimuang. Indoor/outdoor switching algorithm based on wi-fi receive signal strength and gps. In *International Conference on Indoor Positioning and Indoor Navigation*, volume 27, page 30th, 2014.

[30] Abdullah E Shaikh, Faisal Majeed, Muhammad Zeeshan, Tahir Rabbani, and Imran Sheikh. Efficient implementation of deterministic 3-d ray tracing model to predict propagation losses in indoor environments. In *Personal, Indoor and Mobile Radio Communications, 2002. The 13th IEEE International Symposium on*, volume 3, pages 1208–1212. IEEE, 2002.

[31] Informa Telecoms and Media. Understanding today's smartphone user: Demystifying data usage trends on cellular and wi-fi networks. White Paper, 2012.

[32] Informa Telecoms and Media. Smartphone use transforming with the rise of 4g and wi-fi. White Paper, 2014.

[33] Andreas Teuber, Bernd Eissfeller, and Thomas Pany. A two-stage fuzzy logic approach for wireless lan indoor positioning. In *Proc. IEEE/ION Position Location Navigat. Symp*, volume 4, pages 730–738, 2006.

[34] Jean Tourrilhes. Wireless extensions for linux. *Wireless Tools for Linux*, 1:0, 1997.

[35] TP-Link. Tl-ant2414a datasheet. URL: http://www.tp-link.it/Resources/document/TL-ANT2414A_V1_Datasheet.zip.

[36] wikidev. Atheros ar9271. URL: https://wikidevi.com/wiki/Atheros_AR9271, 2015.

[37] Moustafa Youssef and Ashok Agrawala. Handling samples correlation in the horus system. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1023–1031. IEEE, 2004.

[38] Moustafa Youssef and Ashok Agrawala. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218. ACM, 2005.

[39] Moustafa A Youssef, Ashok Agrawala, and A Udaya Shankar. Wlan location determination via clustering and probability distributions. In *Pervasive Computing and Communications, 2003.(PerCom 2003). Proceedings of the First IEEE International Conference on*, pages 143–150. IEEE, 2003.

Ringraziamenti

Il primo grande, grandissimo, grazie va alla mia famiglia. Per avermi regalato la possibilità di studiare ciò che mi appassiona e per avermi sempre assecondato in tutti i momenti del mio percorso. Grazie anche e sopratutto per il sostegno in quei momenti in cui le difficoltà non son state legate allo studio. Grazie per avermi spronato nel momento in cui ho deciso di lavorare e per avermi tenuto sempre ben presente che comunque non dovevo mollare l'università. Poche parole, ma un grazie profondo, dal cuore.

Il secondo grande, grandissimo, grazie va alle persone a cui questa tesi è dedicata: mio nonno e Checco. A mio nonno perché non ha mai dubitato che potessi farcela. Per le volte in cui sin da piccolo mi è stato complice e per le volte in cui è stato inflessibile. Per ogni volta che mi ha aiutato a nascondere il pane a tavola, per avermi fatto credere che la seppia è il pesce di montagna, per avermi fatto buchi quadrati nel pecorino perché era finito l'Emmentaler e per quelle volte in cui mi ha mancato lanciandomi le ciabatte. Per il modo radicale, senza troppe vie di mezzo, di prendere decisioni in base a cosa fosse giusto e cosa sbagliato. Per il suo modo di essere burbero ma di cuore. Per la commozione nei suoi occhi, che raramente avevo visto lucidi, quando ho preso la laurea triennale e per quella che spero di regalargli questa volta. A Checco perché era un amico. Perché seppur ad un certo punto nella vita le frequentazioni cambiano, le persone su cui si può contare, gli amici, no. Per avermi fatto conoscere la musica che poi è diventata l'altra mia grande passione, per ogni nota suonata assieme con gli "Incognita", per ogni giorno dei lunghi 11 anni sugli stessi banchi di scuola, per ogni cazzata detta o

fatta... perché ognuna ora è un ricordo che mi porto appresso ed un grazie che gli devo.

Grazie alla morosa che in questi mesi in cui lavoravo e studiavo, studiavo e lavoravo, è sempre stata paziente (anche se paziente lo sono anch'io :D) capendo le mie necessità. Grazie per la presenza di tutti i giorni, perché c'è.

Grazie a mia sorella, la Mongul, perché è una pazza, isterica, pignola, puntigliosa, quadrata, crucca rompiscatole (...e resto sul fine). Perché nonostante tutti questi aspetti del suo carattere non combacino proprio alla lettera con quelli del mio (ho sempre sostenuto che lei è quella metodica, io quello geniale), ci vogliamo bene. Perché oltre ad essere una sorella, è un'amica.

Grazie a Pezza. Perché volta e gira una volta entrati l'uno nella vita dell'altro non ne siam mai usciti. Perché è l'altro, con Checco, a far parte della ristretta cerchia su cui si può sempre fare affidamento.

Grazie a Celestina, per avermi corretto gran parte della tesi senza neanche schifarsi più di tanto per numeri e nerdate. Grazie anche a Roberto, Micol, Veronica, Giampi, Marco e Maily. Grazie a Vittoria, Viola e Pietro per ogni sorriso, primi nipotini dello Zio Tappo :) .

Grazie al Boccia e allo Stecca (lascio al lettore l'analisi dei soprannomi), a Uovo, Bagna, Toshi e JJ, a Davide, Cruz, Valentini, Viscardo, Carlo e Vincenzo, a Gigio e Badiali e a Luigi. Perché è stato bello studiare insieme. Senza la compagnia, il supporto reciproco durante le infinite slavagnate, e una buona dose di boiate, sarebbe stato veramente più difficile. Tornando alla triennale e rimanendo in tema compagni, ringrazio Bigo e Mone perché, insieme a Uovo, mi hanno introdotto al pinguino. Oltre che a Rudi, compagno di mille progetti, Giona, Rambo, Laura e Raffaele.

Grazie a Vittorio, che mi fa da relatore, per la disponibilità e per quanto fatto per me e la morosa. Trovo che oggi sempre più difficile trovare persone che si interessano a qualcuno come ha fatto lui per noi. Grazie.

Grazie a Bob, Capa, Sergio e Michele, per avermi dato l'opportunità di lavorare a questo progetto durante il lavoro in azienda.

Grazie ai colleghi, all'Antonella e Matteo, per come mi hanno accolto quando ho iniziato la mia esperienza ai labs, a Riccio e Rio, perché mi sbotrano, a Giuseppe, con cui ho iniziato insieme l'avventura, al Benno, a Luigi e a Comodi oltre che a Max e Mattia da meno entrati nel gruppo. A Max anche perché dopo anni che non lo facevo mi ha dato di nuovo la possibilità di suonare. Grazie anche a tutti gli altri: Billi e la Fra, Api, Max, Elisa e Frollo, Antonio e il *Mr. big watermelon* Monta. Spero di non aver dimenticato nessuno.

Grazie a quelli che ho scordato, sappiate che non c'è intenzionalità ;) .