



## UNIVERSITÀ DEGLI STUDI DI FERRARA

Corso di Laurea in Ingegneria Elettronica e Informatica

# Progettazione e Realizzazione di un Sistema Software per il Tracciamento Indoor delle Risorse Umane e Materiali

Tesi di Laurea di:  
Michele Vaccari

Relatore:  
Ing. Mauro Tortonesi



# Ringraziamenti

In primo luogo vorrei ringraziare l'Ing. Mauro Tortonesi, per l'aiuto e il sostegno fornитоми durante la stesura del lavoro.

Un ringraziamento speciale va a Giovanni Borghi, a Sabrina Formica e in generale a tutto il team di Evomatic s.r.l., per avermi permesso di svolgere il lavoro di tirocinio in un ambiente sereno e informale. Un ringraziamento di dovere va anche a Lorenzo Visentin, mio tutor aziendale, per avermi seguito amichevolmente e per avermi dato i consigli e i suggerimenti necessari per lo sviluppo del progetto formativo.

Ringrazio di cuore la mia famiglia, che mi ha sostenuto sia moralmente che economicamente durante questo percorso di studi universitario, e Arianna che mi ha sempre accompagnato anche nei momenti di totale abbandono, senza mai pretendere troppo da me, sostenendomi sempre.

Infine, ringrazio tutti gli amici e compagni di corso per essermi stati vicini in ogni momento, per ogni aiuto e supporto durante il mio percorso universitario.

Ferrara, Febbraio 2018

Michele Vaccari



# Indice

|   |           |
|---|-----------|
| <b>Introduzione</b>                           | <b>1</b>  |
| <b>1 Localizzazione indoor</b>                | <b>3</b>  |
| 1.1 Il GPS e l'ambiente indoor . . . . .      | 3         |
| 1.2 La localizzazione indoor . . . . .        | 3         |
| <b>2 La localizzazione con RSSI</b>           | <b>5</b>  |
| 2.1 Ranging: stima della distanza . . . . .   | 5         |
| 2.2 Received Signal Strength (RSSI) . . . . . | 5         |
| 2.3 L'algoritmo di trilaterazione . . . . .   | 7         |
| <b>3 Progetto</b>                             | <b>11</b> |
| 3.1 Situazione preesistente . . . . .         | 11        |
| 3.2 Specifiche del progetto . . . . .         | 11        |
| 3.3 Vincoli del progetto . . . . .            | 12        |
| 3.4 Architettura del sistema . . . . .        | 12        |
| <b>4 Tecnologie utilizzate</b>                | <b>13</b> |
| 4.1 Bluetooth . . . . .                       | 13        |
| 4.1.1 Bluetooth 4.0 e BLE . . . . .           | 14        |
| 4.1.2 TAG BLE . . . . .                       | 14        |
| 4.2 Il sistema Android . . . . .              | 15        |
| 4.3 Protocolli . . . . .                      | 15        |
| 4.3.1 MQTT . . . . .                          | 15        |
| 4.3.2 HTTP . . . . .                          | 16        |
| 4.3.3 MQTT vs HTTP . . . . .                  | 16        |
| 4.4 NodeJS . . . . .                          | 16        |
| 4.5 JavaScript . . . . .                      | 17        |
| 4.6 SQL . . . . .                             | 17        |
| 4.7 Single Page Application . . . . .         | 17        |
| 4.7.1 HTML . . . . .                          | 18        |
| 4.7.2 CSS . . . . .                           | 18        |
| 4.7.3 AJAX . . . . .                          | 18        |
| 4.8 JSON . . . . .                            | 18        |

|   |           |
|---|-----------|
| <b>5 Implementazione del sistema</b>          | <b>19</b> |
| 5.1 Sviluppo componenti del sistema . . . . . | 19        |
| 5.2 Database . . . . .                        | 20        |
| 5.3 Broker MQTT . . . . .                     | 21        |
| 5.4 Localizzatore . . . . .                   | 22        |
| 5.4.1 Conversione da RSSI a metri . . . . .   | 22        |
| 5.4.2 Algoritmo di Trilaterazione . . . . .   | 22        |
| 5.5 Web Service . . . . .                     | 24        |
| 5.6 Single Page Application . . . . .         | 24        |
| 5.7 Applicazione Android . . . . .            | 25        |
| <b>6 Risultati sperimentali</b>               | <b>27</b> |
| 6.1 Deployment del sistema . . . . .          | 27        |
| 6.2 Valutazione delle performance . . . . .   | 28        |
| 6.2.1 Accuratezza . . . . .                   | 28        |
| 6.2.2 Complessità . . . . .                   | 28        |
| 6.2.3 Robustezza . . . . .                    | 29        |
| 6.2.4 Scalabilità . . . . .                   | 29        |
| 6.2.5 Costo . . . . .                         | 29        |
| <b>Conclusioni</b>                            | <b>31</b> |
| <b>Bibliografia</b>                           | <b>33</b> |

# Introduzione

Il tracciamento di una risorsa, ovverosia la possibilità di conoscerne la posizione geografica e di tracciarne gli spostamenti in tempo reale, è una necessità che si sta affermando prepotentemente nel mondo odierno. Molte delle applicazioni che si trovano in uno smartphone sfruttano, volontariamente o meno, la posizione del dispositivo per migliorare i servizi offerti dai vari applicativi.

Nella maggioranza dei dispositivi è presente un ricevitore GPS che consente di localizzare, tramite segnale satellitare, un terminale in uno spazio aperto. Tuttavia questa tecnologia non è disponibile quando si è all'interno di uno spazio chiuso in cui non è possibile utilizzare il segnale satellitare, che risulterebbe troppo attenuato per fornire indicazioni di posizionamento precise. In questo caso, è necessario adottare tecnologie di localizzazione espressamente pensate per ambienti indoor.

La società Evomatic s.r.l., nello sviluppo della propria piattaforma per la gestione delle risorse in movimento, utilizza in modo massivo la tecnologia GPS per consentire la geolocalizzazione delle risorse dei propri clienti in campo aperto. L'interesse della società è quello di estendere i servizi offerti dalla piattaforma anche in ambito indoor, con il riutilizzo della maggior parte delle tecnologie impiegate per lo sviluppo dei servizi di localizzazione in campo aperto, quali l'uso di TAG Bluetooth Low Energy (BLE) e di smartphone con sistema Android.

Questa tesi descrive la realizzazione di un sistema software per la localizzazione indoor offrendo una soluzione che fa uso di tecnologia BLE presente all'interno di molti smartphone Android disponibili in commercio. Le attività di progettazione e sviluppo del sistema sono state svolte durante il tirocinio presso la divisione sistemi di localizzazione satellitare e Internet-of-Things di Evomatic s.r.l., che si occupa della gestione e della manutenzione dei servizi in produzione nella propria piattaforma e, inoltre, svolge le attività di ricerca e sviluppo per l'integrazione di nuove funzionalità fruibili dai vari clienti nella piattaforma stessa.

Il sistema sviluppato è composto da vari componenti che cooperano per fornire la posizione di una risorsa in tempo reale. Dal browser si accede all'applicazione web che consente di caricare una piantina dell'ambiente in cui si desiderano monitorare le risorse. Dall'applicazione web, inoltre, è possibile gestire il posizionamento, la modifica della posizione e l'eliminazione dei TAG BLE, che operano come beacon, all'interno della piantina caricata precedentemente. Si posizionano i TAG BLE all'interno dell'ambiente indoor coerentemente con quanto fatto nell'applicazione web. Si avvia l'applicazione installata nello smartphone Android che permette di avviare la rilevazione della risorsa, previa autenticazione. L'applicazione Android, invia periodicamente i dati che ha raccolto dai beacon BLE per effettuare il calcolo della posizione e, se ci sono problemi di connessione, i dati

vengono inviati appena è possibile stabilire una connessione. Da quel momento è possibile visualizzare dall'applicazione web l'ultima posizione valida della risorsa all'interno della piantina con informazioni riguardo il nome della risorsa, l'orario e la data della rilevazione.

Il sistema progettato è realizzato in Java per quanto riguarda lo sviluppo dell'applicazione Android, mentre per lo sviluppo degli altri componenti si è utilizzato in modo predominante il linguaggio Javascript. Il formato comune utilizzato per l'interscambio dei dati tra i vari componenti è JSON. Per l'interscambio dei dati si sono utilizzati i protocolli MQTT e HTTP. Per le interrogazioni al database interno del sistema Android si è utilizzato il linguaggio SQL specifico per SQLite, mentre per le interrogazioni al database di Microsoft SQL Server si è utilizzato il linguaggio SQL proprietario di Microsoft.

Il funzionamento e l'efficienza del sistema progettato sono stati verificati attraverso una serie di test effettuati all'interno degli uffici della società Evomatic s.r.l.. Inizialmente si è verificato il corretto funzionamento del sistema per il rilevamento real time della posizione di una singola risorsa in una singola stanza, successivamente si è esteso il test per il rilevamento della posizione di una singola risorsa in più stanze per poi estendere il numero di risorse.

La tesi è organizzata secondo la seguente struttura: il Capitolo 1 introduce la geolocalizzazione indoor e la differenza con la geolocalizzazione in campo aperto; il Capitolo 2 introduce l'RSSI e il suo utilizzo nell'algoritmo di trilaterazione per il calcolo della posizione; il Capitolo 3 presenta i vincoli di progetto e l'architettura del sistema; il Capitolo 4 descrive le tecnologie usate nel sistema; il Capitolo 5 descrive l'implementazione dei vari componenti del sistema progettato; il Capitolo 6 riporta i risultati sperimentali che descrivono le prestazioni del sistema; infine un capitolo conclusivo riporta alcune considerazioni generali sul lavoro svolto e sviluppi futuri.

Durante la progettazione e lo sviluppo del sistema si è cercato di disaccoppiare i diversi componenti in modo da renderli meno sensibili a modifiche e possibili malfunzionamenti. Nella realizzazione del sistema si è tralasciata la possibilità di visualizzare lo storico delle posizioni di ogni risorsa, concentrandosi nella visualizzazione della posizione in real time.

# Capitolo 1

## Localizzazione indoor

La posizione di un dispositivo è un dato di centrale importanza che può essere utilizzato per ottimizzare il dispositivo stesso. Le tecnologie che consentono di ottenere la posizione di una risorsa sono diverse nel caso in cui ci troviamo in campo aperto o in campo chiuso.

L'analisi della tecnologia GPS permette di comprendere come si ottiene la posizione di un dispositivo in campo aperto e le problematiche per cui questa tecnica non può essere utilizzata in campi chiusi come gli ambienti indoor.

Per poter localizzare un dispositivo in ambiente indoor è possibile utilizzare altre tecniche basate anch'esse su portanti radio ma che non utilizzano una rete di satelliti in orbita.

### 1.1 Il GPS e l'ambiente indoor

Il Global Positioning System (GPS) [16] è un sistema di navigazione globale basato su satelliti in orbita. La tecnologia GPS consente di ottenere un'informazione precisa circa la posizione e l'orario in cui un determinato ricevitore effettua la richiesta di localizzazione. Questo è possibile grazie a una rete satellitare in funzione a ogni ora del giorno in ogni condizione climatica. La localizzazione tramite GPS, pertanto, è attendibile a patto che il ricevitore riesca a captare il segnale da almeno quattro satelliti distinti.

In un ambiente più complesso, costituito da muri, aree chiuse, come può essere un palazzo o un ospedale, il segnale GPS risente di forti attenuazioni. Per questo motivo la localizzazione mediante segnale satellitare risulta inefficace.

Questo non vuol dire che è impossibile ottenere la posizione di un dispositivo all'interno di un'area indoor ma significa che la tecnologia GPS è una tecnologia sfruttabile in modo efficiente solo in campo aperto.

### 1.2 La localizzazione indoor

Il mondo della localizzazione indoor si basa sul concetto di sfruttare l'ambiente e le tecnologie presenti, o comunque di facile reperibilità, per avere la possibilità di localizzare una risorsa anche in ambienti interni. In molti casi pratici l'ecosistema utile alla localizzazione viene creato sfruttando combinazioni delle tecnologie dei

dispositivi già presenti in ambiente indoor che sfruttano la portante radio, al fine di ottenere il massimo risultato dai benefici derivanti dall'uso di una tecnica, minimizzando i limiti. Questo approccio è favorito dalla larga diffusione di dispositivi che dispongono nativamente di buona parte di queste tecnologie.

È importante osservare che il risultato ottenuto da queste tecniche di localizzazione non fornisce un dato di posizione “assoluto”, come normalmente restituito da un’interrogazione GPS, ma viene fornita un’informazione di posizione “relativa” che deve poi essere interpretata correttamente per far comprendere all’utente dove è localizzato.

Le tecniche attualmente più utilizzate sfruttano tecnologie come: infrarossi (IR), Bluetooth, identificazione a radio frequenza (RFID), ultrasuoni, tecniche di riconoscimento mediante tracciamento ottico e tecniche basate sui segnali wireless (RSS techniques).

In seguito si è scelto di approfondire, rispetto alle altre tecniche, la localizzazione con RSSI, data la maggior reperibilità della tecnologia che consente di sviluppare questa tecnica. L’hardware Bluetooth, infatti, è comunemente integrato in molti dispositivi e smartphone.

# Capitolo 2

## La localizzazione con RSSI

Una delle tecniche di localizzazione più comuni per la stima della posizione in ambiente indoor è la tecnica *Anchor based*. Nella tecnica Anchor based un sottinsieme di nodi della rete, chiamati *anchor* o *beacon*, sono a conoscenza della loro posizione, ottenuta grazie al GPS o avendo stabilito un sistema di coordinate predefinite. Gli altri nodi (*target*) usano le informazioni di posizione dai nodi anchor per determinare la propria posizione. Il ranging tra i beacon e gli altri nodi può essere ottenuto con

- interazione diretta (Single-Hop)
- indirettamente, per mezzo di nodi intermedi (Multi-Hop)

Per quanto riguarda il processo di localizzazione, questo può essere affrontato usando la tecnica *Range based*. Nella tecnica Range based vengono effettuate una serie di misurazioni per conoscere la distanza tra beacon e target (*ranging*). A questo punto, con l'algoritmo di trilaterazione (vedi paragrafo 2.3) è possibile stimare la posizione dei nodi target.

### 2.1 Ranging: stima della distanza

Per stimare la distanza tra beacon e target si è utilizzata una tecnica di tipo range-based dato che è una tra le tecniche più economiche, essendo l'economicità uno dei requisiti fondamentali e maggiormente apprezzabili quando si va a verificare un'applicazione. Il primo passo da affrontare in un processo di localizzazione di questo tipo è il ranging. Il ranging tra due nodi è dunque la tecnica utilizzata da questi per determinare la distanza che si separa. Una delle tecniche su cui ci si può basare per la stima della distanza è la valutazione dell'intensità del segnale ricevuto (RSSI).

### 2.2 Received Signal Strength (RSSI)

Valutando l'intensità di un segnale ricevuto è possibile stimare la distanza dalla stazione che l'ha trasmesso. Nel caso di dispositivi Bluetooth quello che si va ad utilizzare è un indicatore di potenza, l'RSSI (Received Signal Strength Indicator),

il quale indica una stima della potenza del segnale ricevuto misurato in dBm. Per andare a valutare la distanza dalla quale questo segnale è stato trasmesso si può utilizzare l'equazione di Friis:

$$P_R = P_T \frac{G_T G_R \lambda^2}{(4\pi)^2 d^n} \quad (2.1)$$

dove:

- $P_R$  è la potenza del segnale ricevuto in Watt
- $P_T$  è la potenza del segnale trasmesso in Watt
- $G_R$  è il guadagno dell'antenna ricevente
- $G_T$  è il guadagno dell'antenna trasmittente
- $\lambda$  è la lunghezza d'onda dove  $\lambda = c/f$  dove  $c$  è la velocità della luce (299792458 m/s) e  $f$  è la frequenza
- $d$  è la distanza in metri
- $n$  è la costante di propagazione del segnale, detto anche esponente di propagazione, dipende dall'ambiente in cui ci si trova

Risulta molto utile, dal punto di vista pratico la seguente formula, che ci permette di convertire la potenza da Watt a dBm, dal momento che l'indicatore RSSI è solitamente espresso in dBm.

$$P[dBm] = 10 \cdot \log_{10}(P[W] \cdot 10^3) \quad (2.2)$$

Per motivi di semplicità, è possibile combinare assieme la 2.1 e la 2.2. Applicando le proprietà dei logaritmi, si ottiene la seguente formula:

$$RSSI = -(10 \cdot n \cdot \log_{10}d - A) \quad (2.3)$$

Dove  $A$  è la potenza del segnale ricevuto, in dBm, alla distanza di riferimento di un metro e  $n$  è la costante di propagazione del segnale.

In figura 2.1 è rappresentato l'andamento ideale dell'indicatore RSSI in ambiente aperto ( $n = 2$ ,  $A = -40dBm$ ).

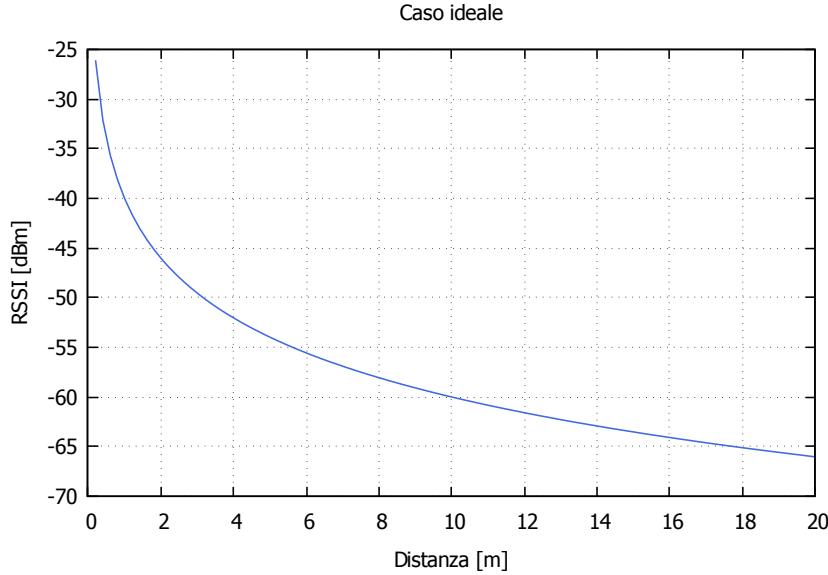
A questo punto è conveniente aggiustare la relazione sopra scritta:

$$\log_{10}d = \frac{A - RSSI}{10 \cdot n} \quad (2.4)$$

Infine, isolando la distanza:

$$d = 10^{(\frac{A - RSSI}{10 \cdot n})} \quad (2.5)$$

Quest'ultima formula, ci permette quindi di calcolare la distanza tra due nodi conoscendo il valore di RSSI ed i parametri  $A$  e  $n$ .



**Figura 2.1:** Andamento ideale della potenza ricevuta in funzione della distanza

## 2.3 L'algoritmo di trilaterazione

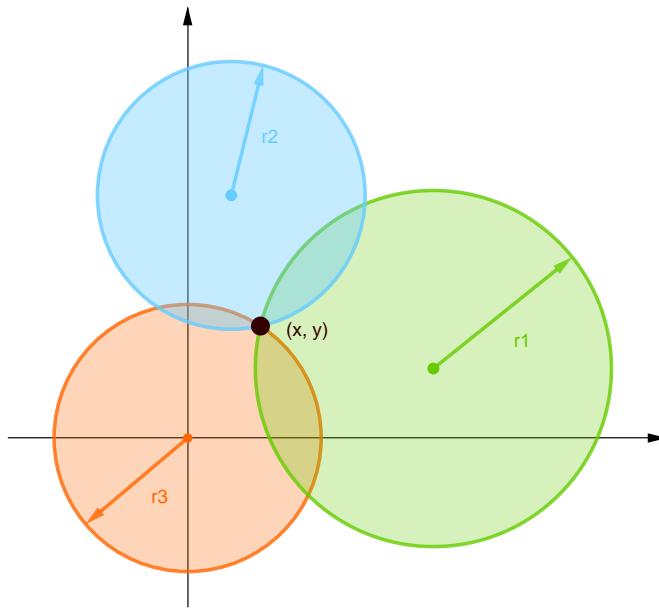
Uno degli algoritmi più utilizzati per la localizzazione è l'algoritmo di trilaterazione. Questo algoritmo non necessita di mappature del segnale radio, riducendo così fortemente il tempo richiesto per la calibrazione e, allo stesso tempo, garantendo maggior flessibilità al dinamismo e alle modifiche dell'ambiente. La trilaterazione è un algoritmo di localizzazione basato su principi geometrici. Per poter calcolare la posizione di un dispositivo si deve conoscere la distanza dal dispositivo da ogni beacon. Vengono quindi disegnati dei cerchi avente come centro le coordinate del nodo riferimento e raggio uguale alla distanza stimata. Idealmente, il nodo target si trova nel punto di intersezione tra i tre cerchi come mostrato in figura 2.2. Sapendo che ogni circonferenza è descritta dall'equazione

$$(x - x_i)^2 + (y - y_i)^2 = r_i^2$$

con  $(x_i, y_i)$  coordinate dei vari centri e  $r_i$  raggio del cerchio  $i$ -esimo, per trovare l'intersezione basta risolvere il seguente sistema:

$$\begin{cases} (x - x_1)^2 + (y - y_1)^2 = r_1^2 \\ (x - x_2)^2 + (y - y_2)^2 = r_2^2 \\ (x - x_3)^2 + (y - y_3)^2 = r_3^2 \end{cases} \quad (2.6)$$

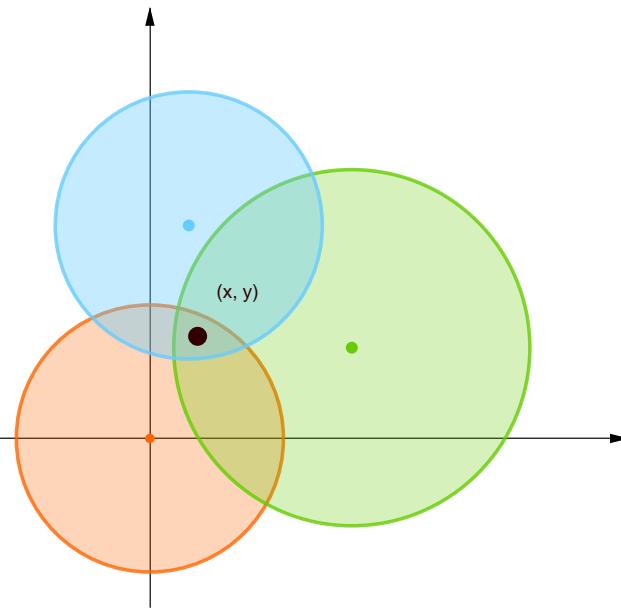
In pratica, tuttavia, le stime della distanza non sono mai così perfette e non viene quasi mai ottenuto un unico punto come intersezione. È più probabile che il nodo target si trovi all'interno di un'area di intersezione tra i cerchi, o è addirittura possibile che i cerchi non si tocchino. Per ovviare a questi inconvenienti si utilizzano dei piccoli accorgimenti al metodo tradizionale della Trilaterazione. Questo metodo che andremo a descrivere viene realizzato iterativamente per tutti e tre i nodi beacon.



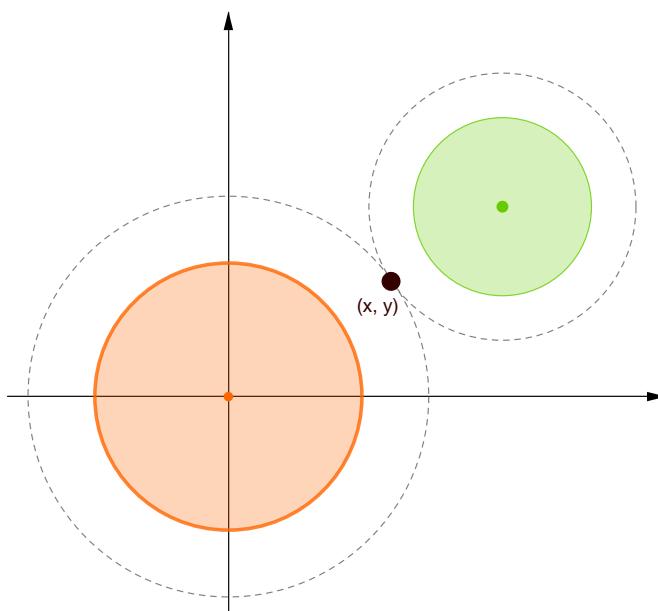
**Figura 2.2:** Le coordinate del nodo target si trovano nel punto di intersezione dei tre cerchi

1. Vengono disegnati due cerchi attorno a due nodi beacon a scelta;
2. Se il punto di intersezione è unico, si salvano le coordinate di questo;
3. Se i due cerchi non si toccano, vengono aumentati i raggi in maniera proporzionale in modo che si tocchino in un unico punto;
4. Se i due cerchi si toccano in due punti, viene considerato quello che ha distanza dal terzo nodo beacon più vicina alla stima della distanza calcolata per questo nodo;
5. Alla fine di questo procedimento, iterato per tutti e tre i nodi, viene calcolata la media delle coordinate, che corrispondono alla posizione stimata del nodo.

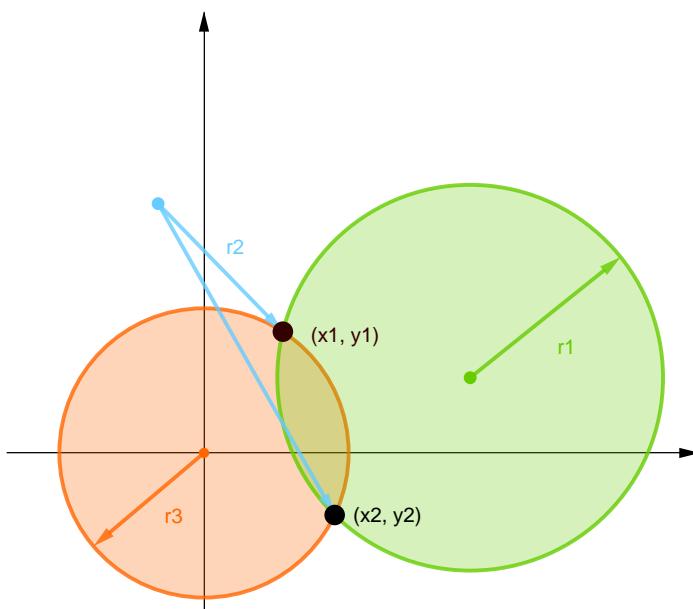
Questo algoritmo è leggermente più complesso, almeno in linea di principio, ma offre performance migliori. In figura 2.3, figura 2.4 e figura 2.5 è mostrato come opera questo adattamento dell'algoritmo di Trilaterazione.



**Figura 2.3:** Esempio del caso in cui il punto di intersezione non sia unico.



**Figura 2.4:** Esempio del caso in cui i cerchi non si tocchino. Vengono aumentati i raggi in maniera proporzionale fin quando non si raggiunge il punto di incontro.



**Figura 2.5:** Esempio del caso in cui vi siano due punti di intersezione. Il punto tenuto in considerazione è quello con distanza più vicina a quella stimata.

# Capitolo 3

## Progetto

La fase di progettazione del sistema si è svolta partendo dall’analisi dei requisiti del progetto sulla base di tecnologie preesistenti. I vincoli e gli obiettivi del progetto sono stati definiti all’inizio dell’esperienza di tirocinio, svolta all’interno della società Evomatic s.r.l. di Rovigo.

### 3.1 Situazione preesistente

La società Evomatic s.r.l. è una società che commercializza la propria piattaforma web per la gestione e il tracciamento delle risorse in movimento. La piattaforma utilizza in modo massivo la tecnologia GPS per il tracciamento delle risorse in campo aperto. Il tracciamento delle risorse, sprovviste di tecnologia GPS, può avvenire tramite l’installazione di opportuni moduli GPS oppure si può utilizzare direttamente l’antenna GPS di uno smartphone Android. L’invio dei dati di posizione delle risorse verso la piattaforma avviene in entrambi i casi sfruttando la connessione internet dello smartphone Android. Oltre alla tracciabilità delle risorse, la piattaforma offre la possibilità di fornire un controllo sui mezzi. Il controllo sui mezzi è possibile tramite l’uso di TAG BLE, che implementano, tramite un’applicazione Android che si interfaccia con la piattaforma, le funzionalità di identificazione dell’utilizzatore della risorsa consentendo, o meno, di usare le funzionalità della risorsa.

### 3.2 Specifiche del progetto

La piattaforma esistente sfrutta la tecnologia GPS per la tracciabilità delle risorse in campo aperto. La società Evomatic s.r.l. ha espresso l’esigenza di avere la possibilità di estendere le funzionalità della piattaforma anche in ambiente indoor, utilizzando la maggior parte delle tecnologie impiegate per l’uso in campo aperto. L’obiettivo del progetto è di realizzare un sistema software per il tracciamento indoor delle risorse, con utilizzo della tecnologia BLE (*Bluetooth Low Energy*).

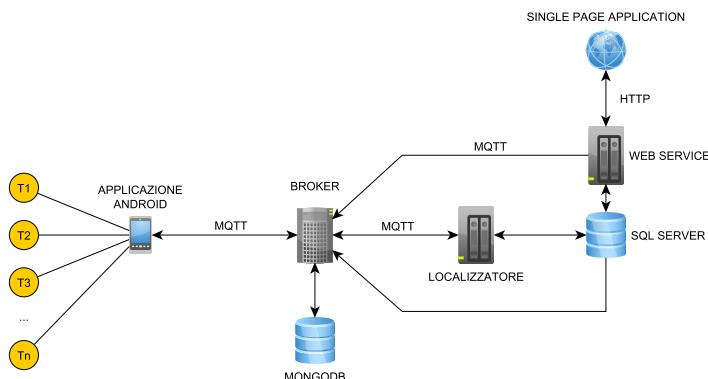
### 3.3 Vincoli del progetto

I vincoli del sistema sono stati definiti attraverso vari incontri con il responsabile area ricerca e sviluppo e con il tutor aziendale. Al termine di questi incontri sono state stilate le caratteristiche che deve avere il sistema:

- Il sistema deve avere un’interfaccia web, che consenta la configurazione e la gestione dei TAG BLE, che operano come beacon, in un ambiente indoor;
- Il sistema, tramite l’interfaccia web, deve dare la possibilità di visualizzare le varie risorse in movimento all’interno dell’ambiente indoor;
- Il sistema deve implementare un’applicazione su smartphone Android che consenta di identificare la risorsa e di avviare o meno il tracciamento;
- L’applicazione Android deve essere ottimizzata per consentire l’utilizzo dell’applicazione in modo che lo smartphone, con batteria iniziale totalmente carica, non esaurisca tutta la carica a disposizione prima di 8 ore di attività continua;
- L’applicazione Android deve trasmettere in tempo reale la propria posizione sfruttando la connessione internet dello smartphone e se, per qualsiasi motivo, la connessione internet viene a mancare, il dispositivo Android deve poter bufferizzare i dati rilevati per poi inviarli appena possibile;
- Tutti i dati riguardanti la posizione di ogni risorsa devono essere memorizzati in un database. Il motore di database, per mantenere la compatibilità con il database esistente della piattaforma, dev’essere Microsoft SQL Server.

### 3.4 Architettura del sistema

L’architettura del sistema progettata per la localizzazione indoor utilizzando TAG BLE e smartphone Android è schematizzata in figura 3.1. Rispettando i vincoli di progetto, si sono progettati: il database centrale, l’applicazione Android, il broker MQTT, il localizzatore, il web service e la single page application.



**Figura 3.1:** Architettura del sistema progettato

# Capitolo 4

## Tecnologie utilizzate

La tecnologia principale con cui viene effettuata la stima della distanza è la tecnologia BLE. La tecnologia BLE è presente sia sui TAG, dislocati nell'ambiente indoor, sia sullo smartphone, usato per identificare la risorsa in movimento. Per lo sviluppo dell'applicazione sullo smartphone si è analizzata la struttura del sistema Android per poi implementare l'applicazione in linguaggio Java. La scelta dei protocolli da utilizzare a livello applicativo è ricaduta su HTTP, per le comunicazioni effettuate attraverso l'interfaccia web, e MQTT, per le comunicazioni effettuate attraverso lo smartphone Android. I fattori che hanno portato a scegliere MQTT, rispetto ad altri protocolli, sono la leggerezza e l'affidabilità oltre al fatto di essere uno standard definito e aperto.

Per lo sviluppo dei vari componenti si è utilizzato Node.js, runtime Javascript costruito sul motore JavaScript V8 di Google Chrome.

I database utilizzati per la memorizzazione dei dati sono di tipo relazionale e in particolare si è utilizzato SQLite per la memorizzazione dei dati nel sistema Android mentre si è utilizzato Microsoft SQL Server per la memorizzazione delle posizioni di tutte le risorse.

Dato che si sono utilizzati database relazionali, SQLite per Android e Microsoft SQL Server per la gestione della persistenza del sistema, per le interrogazioni si utilizza il linguaggio SQL.

Per l'applicazione web si è scelto l'approccio *Single Page Application* con il fine di favorire l'interattività. Per realizzare l'interfaccia web si sono utilizzate diverse tecnologie, quali: HTML, CSS, AJAX, Canvas.

Il formato comune utilizzato per l'interscambio dei dati tra i vari componenti è il JSON.

Di seguito si descrivono brevemente le varie tecnologie utilizzate.

### 4.1 Bluetooth

Nelle telecomunicazioni, Bluetooth [3] è uno standard tecnico-industriale di trasmissione dati per reti personali senza fili. Fornisce un metodo standard, economico e sicuro per scambiare informazioni tra dispositivi diversi attraverso una frequenza radio sicura a corto raggio. La specifica Bluetooth è stata sviluppata da Ericsson e in seguito formalizzata dalla Bluetooth Special Interest Group (SIG). Il protocollo

Bluetooth opera nel campo di frequenze assegnato intorno ai 2,45 GHz. Per ridurre le interferenze il protocollo divide la banda in 79 canali e provvede a commutare tra i vari canali 1.600 volte al secondo (frequency hopping).

Nel corso degli anni sono state emesse diverse versioni del protocollo ognuna delle quali implementa diverse specifiche. Ogni dispositivo Bluetooth è configurabile per cercare costantemente altri dispositivi e per collegarsi a questi.

### 4.1.1 Bluetooth 4.0 e BLE

In data 6 luglio 2010 sono diventate definitive le specifiche della versione 4.0.

Rispetto alle versioni precedenti, la versione 4.0 punta alla riduzione dei consumi energetici. L'obiettivo principale di questa funzionalità opzionale, denominato Low Energy (LE), tramite un'ottimizzazione della struttura di trama e l'impiego di dispositivi più efficienti, comporta una riduzione della velocità, che in questa modalità si attesta a 1 Mbps. In termini trasmissivi, sono stati potenziati i meccanismi di rilevazione e correzione di errore e di cifratura del segnale col supporto di AES-128.

### 4.1.2 TAG BLE

Un TAG BLE [6] è un beacon BLE che può contenere, inoltre, un accelerometro, un sensore di temperatura e di umidità. In figura 4.1 è possibile vedere com'è fatto un TAG BLE commercializzato da Global Tag.

Un TAG BLE dispone di un pulsante ON/OFF che consente di attivare o disattivare il tag, al fine di permettere all'utente di risparmiare la batteria, qualora non abbia la necessità di mantenere il tag acceso. La distanza di lettura arriva fino a 90 metri. I diversi parametri del beacon BLE, come l'intervallo di emissione, la potenza del segnale, il “friendly name” si possono configurare tramite software.



**Figura 4.1:** TAG BLE

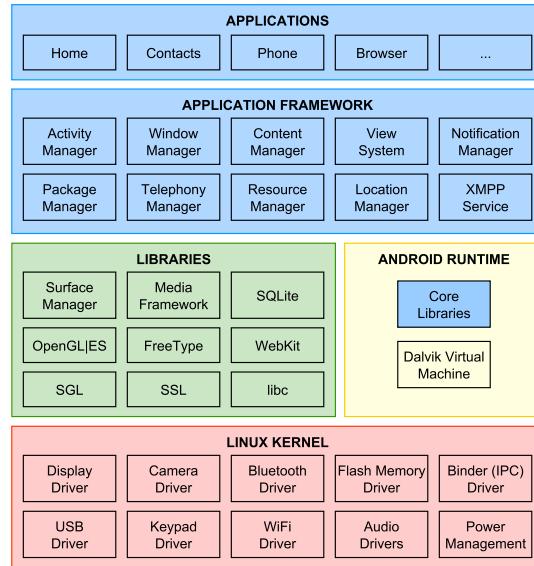
## 4.2 Il sistema Android

Android [2] è un sistema operativo per dispositivi mobili sviluppato da Google Inc. basato sul kernel Linux. È un sistema embedded progettato principalmente per smartphone e tablet.

Lo sviluppo di Android prosegue attraverso l'Android Open Source Project il quale è software libero ad esclusione di diversi firmware non-liberi inclusi per i produttori di dispositivi e delle cosiddette “Google Apps” come ad esempio Google Play. È distribuito sotto i termini della licenza libera Apache 2.0 riservandosi di non-includere software coperto da licenze copyleft.

Android è costituito da un kernel Linux 2.6 e 3.x (da Android 4.0 in poi), con middleware, librerie e API scritte in C (o C++) e software in esecuzione su un framework di applicazioni che include librerie Java compatibili con librerie basate su Apache Harmony.

La piattaforma hardware principale di Android è l'architettura ARM mentre le applicazioni Android sono Java-based. L'architettura software del sistema Android è schematizzata in figura 4.2.



**Figura 4.2:** Architettura del sistema Android

## 4.3 Protocolli

### 4.3.1 MQTT

MQ Telemetry Transport (MQTT) [11] è un protocollo ISO standard di messaggistica leggero di tipo publish-subscribe posizionato in cima a TCP/IP. Il protocollo è stato inventato da Andy Stanford-Clark di IBM, e Arlen Nipper di Cirrus Link Solutions nel 1999.

MQTT è stato progettato per avere bassa latenza, messaggistica sicura su reti fragili e una distribuzione efficiente a uno o più ricevitori. Il protocollo si concentra

**Tabella 4.1:** Test di spedizione di 1024 messaggi con payload da 1 byte con HTTPS e MQTT

| Connessione            | 3G       |           | Wifi     |           |
|------------------------|----------|-----------|----------|-----------|
| Protocollo             | HTTPS    | MQTT      | HTTPS    | MQTT      |
| % Batteria / Ora       | 18.43%   | 16.13%    | 3.45%    | 4.23%     |
| Messaggi / Ora         | 1708     | 160278    | 3628     | 263314    |
| % Batteria / Messaggio | 0.01709  | 0.00010   | 0.00095  | 0.00002   |
| Messaggi Ricevuti      | 240/1024 | 1024/1024 | 524/1024 | 1024/1024 |

sulla riduzione al minimo della quantità di byte necessari per l'invio del messaggio e sul basso consumo energetico. La dimensione massima dei messaggi di 256 mb, quindi non è possibile inviare per ogni messaggio grandi quantità di dati ma è possibile inviare un numero elevato di messaggi di dimensioni ridotte. MQTT fornisce, inoltre, un canale di comunicazione bidirezionale. Questo protocollo viene utilizzato in situazioni in cui è richiesto un basso impatto e dove la banda è limitata. Per poter essere utilizzato il pattern publish-subscribe richiede un *message broker* il quale è responsabile della distribuzione dei messaggi ai client destinatari.

### 4.3.2 HTTP

L'HyperText Transfer Protocol (HTTP) [8] è un protocollo a livello applicativo usato come principale sistema per la trasmissione d'informazioni sul web ovvero in un'architettura tipica client-server. Le specifiche del protocollo sono gestite dal World Wide Web Consortium (W3C). Un server HTTP generalmente resta in ascolto delle richieste dei client sulla porta 80 usando il protocollo TCP a livello di trasporto. Nell'HTTP le connessioni vengono generalmente chiuse una volta che una particolare richiesta, o una serie di richieste correlate, è stata soddisfatta.

### 4.3.3 MQTT vs HTTP

Uno dei fattori per cui si è scelto MQTT nell'applicazione Android è dato dal minor consumo di batteria e alla maggiore leggerezza ed affidabilità nella connessione. In tabella 4.1 vengono riportati i risultati di un test [17] di spedizione di 1024 messaggi con payload di 1 byte. Come si può notare dai dati in tabella utilizzando MQTT si ha una totale consegna dei messaggi inviati oltre avere un consumo di batteria inferiore per ogni messaggio inviato rispetto ad HTTPS.

## 4.4 NodeJS

Node.js [12] è un runtime Javascript costruito sul motore JavaScript V8 di Google Chrome. Il modello utilizzato da Node.js è il modello I/O non bloccante ad eventi,

che lo rende un framework leggero ed efficiente. Il modello I/O non bloccante richiede al sistema operativo di ricevere notifiche al verificarsi di determinati eventi, rimanendo quindi in sleep fino alla notifica stessa. Quando la notifica avviene, allora il sistema operativo torna attivo per eseguire le istruzioni previste nella funzione di callback, funzione che viene eseguita una volta che il risultato dell'elaborazione da parte del sistema operativo è disponibile.

Node.js è organizzato in moduli i quali sono scritti in JavaScript. Ogni sviluppatore può scrivere nuovi moduli in JavaScript oppure può includere dei moduli già scritti prelevandoli da npm, gestore dei pacchetti predefinito di Node.js.

## 4.5 JavaScript

JavaScript [10] è un linguaggio di scripting orientato agli oggetti e agli eventi, comunemente utilizzato nella programmazione Web lato client per la creazione, in siti web e applicazioni web, di effetti dinamici interattivi tramite funzioni di script invocate da eventi innescati a loro volta in vari modi dall'utente sulla pagina web in uso. Tali funzioni di script, utilizzati dunque nella logica di presentazione, possono essere opportunamente inserite in file HTML. JavaScript è stato standardizzato per la prima volta il 1997 dalla ECMA con il nome ECMAScript ed è, inoltre, uno standard ISO.

## 4.6 SQL

SQL (Structured Query Language) [19] è un linguaggio standardizzato per database basati sul modello relazionale (RDBMS) progettato per:

- Creare e modificare schemi di database (Data Definition Language);
- Inserire, modificare e gestire dati memorizzati (Data Manipulation Language);
- Interrogare i dati memorizzati (Data Query Language);
- Creare e gestire strumenti di controllo ed accesso ai dati (Data Control Language).

SQL non è solo di un semplice linguaggio di interrogazione, ma alcuni suoi sottosistemi si occupano della creazione, della gestione e dell'amministrazione dei database.

## 4.7 Single Page Application

Con Single Page Application [15] si intende un'applicazione web che può essere usata o consultata su una singola pagina web con l'obiettivo di fornire un'esperienza utente più fluida, simile alle applicazioni desktop dei sistemi operativi tradizionali. In un'applicazione su singola pagina tutto il codice necessario è recuperato in un

singolo caricamento della pagina mentre le risorse appropriate sono caricate dinamicamente e aggiunte alla pagina quando necessario, di solito in risposta ad azioni dell’utente. In particolare, la pagina non si ricarica in nessun punto del processo, né lascia il controllo a un’altra pagina. Di seguito si presentano brevemente le tecnologie con cui si sviluppa un’applicazione web di questo tipo.

### 4.7.1 HTML

L’HTML (HyperText Markup Language) [7] è un linguaggio di markup. Nato per la formattazione e impaginazione di documenti ipertestuali disponibili nel web 1.0, oggi è utilizzato principalmente per il disaccoppiamento della struttura logica di una pagina web. L’HTML è un linguaggio di pubblico dominio, la cui sintassi è stabilita dal World Wide Web Consortium (W3C). È derivato da SGML, un metalinguaggio finalizzato alla definizione di linguaggi utilizzabili per la stesura di documenti destinati alla trasmissione in formato elettronico.

### 4.7.2 CSS

Il CSS (Cascading Style Sheets) [5] è un linguaggio usato per definire la formattazione di documenti HTML, XHTML e XML. L’introduzione del CSS si è resa necessaria per separare i contenuti delle pagine HTML dalla loro formattazione e permettere una programmazione più chiara e facile da utilizzare, sia per gli autori delle pagine stesse sia per gli utenti, garantendo contemporaneamente anche il riutilizzo di codice e una sua più facile manutenzione.

### 4.7.3 AJAX

AJAX (Asynchronous JavaScript and XML) [1] è una tecnica di sviluppo software per la realizzazione di applicazioni web interattive. Lo sviluppo di applicazioni HTML con AJAX si basa su uno scambio di dati in background fra web browser e server, che consente l’aggiornamento dinamico di una pagina web senza esplicito ricaricamento da parte dell’utente. AJAX è asincrono nel senso che i dati extra sono richiesti al server e caricati in background senza interferire con il comportamento della pagina esistente. Normalmente le funzioni richiamate sono scritte con il linguaggio JavaScript. Tuttavia, l’uso di JavaScript e di XML non è obbligatorio, come non è detto che le richieste di caricamento debbano essere necessariamente asincrone.

## 4.8 JSON

JSON (JavaScript Object Notation) [9] è un formato adatto all’interscambio di dati fra applicazioni client-server. È basato sul linguaggio JavaScript Standard ECMA-262 3<sup>a</sup> edizione dicembre 1999, ma ne è indipendente. Viene usato in AJAX come alternativa a XML/XSLT.

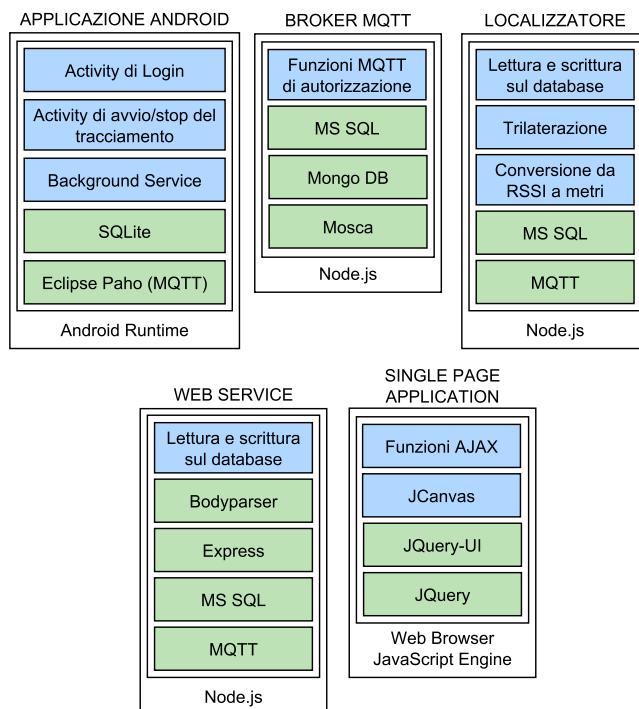
# Capitolo 5

## Implementazione del sistema

Questo capitolo descrive la fase dello sviluppo del sistema, partendo dall'architettura presentata nel paragrafo 3.4, per poi scendere nel dettaglio dei vari componenti.

### 5.1 Sviluppo componenti del sistema

Coerentemente con l'architettura illustrata in figura 3.1 si è implementato ogni componente del sistema. Per lo sviluppo dei vari componenti si sono utilizzati dei framework. In figura 5.1 si riporta il grafico color coded dei vari componenti del sistema dove in azzurro sono evidenziate le parti sviluppate mentre in verde sono indicati i framework e i moduli utilizzati. Di seguito si descrive, in ordine di sviluppo, l'implementazione di ogni componente del sistema.



**Figura 5.1:** Grafico color coded dei componenti sviluppati

## 5.2 Database

Il motore del database centrale è Microsoft SQL Server, coerentemente con quanto imposto nei vincoli di progetto. Per l'immagazzinamento dei dati si è realizzato il modello entità-relazione rappresentato in figura 5.2. Il database contiene quattro entità in relazione tra loro. Le entità sono:

**UTENTE** si occupa di memorizzare i dati relativi ad un utente che può accedere al sistema;

**TRACCIA** si occupa di memorizzare le tracce dei vari utenti;

**POSIZIONE** contiene per ogni traccia le coordinate dell'ambiente indoor;

**TAG BLE** contiene le informazioni riguardanti la posizione di ogni TAG BLE.

Le relazioni che intercorrono tra le entità sono:

**Registra** per ogni utente possono essere presenti una o più tracce con datetime diverso per ogni traccia.

**Corrisponde** ad ogni traccia applicando l'algoritmo di trilaterazione corrisponde una sola posizione e viceversa;

**Contiene** ogni traccia contiene 3 TAG BLE i quali possono avere valori di RSSI diversi ed ogni TAG BLE può essere presente in più tracce;

Utilizzando il modello ER appena presentato si è costruito il modello relazionale e, infine, si è applicata la procedura di normalizzazione per minimizzare la ridondanza dei dati.

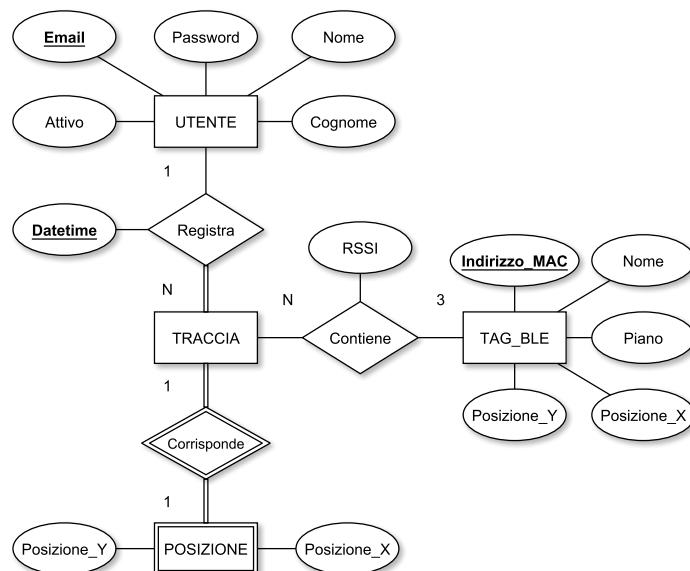


Figura 5.2: Modello E-R del database centrale

## 5.3 Broker MQTT

Per lo sviluppo del broker MQTT si è scelto di utilizzare il broker Mosca. Mosca [4] è un broker MQTT che può essere integrato in applicazioni Node.js. Il broker implementa diverse caratteristiche, tra cui: MQTT 3.1 e 3.1.1, QoS 0 e QoS 1, varie opzioni di archiviazione per i pacchetti offline QoS 1 e le sottoscrizioni. Per avere la garanzia nella comunicazione, si deve utilizzare un database. Mosca offre la compatibilità con diversi database non relazionali. Si è scelto di utilizzare il database MongoDB che si occuperà di rendere disponibili le funzioni di:

**Keep-Alive message** il broker riesce a identificare una disconnessione non esplicita del client;

**Will message** viene impostato nel messaggio di CONNECT con topic, QoS e retain. In caso di disconnessione inaspettata il messaggio “Will” viene mandato ai subscribers registrati;

**Retain message** un messaggio pubblicato su un topic viene mantenuto sul broker. Un successivo subscriber che si connette sullo stesso topic riceve il messaggio (last known good message).

**Persistent Session** dopo la disconnessione del client, tutte le sottoscrizioni vengono mantenute nel broker e recuperate alla connessione successiva.

Dopo aver integrato il broker seguendo le linee guida della documentazione, si sono riscritte le funzioni che permettono di verificare se un utente può utilizzare o meno il sistema. La funzione di autenticazione, visibile nel listato 5.1, utilizza la libreria Mssql per interfacciarsi con Microsoft SQL Server.

```

1 // Autenticazione
2 authenticate = function(client, email, password, callback) {
3   var autorizzato = false;
4   return mssql.connect(sqlConfig).then(function() {
5     var request = new mssql.Request(mssql);
6     // conta gli utenti che hanno la stessa email e password
7     var stringRq = "SELECT COUNT(*) AS presente FROM utente WHERE
8       Email = '" + email + "' AND Password = '" + password + "';";
9     // autorizzo se presente altrimenti rifiuto la connessione
10    return request.query(stringRq).then(function(result) {
11      autorizzato = result[0].presente == 1;
12      if (!autorizzato) { // email non autorizzata
13        return callback(null, autorizzato); }
14      else { // email autorizzata
15        client.user = email;
16        return callback(null, autorizzato); }
17      mssql.close(); })
18      .catch(function (err) {
19        console.log(err);
20        mssql.close(); });
21    }).catch(function (err) {
22      console.log(err); });
23  }

```

Codice 5.1: Funzione di autenticazione del broker MQTT

## 5.4 Localizzatore

Il localizzatore è un componente che si occupa di interfacciarsi con il database utilizzando il protocollo MQTT. Si è utilizzata la libreria MQTT.js che implementa le funzioni del protocollo MQTT per Node.js. Il localizzatore, al suo avvio, deve pubblicare la lista di tutti i TAG BLE presenti nel database centrale con il flag retain. Quando il broker riceve da un utente un messaggio contenente un insieme di tracce, il broker lo inoltra al localizzatore. Il localizzatore a questo punto, memorizza le tracce ricevute nel database e successivamente effettua prima la conversione da RSSI a metri, poi ottenendo la posizione relativa dei TAG BLE interessati effettua il calcolo della posizione eseguendo l'algoritmo di Trilaterazione.

### 5.4.1 Conversione da RSSI a metri

La conversione da RSSI a metri si è implementata nel localizzatore con il codice visibile nel listato 5.2, coerentemente con quanto spiegato nel paragrafo 2.1.

```

1 // Conversione da RSSI a Metri
2 calcoloRssiMetri = function(rssi) {
3   var A = -40 // potenza del segnale ricevuto in campo aperto
4   var n = 2 // costante di propagazione in campo aperto
5
6   var esponente = (A - rssi) / (10 * n);
7   return Math.pow(10, esponente);
8 }
```

**Codice 5.2:** Conversione da RSSI a Metri

### 5.4.2 Algoritmo di Trilaterazione

L'algoritmo di Trilaterazione si è implementato seguendo quanto indicato nel paragrafo 2.3. In primo luogo si implementa una funzione che permetta di calcolare il punto medio di un array di punti. La firma della funzione è visibile nel listato 5.3.

```
1 getMediaPunti = function(punti) { /* ... */ }
```

**Codice 5.3:** Calcolo del punto medio

Successivamente si definisce una funzione che ritorna il punto medio della Trilaterazione iterata per tutti e tre i nodi beacon, visibile nel listato 5.4.

```

1 trilaterazione = function(x1, y1, d1, x2, y2, d2, x3, y3, d3) {
2   return getMediaPunti(new Array(
3     getTrilaterazione(x1, y1, d1, x2, y2, d2, x3, y3, d3),
4     getTrilaterazione(x3, y3, d3, x1, y1, d1, x2, y2, d2),
5     getTrilaterazione(x2, y2, d2, x3, y3, d3, x1, y1, d1)))
6 }
```

**Codice 5.4:** Calcolo del punto medio della Trilaterazione iterata per i tre nodi beacon

Infine si scrive l'algoritmo di Trilaterazione, visibile nel listato 5.5, risolvendo il sistema di equazioni 2.6.

```

1  getTrilaterazione = function(x1, y1, d1, x2, y2, d2, x3, y3, d3) {
2    var r = new Object(); // è l'oggetto da ritornare
3    // verifico se i due cerchi si incrociano
4    if (Math.sqrt(Math.pow(x1-x2,2) + Math.pow(y1-y2,2)) <= (d1+d2)) {
5      var a1=-2*x1; var b1=-2*y1; var a2=-2*x2; var b2=-2*y2;
6      var c1=Math.pow(x1,2)+Math.pow(y1,2)-Math.pow(d1,2);
7      var c2=Math.pow(x2,2)+Math.pow(y2,2)-Math.pow(d2,2);
8      if(a1 != a2) { // ricavo le coordinate del polinomio
9        var a=1+Math.pow(b2-b1,2)/Math.pow(a1-a2,2);
10       var b=2*(b2-b1)*(c2-c1)/Math.pow(a1-a2,2)+(a1*(b2-b1))/(a1-a2)+b1;
11       var c=Math.trunc(c1+a1*(c2-c1)/(a1-a2)+Math.pow(c2-c1,2)/Math.pow(
12         a1-a2,2));
13       // risolvo l'eq. di secondo grado a*x^2 + b*x + c
14       var delta = Math.sqrt(Math.pow(b,2) - 4*a*c);
15       var ys1=(-b+delta)/(2*a); var ys2=(-b-delta)/(2*a);
16       // ricavo i due valori di x risolvendo l'eq.
17       var xs1=(c2-c1)/(a1-a2)+((b2-b1)*ys1)/(a1-a2);
18       var xs2=(c2-c1)/(a1-a2)+((b2-b1)*ys2)/(a1-a2);
19       // se le due soluzioni sono uguali ho il punto di intersezione
20       if(xs1==xs2 && ys1==ys2) { r.x=xs1; r.y=ys1; return r; }
21     else { // altrimenti la distanza del nodo stimato dal terzo anchor
22       d1=Math.sqrt(Math.pow(xs1-x3,2)+Math.pow(ys1-y3,2));
23       d2=Math.sqrt(Math.pow(xs2-x3,2)+Math.pow(ys2-y3,2));
24       if ((d1-d3)<(d2-d3)) { r.x=xs1; r.y=ys1; return r; }
25     else { r.x=xs2; r.y=ys2; return r; }
26   }
27   // ricavo le coordinate del polinomio
28   var a=1+Math.pow(a2-a1,2)/Math.pow(b1-b2,2);
29   var b=2*(a2-a1)*(c2-c1)/Math.pow(b1-b2,2)+(b1*(a2-a1))/(b1-b2)+a1;
30   var c=Math.trunc(c1+b1*(c2-c1)/(b1-b2)+Math.pow(c2-c1,2)/Math.pow(
31     b1-b2,2));
32   // risolvo l'eq. di secondo grado a*x^2 + b*x + c
33   var delta=Math.sqrt(Math.pow(b,2)-4*a*c);
34   var xs1=(-b+delta)/(2*a); var xs2=(-b-delta)/(2*a);
35   // ricavo i due valori di y risolvendo l'eq.
36   var ys1=(c2-c1)/(b1-b2)+((a2-a1)*xs1)/(b1-b2);
37   var ys2=(c2-c1)/(b1-b2)+((a2-a1)*xs2)/(b1-b2);
38   // se le due soluzioni sono uguali ho il punto di intersezione
39   if(xs1==xs2 && ys1==ys2) { r.x=xs1; r.y=ys1; return r; }
40   else { // altrimenti la distanza del nodo stimato dal terzo anchor
41     d1=Math.sqrt(Math.pow(xs1-x3,2)+Math.pow(ys1-y3,2));
42     d2=Math.sqrt(Math.pow(xs2-x3,2)+Math.pow(ys2-y3,2));
43     if ((d1-d3)<(d2-d3)) { r.x=xs1; r.y=ys1; return r; }
44   else { r.x = xs2; r.y = ys2; return r; }
45 }
46 // nuovi raggi
47 d1 += dr; d2 += dr;
48 return getTrilaterazione(x1, y1, d1, x2, y2, d2, x3, y3, d3);
49 }
```

Codice 5.5: Algoritmo di trilaterazione

## 5.5 Web Service

Il web service si occupa di rendere disponibili le varie funzioni per la Single Page Application. Per lo sviluppo si sono utilizzate le seguenti librerie:

**MQTT.js** per pubblicare al broker MQTT la lista dei TAG BLE, se questi vengono modificati dalla Single Page Application, con il flag retain;

**Express** per realizzare le varie funzioni REST del webservice;

**Body-parser** per effettuare il parsing middleware del corpo del messaggio ricevuto dalla Single Page Application;

**Mssql** per interfacciarsi con Microsoft SQL Server.

Si sono sviluppate le varie API del web service utilizzando prevalentemente le librerie Express e Mssql, effettuando delle interrogazioni SQL specifiche per poi inviare alla Single Page Application i dati richiesti.

## 5.6 Single Page Application

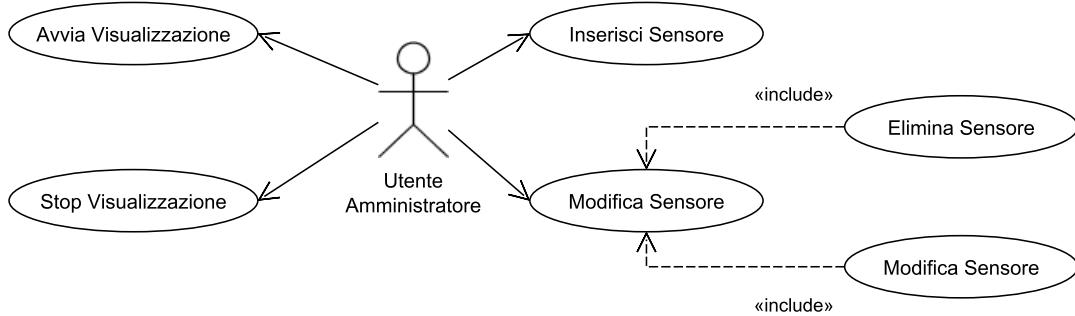
La Single Page Application offre all'utente la possibilità di configurare i TAG BLE e di vedere le risorse all'interno dell'ambiente indoor desiderato. I casi d'uso implementati sono visibili in figura 5.3. Per lo sviluppo dell'applicazione si sono utilizzati i seguenti framework:

**JQuery** utilizzato per le chiamate AJAX e per il supporto multibrowser;

**JQuery-UI** utilizzato per creare le varie maschere di inserimento e modifica dei dati riguardanti alla configurazione dei TAG BLE;

**JCanvas** utilizzato per disegnare sulla mappa dell'ambiente indoor e per animare i movimenti real time delle varie risorse monitorate.

Si è utilizzato, inoltre, a corredo dei framework JavaScript sopra citati anche il linguaggio HTML e CSS per definire la struttura e l'aspetto grafico dell'applicazione.



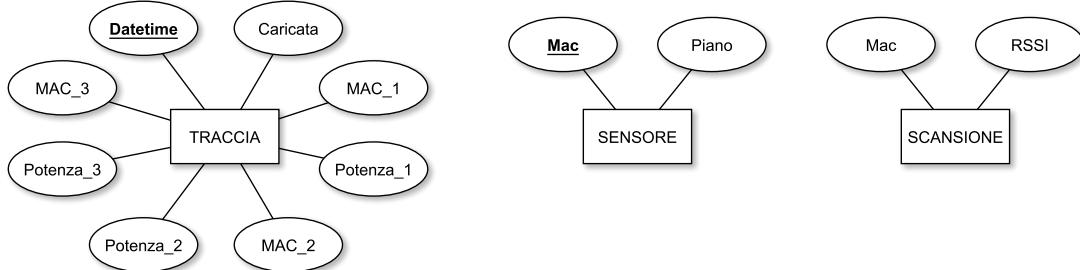
**Figura 5.3:** Casi d'uso Single Page Application

## 5.7 Applicazione Android

L'applicazione Android contiene due activity: l'activity di login e l'activity per avviare o fermare la rilevazione della risorsa. L'activity di login si interfaccia con l'utente chiedendo di inserire email e password per verificare la possibilità di accedere al sistema. L'activity di gestione della rilevazione si avvia automaticamente una volta che l'autenticazione è andata a buon fine. Le due activity e i casi d'uso ad esse associate, sono visibili in figura 5.4. Per la gestione dei dati si utilizza il database SQLite presente sul sistema operativo Android. Si realizza, quindi, il diagramma entità-relazione, rappresentato in figura 5.5.



**Figura 5.4:** Casi d'uso applicazione Android



**Figura 5.5:** Modello E-R del database dell'applicazione Android

Per lo sviluppo delle funzioni del protocollo MQTT si è utilizzata la libreria Eclipse Paho, libreria di riferimento per l'ecosistema MQTT, disponibile open-source in diversi linguaggi. Al suo avvio la seconda Activity effettua la sottoscrizione al topic in cui è presente la lista dei TAG BLE e, dato che o il localizzatore o il web service hanno pubblicato sullo stesso topic la lista dei TAG BLE disponibili con flag retain, allora si riceve subito la lista dei TAG BLE.

La lista dei TAG BLE serve per poter filtrare durante la scansione dei dispositivi Bluetooth solo quelli che effettivamente risultano essere registrati nella piattaforma, evitando in questo modo di memorizzare l'RSSI di altre periferiche che implementano l'interfaccia Bluetooth LE.

All'avvio della rilevazione, si avvia un service in background composto da due thread `TracciaUpload` e `TracciaBuffer`.

Il thread `TracciaBuffer` finchè la rilevazione è attiva esegue sequenzialmente le seguenti operazioni:

1. Avvia la scansione dei dispositivi BLE;
2. Attende un secondo;
3. Ferma la scansione dei dispositivi BLE;
4. Filtra le letture dei dispositivi BLE memorizzati, eliminando le letture dei dispositivi che non sono presenti nella lista dei TAG BLE;
5. Esegue la media se sono presenti più valori di RSSI riferiti allo stesso TAG BLE;
6. Aggiunge la traccia al Database, se esistono le letture di almeno 3 TAG BLE distinti, inserendo all'interno della traccia le letture dei 3 TAG BLE con indice RSSI più basso.

La scansione dei dispositivi BLE è realizzata mediante il codice [5.6](#).

```

1 BluetoothAdapter.startLeScan:
2   Callback{
3     db.addScansione();
4 }
```

**Codice 5.6:** Scansione dei dispositivi BLE

Il thread `TracciaUpload` avvia il thread `TracciaBuffer` e finchè la rilevazione è attiva esegue sequenzialmente le seguenti operazioni:

1. Attende 2 secondi;
2. Apre una connessione MQTT verso il broker;
3. Invia le tracce memorizzate nel database non ancora inviate;
4. Cancella le tracce inviate dal database;
5. Chiude la connessione MQTT.

# Capitolo 6

## Risultati sperimentali

I risultati ottenuti dal sistema, forniscono delle risposte significative circa la posizione della risorsa. Occorre precisare che la limitata complessità degli algoritmi utilizzati influenza, talvolta negativamente, sulla risposta di localizzazione in alcune particolari situazioni che verranno analizzate a parte.

Di seguito è riassunto l'esperimento effettuato al fine di valutare le performance del sistema.

### 6.1 Deployment del sistema

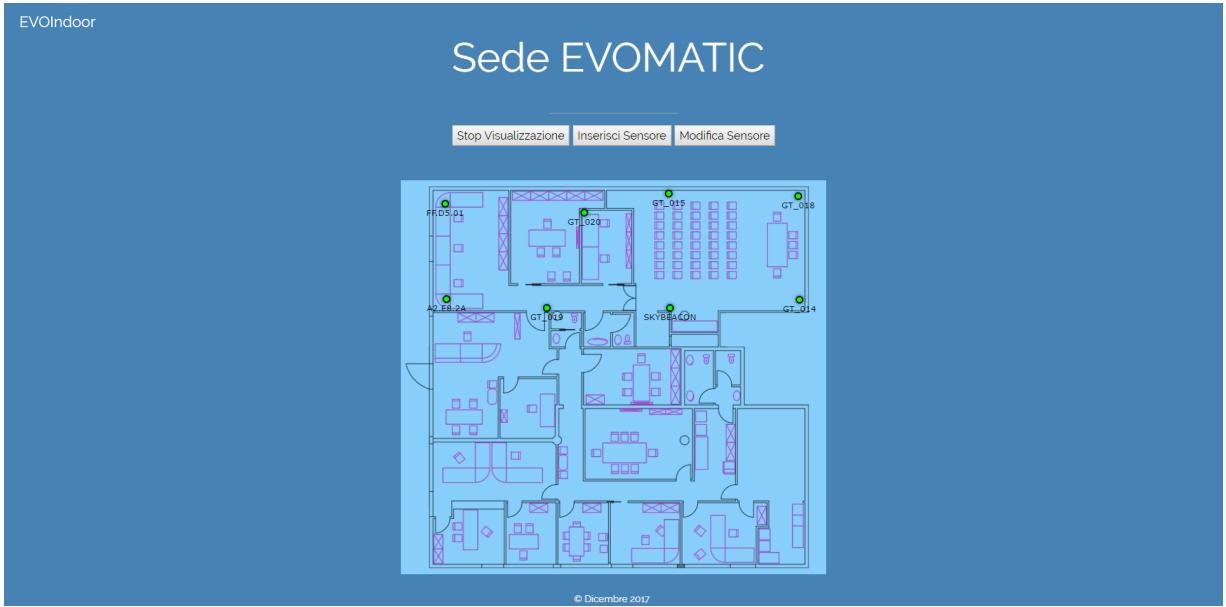
La prova è stata effettuata in un reale ambiente aziendale, di circa 300 metri quadrati.

Lo smartphone utilizzato per i test è un Alcatel A3 (TCL 5046Y), terminale Android di fascia bassa, mentre sono stati utilizzati da un minimo di 3 a un massimo di 8 TAG BLE.

Le varie connessioni implementate nell'applicazione Android puntano verso l'indirizzo IP del computer utilizzato per lo sviluppo. Nel computer utilizzato per lo sviluppo sono stati avviati nell'ordine: il broker mqtt, il localizzatore e il web service.

Si è avviata, accedendo dal browser, la Single Page Application e si è inserita la piantina dell'ambiente indoor che si desidera monitorare. Successivamente, si sono registrati e posizionati i vari TAG BLE all'interno dell'applicazione. Al termine della configurazione si ottiene un risultato come quello in figura 6.1. Successivamente si sono posizionati i TAG BLE, precedentemente registrati nell'ambiente indoor, coerentemente con quanto fatto nell'applicazione.

Si avvia l'applicazione sul dispositivo Android che, dopo aver effettuato l'autenticazione al broker MQTT, riceve e memorizza dal broker la lista dei TAG registrati. Si avvia quindi il tracciamento della risorsa. A questo punto, l'applicazione Android, invia periodicamente al broker MQTT, per ogni intervallo di tempo, i valori RSSI dei 3 TAG più vicini. Il broker MQTT inoltra i messaggi al localizzatore che si occupa di registrare il messaggio, di effettuare la conversione da RSSI a metri per ogni valore RSSI ricevuto e di ottenere la posizione tramite l'algoritmo di Trilaterazione.



**Figura 6.1:** Configurazione dell’ambiente indoor e posizionamento dei TAG BLE nell’interfaccia web

Dalla Single Page Application, avviando la visualizzazione delle risorse, si interroga periodicamente il web service per ottenere l’ultima posizione disponibile di tutte le risorse, visualizzandole nella piantina dell’ambiente indoor come in figura 6.2.

## 6.2 Valutazione delle performance

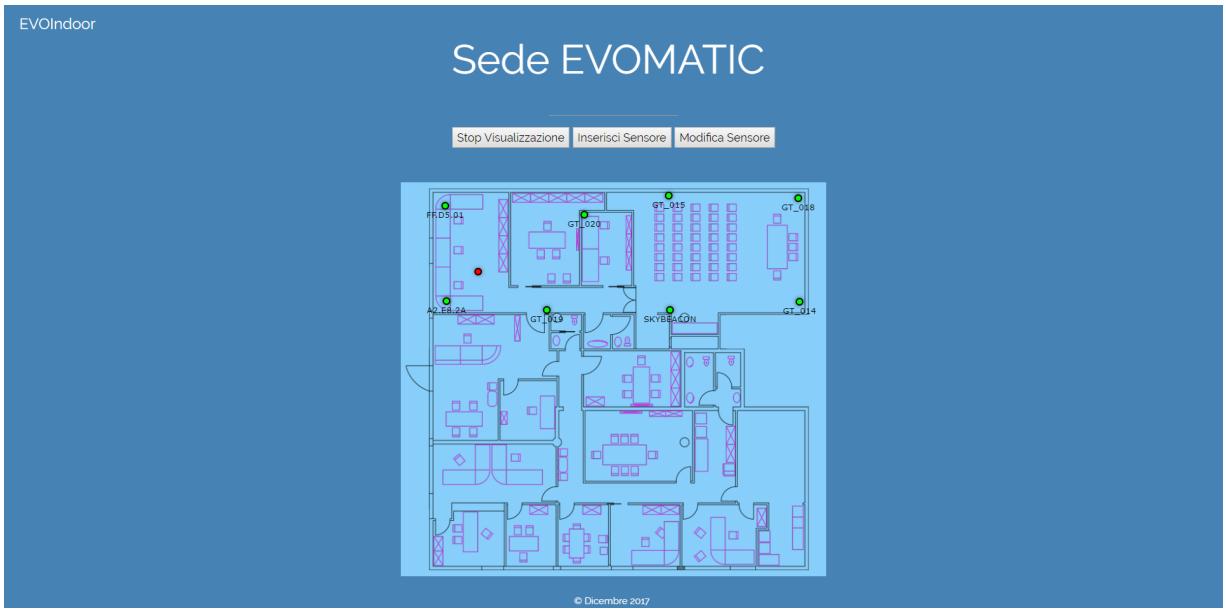
A corredo dell’analisi sperimentale è importante effettuare una stima delle performance del sistema, considerando come indici i parametri fondamentali. Di seguito il sistema è analizzato alla luce dei risultati ottenuti dai test eseguiti a run-time.

### 6.2.1 Accuratezza

L’accuratezza del sistema dipende da tanti fattori. I test effettuati dimostrano che, con gli algoritmi utilizzati, l’accuratezza della risposta di localizzazione è intorno a 1,5 metri, considerando praticamente sempre di localizzarsi all’interno della stanza giusta. Nei casi limite, ovvero fra una stanza e un’altra, la risposta “ondeggiava” fra i due TAG BLE di confine.

### 6.2.2 Complessità

La complessità del sistema risulta tutto sommato bassa, il costo computazionale è limitato ai soli confronti e a pochi calcoli semplici che rendono adatto il sistema anche a terminali più datati.



**Figura 6.2:** Visualizzazione della risorsa nell’interfaccia web

### 6.2.3 Robustezza

La robustezza sistema dipende dalla distribuzione dei TAG BLE in funzione dei distrubbi presenti e della distanza, quindi dalla complessità dell’ambiente che si vuole localizzare. Se in un abiente molto disturbato la distribuzione dei TAG BLE è bassa, allora la posizione della risorsa può essere errata a causa di una errata interpretazione dell’indicatore RSSI.

### 6.2.4 Scalabilità

Il sistema ha come vantaggio un’alta scalabilità che lo rende adatto sia in ambito domestico, che per l’utilizzo in grandi spazi nel quale risulta ancora più performante.

### 6.2.5 Costo

Come previsto, il costo del sistema è relativamente basso, l’infrastruttura è composta unicamente da TAG BLE e da smartphone, non richiede quindi sistemi dedicati. Altro costo è invece quello di messa in funzione e manutenzione del sistema; sicuramente più elevato, dati i tempi necessari alla costruzione della mappa.



# Conclusioni

Lo sviluppo delle comunicazioni wireless, dei dispositivi a basso costo, e in generale il facile accesso alle reti, aprono diverse possibilità di applicazione interessanti per il mercato.

La localizzazione indoor, implementata con TAG BLE, è una soluzione economica e dinamica. L'indice RSSI, come dice il nome, non è la potenza reale ricevuta dall'antenna del nodo ricevente, bensì un indice di questa. L'indice viene calcolato basandosi sul numero di errori di trasmissione rilevati nel pacchetto appena ricevuto, in quanto è giusto aspettarsi un numero di errori più elevato per potenze più basse.

Costruire un modello che caratterizzi al meglio il canale radio dell'ambiente in cui si va a realizzare la localizzazione è estremamente complesso. Sono troppe, infatti, le variabili che possono portare a cambiare significativamente il comportamento del sistema. Tra queste variabili ci sono riflessioni, rifrazioni e multipath del segnale radio assieme ad altri disturbi come oggetti in movimento. Di conseguenza l'indice RSSI non è sicuramente il metodo migliore per effettuare la localizzazione.

Tuttavia, con degli opportuni accorgimenti e con una buona modellazione, si possono ottenere discreti risultati. Dalle verifiche sperimentali è emerso che, per una buona localizzazione non sono necessari algoritmi troppo complessi. L'algoritmo della Trilaterazione, estremamente semplice dal punto di vista logico e computazionale, si è dimostrato accurato e preciso.

Il fatto di utilizzare più TAG BLE in ricezione per compensare eventuali disturbi rappresenta una soluzione molto efficace per aumentare la qualità e l'efficienza della localizzazione, tenuto conto infatti del costo ridotto di un TAG BLE.

Infine, per ottenere dei buoni risultati basandosi sull'indicatore RSSI è sconsigliabile avere scenari molto ampi con pochi TAG BLE. Nel complesso, nonostante la debolezza intrinseca dell'indice RSSI, si può comunque affermare che il sistema realizzato si comporti bene, ottenendo delle buone localizzazioni con errori di circa 1 metro, in ambiente indoor.

In futuro, il primo miglioramento potrebbe essere un raffinamento nel calcolo della distanza implementando un procedimento di calibratura automatica per quanto riguarda il calcolo dei parametri  $A$  e  $n$ , in modo da velocizzare la fase di caratterizzazione del canale ogni qualvolta si presenti la necessità di operare in un nuovo ambiente. Una procedura di questo genere dovrebbe richiedere all'utente di porre a varie distanze note un dato nodo mobile rispetto ad un nodo riferimento, posizionato nell'ambiente per cui si vogliono calcolare i parametri ambientali. I parametri  $A$  e  $n$ , caratteristici del canale, si possono quindi ricavare provando ad interpolare una funzione di tipo logaritmico utilizzando i valori di RSSI.

Il progetto realizzato ha consentito un approfondimento delle conoscenze dei pattern del linguaggio Java per lo sviluppo di applicazioni Android e per il multi-thread. Si sono rafforzate, inoltre, le conoscenze riguardo il linguaggio JavaScript, per lo sviluppo di applicazioni sia lato client che lato server.

I framework analizzati e utilizzati durante il periodo di tirocinio erano totalmente sconosciuti, mentre al termine del tirocinio si sono acquisite delle conoscenze basilari riguardo i vari framework. In particolar modo è stata raggiunta una buona padronanza nello sviluppo di applicazioni in Node.js, specialmente nello sviluppo di un'applicazione secondo i pattern tipici, anch'essi sconosciuti prima dell'inizio del tirocinio.

Sempre legato allo sviluppo, sono state acquisite delle competenze riguardo i tools offerti da Android Studio per il debug di applicazioni Android, e Visual Studio Code per il debug di applicazioni lato server, che prima non erano mai stati utilizzati.

Si sono rafforzate, inoltre, le competenze riguardo il linguaggio JavaScript, già conosciuto prima del tirocinio, ma del quale non si conosceva lo standard ES6.

Si sono acquisite delle nozioni riguardo alcuni aspetti del mondo aziendale, in particolar modo si è osservato come un'azienda valuta la possibilità di sviluppare nuove funzionalità stimandone benefici e costi.

Si è osservato, inoltre, come l'esperienza utente e i feedback forniti dai clienti influiscono sulla progettazione di un'interfaccia grafica, costringendo l'utilizzo di strategie per massimizzare le prestazioni al fine di soddisfare le diverse esigenze.

# Bibliografia

- [1] *AJAX Docs.* 2018. URL: <https://developer.mozilla.org/it/docs/Web/Guide/AJAX> (cit. a p. 18).
- [2] *Android Developer.* 2018. URL: <https://developer.android.com> (cit. a p. 15).
- [3] *Bluetooth.* 2018. URL: <https://it.wikipedia.org/wiki/Bluetooth> (cit. a p. 13).
- [4] Matteo Collina. *Mosca.* 2016. URL: <https://github.com/mcollina/mosca> (cit. a p. 21).
- [5] *CSS Docs.* 2018. URL: <https://developer.mozilla.org/en-US/docs/Web/CSS> (cit. a p. 18).
- [6] *DISK BEACONY – BLE Beacon Tag.* 2018. URL: <http://www.global-tag.com> (cit. a p. 14).
- [7] *HTML Docs.* 2018. URL: <https://developer.mozilla.org/it/docs/Web/HTML> (cit. a p. 18).
- [8] *HTTP Docs.* 2018. URL: <https://developer.mozilla.org/en-US/docs/Web/HTTP> (cit. a p. 16).
- [9] *Introducing JSON.* 2018. URL: <https://www.json.org> (cit. a p. 18).
- [10] *JavaScript Docs.* 2018. URL: <https://developer.mozilla.org/it/docs/Web/JavaScript> (cit. a p. 17).
- [11] *MQTT.* 2018. URL: <https://mqtt.org> (cit. a p. 15).
- [12] *NodeJS.* 2018. URL: <https://nodejs.org> (cit. a p. 16).
- [13] Davide Pozza. *MQTT: il protocollo che rende possibile l'Internet of Things.* 2015. URL: [https://www.slideshare.net/omnys\\_keynotes/mqtt-il-protocollo-che-rende-possibile-linternet-of-things](https://www.slideshare.net/omnys_keynotes/mqtt-il-protocollo-che-rende-possibile-linternet-of-things).
- [14] Marco Risi. «Confronto di tecniche di localizzazione per le WSN basate su RSSI». Tesi di laurea triennale. Università degli Studi di Pavia, 2009.
- [15] *Single page application.* 2017. URL: [https://it.wikipedia.org/wiki/Single-page\\_application](https://it.wikipedia.org/wiki/Single-page_application) (cit. a p. 17).
- [16] *Sistema di posizionamento globale.* 2018. URL: [https://it.wikipedia.org/wiki/Sistema\\_di\\_posizionamento\\_globale](https://it.wikipedia.org/wiki/Sistema_di_posizionamento_globale) (cit. a p. 3).

- [17] Nicholas Stephen. *Power Profiling: HTTPS Long Polling vs. MQTT with SSL, on Android*. 2012. URL: <http://stephendnicholas.com/posts/power-profiling-mqtt-vs-https> (cit. a p. 16).
- [18] Federico Stivani. «Studio e valutazione sperimentale di tecniche di radio fingerprinting per navigazione indoor con dispositivi mobili Android». Tesi di laurea triennale. Università degli Studi di Bologna, 2014.
- [19] *Structured Query Language*. 2017. URL: [https://it.wikipedia.org/wiki/Structured\\_Query\\_Language](https://it.wikipedia.org/wiki/Structured_Query_Language) (cit. a p. 17).