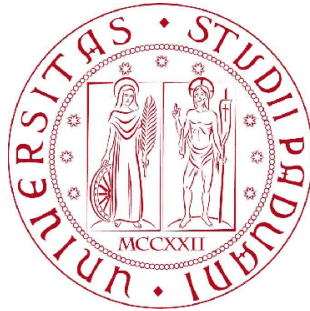


UNIVERSITÀ DI PADOVA

FACOLTÀ DI INGEGNERIA



Corso di Laurea in Ingegneria Informatica

REALIZZAZIONE DI UNA RETE MESH DI DISPOSITIVI BLUETOOTH PER IL PROXIMITY MARKETING

Relatore:
Federico Filira

Laureando:
Alessandro Zanella

Anno Accademico 2009/2010
28 settembre 2010

Indice

1	Introduzione	1
1.1	Specifiche dell'access point 3201 e access server A	2
2	Realizzazione dell'interfaccia HTML	7
2.1	Introduzione	7
2.2	Obiettivi	8
2.3	Piano di sviluppo	8
2.4	Utilizzo da parte dei genitori	9
2.5	Utilizzo da parte della segreteria	9
2.6	Realizzazione interfaccia HTML	10
2.6.1	Obiettivi	10
2.6.2	Dettagli implementativi	11
2.6.3	Gestione dei file nell'access point	11
2.6.4	Gestione delle regole di invio	12
2.6.5	Gestione degli indirizzi MAC	12
2.6.6	Gestione delle classi	13
2.6.7	Creazione di un report	14
2.6.8	Layout dell'interfaccia HTML	15
3	Realizzazione di un programma per la sincronizzazione	17
3.1	Introduzione	17
3.2	Obiettivi	18
3.3	Pianificazione del lavoro	19
3.4	Studio delle problematiche di sincronizzazione	19

3.4.1	Ricerca degli access point adiacenti	20
3.4.2	Strategie per bloccare l'invio dei messaggi	20
3.4.3	Dettagli implementativi	21
3.4.4	Struttura del programma	22
3.5	Realizzazione della rete mesh	22
3.5.1	Pianificazione del lavoro	22
3.5.2	Ricerca dei dispositivi adiacenti	23
3.5.3	Realizzazione di una procedura per la disposizione della rete mesh	25
3.5.4	Studio dell'algoritmo di routing	26
3.5.5	Studio sulle problematiche delle aree di interferenza degli access point	28
3.6	Realizzazione di un programma per la sincronizzazione	29
3.6.1	Pianificazione del lavoro	29
3.6.2	Invio e aggiornamento della lista dei dispositivi bloccati	29
3.6.3	Procedura d'avvio del programma	32
3.6.4	Comando tramite cellulare	33
3.7	Test	34
3.7.1	Pianificazione del lavoro	34
3.7.2	Rete utilizzata	34
3.7.3	Test di ricerca dei dispositivi	35
3.7.4	Test generale simulato	35
3.7.5	Test generale reale	37
4	Conclusioni	39
A	Manuale dell'interfaccia HTML per la bacheca scolastica	41
A.1	Installazione	41
A.1.1	Collegamento dispositivo	41
A.1.2	Installazione applicazione	41
A.1.3	Installazione interfaccia	42
A.2	Utilizzo interfaccia HTML	43

INDICE

A.2.1	Accesso Al programma	43
A.2.2	Gestione classi	43
A.2.3	Gestione file	44
A.2.4	Registrazione utenti	44
A.2.5	Report	45
B	Manuale del programma di sincronizzazione	47
B.1	Installazione	47
B.2	Disposizione della rete	48
B.3	Avviare il programma	49

Capitolo 1

Introduzione

Il Bluetooth è uno standard per le comunicazioni wireless a breve distanza. La prima versione dello standard fu rilasciata nel 1999 da Bluetooth Special Interest Group (SIG), una associazione composta da numerose società come Sony Ericsson (la prima società che iniziò a lavorare allo standard nel 1994), Sony, Toshiba, IBM, Intel ed altre che si sono aggiunte in momenti successivi. Lo standard Bluetooth è stato racchiuso nello standard IEEE 802.15, anche se sono presenti ancora delle differenze dalla versione della SIG Bluetooth. Il protocollo ha come obiettivo quello di creare un tipo di connessione wireless a basso costo ed a bassa potenza per le comunicazioni tra dispositivi mobili (cellulari, palmari, portatili). L'ultima versione Bluetooth è la 4.0 rilasciata nell'anno 2010, che presenta retro compatibilità con le versioni precedenti. Al giorno d'oggi un chip che implementa il Bluetooth ha un costo inferiore ai 4 euro, la sua implementazione in molti tipi di cellulari e di portatili è addirittura scontata. Già nel 2007 la SIG Bluetooth dichiarò che erano già stati venduti più di un miliardo di dispositivi Bluetooth e che venivano venduti più di 12 milioni di dispositivi alla settimana. Il numero di dispositivi attualmente è ovviamente aumentato e si stanno trovando sempre nuovi utilizzi in nuovi settori (medico, sportivo, ecc). Questa enorme diffusione ha determinato la nascita di un nuovo modo per rendere accessibili nuovi servizi. Questo nuovo metodo viene denominato proximity marketing

(marketing di prossimità), dove un determinato servizio viene proposto a tutte le persone aventi un dispositivo Bluetooth che sono nelle vicinanze di un access point Bluetooth. Per fare un esempio si potrebbe utilizzare questa particolare tecnica per realizzare una campagna di promozione per un negozio; un access point potrebbe inviare a tutti i cellulari con Bluetooth attivo un messaggio pubblicitario, sarebbe in pratica una versione automatica di un normale distribuzione di volantini.

Il compito della mia tesi è stato la realizzazione di due applicazioni per gli access point Bluetooth della BlueGiga da utilizzare per il proximity marketing. Il primo compito è stato lo sviluppo di un'interfaccia per la configurazione di un access point Bluetooth BlueGiga per la realizzazione di una bacheca scolastica. Il secondo compito, il nucleo centrale della tesi, è stata la realizzazione di un'applicazione per lo sviluppo di una rete mesh Bluetooth tra access point per la sincronizzazione tra i vari dispositivi. Nei prossimi due capitoli verranno spiegati in dettaglio gli obiettivi e la realizzazione dei due applicativi.

1.1 Specifiche dell'access point 3201 e access server A

Nella seguente sezione verranno elencate le diverse specifiche dei due tipi access point utilizzati: access point 3201 e l'access server A entrambi della Bluegiga. I due dispositivi possiedono caratteristiche hardware molto simili e montano il medesimo software, per questo motivo verrà spiegato in dettaglio, soprattutto il software e l'hardware dell'access point 3201 mentre per il secondo verranno solamente elencate le poche e, non molto influenti, differenze hardware.

I dati hardware più importanti dell'access point 3201 sono:

- Antenna Bluetooth della classe 1 e 2.1.

1.1. SPECIFICHE DELL'ACCESS POINT 3201 E ACCESS SERVER A

- CPU ARM 9 avente 200Mhz di frequenza massima e 16MB di memoria flash.
- Copertura massima di 100 metri, reale 70 metri.
- Profili supportati Bluetooth: SPP, OBEX OPP, OBEX FTP, PAN, LAN Access DI, HDP.
- Porta ethernet.
- Porta USB per poter aggiungere ulteriore memoria, mediante chiavetta USB.

Per quanto riguarda il software il dispositivo monta una versione embedded del sistema operativo Linux con il kernel 2.6.27. Il dispositivo viene impostato e comandato collegandolo ad un computer mediante connessione ethernet ed interfacciandosi ad esso in due modi diversi: mediante una interfaccia html o tramite una sessione ssh. Il dispositivo possiede tutti i comandi di una Shell, questo crea la possibilità di poter scrivere script molto utili nella creazione di semplici programmi. Il software più importante è l'**obexsender** che gestisce l'invio e la ricezione dei file tramite Bluetooth. Tramite questo software si possono impostare diverse regole di invio e ricezione di file come:

1. Invio dei file in un determinato giorno e in una determinata ora.
2. Eseguire programmi quando viene ricevuto un file contraddistinto da una particolare stringa.
3. Rispondere automaticamente quando viene ricevuto un particolare file.
4. La possibilità di inverntarsi delle proprie regole creando degli script.

Queste configurazioni possono essere facilmente impostate scrivendo, con una particolare sintassi, all'interno di un file (`/etc/obexsender.conf`). Un altro programma utile è il **finder** il quale una volta che si sono connessi diversi access point in una LAN li sincronizza automaticamente, ovvero i vari

dispositivi si inviano la lista dei dispositivi già serviti per non servirli ulteriormente, (se la regola dell'obexsender non lo prevede) inoltre condividere le diverse impostazioni. L'access point può essere programmato tramite il suo SDK, scrivendo programmi in C e C++.

L'access server A presenta solamente poche differenze nel hardware che è equipaggiato rispetto all'access point 3201, vengono di seguito elencate le differenze più importanti che hanno influenzato nello sviluppo dei diversi applicativi:

- Il dispositivo possiede 3 antenne Bluetooth per poter avere un numero maggiore di connessioni attive nello stesso momento. Questo comporta che ogni dispositivo possiede 3 indirizzi MAC diversi.
- Oltre alle porte ethernet e USB il dispositivo possiede anche una porta seriale che può essere usata per collegarlo ad un computer.



Figura 1.1: Immagine dell'access point server A

1.1. SPECIFICHE DELL'ACCESS POINT 3201 E ACCESS SERVER A



Figura 1.2: Immagine dell'access point A3021

Capitolo 2

Realizzazione dell'interfaccia HTML

2.1 Introduzione

Tramite gli access point della BlueGiga si è creata un'applicazione per la creazione di una bacheca scolastica. Per bacheca scolastica si intende un metodo per distribuire documenti di vario genere tramite Bluetooth verso i cellulari dei genitori degli studenti. Sarebbe la sostituzione delle normali bacheche appese ai muri all'entrata delle scuole, con la differenza che questa bacheca riconosce chi sta accedendo alle informazioni e quindi consente l'accesso solamente a determinate informazioni. In questo modo possono essere gestiti documenti che altrimenti mai potrebbero essere posizionati su una normale bacheca. Per fare un esempio un possibile scenario potrebbe essere questo: un genitore quando accompagna il figlio a scuola potrebbe scaricare le informazioni sul proprio cellulare di cui ha bisogno senza dover andare alla segreteria scolastica. In modo tale da rendere le informazioni maggiormente disponibili e più facilmente reperibili.

2.2 Obiettivi

Vengono di seguito spiegati in modo schematico i differenti obiettivi che si è proposti nella realizzazione dell'applicativo.

- Il sistema deve consentire l'accesso dei dati solamente ai cellulari precedentemente registrati.
- Gli utenti possono decidere quale file scaricare.
- Gli utenti possono visualizzare i file da poter scaricare.
- gli utenti sono divisi in classi (le classi scolastiche dei figli) e in base all'appartenza del proprio figlio a queste classi viene consentito il download dei file.
- L'utente visualizza solamente i file che non ha già scaricato.
- Il sistema deve supportare almeno 5 file.
- Il sistema deve registrare le informazioni dei diversi contatti con i cellulari. Le informazioni sono: ora, data, riuscita della connessione, chi ha effettuato l'accesso ai dati.

2.3 Piano di sviluppo

L'applicazione è stata scomposta in 3 diversi moduli:

1. Un programma per il cellulare per facilitare le operazioni per scaricare i documenti dall'access point.
2. Un applicazione lato access point che ha l'onere di realizzare la lista dei file che il cellulare può scaricare.
3. Un interfaccia HTML lato access point per facilitare la configurazione del programma `obexsender`.

2.4 Utilizzo da parte dei genitori

Vengono di seguito elencate le diverse operazioni che vengono compiute da un genitore per utilizzare la bacheca scolastica:

1. Il genitore deve andare in segreteria studenti per registrare il proprio cellulare e quindi avere il permesso di scaricare i dati alla classe del proprio figlio.
2. Il genitore avvicinandosi all'access point scarica automaticamente l'applicazione che consente il download dei file dalla bacheca.
3. Il genitore attraverso l'applicazione scaricata può visualizzare i file da scaricare.
4. Il genitore una volta scelto il file lo scarica utilizzando l'applicazione per cellulare.

2.5 Utilizzo da parte della segreteria

In questa sezione vengono elencate le diverse operazioni che può compiere un addetto alla segreteria (potrebbe anche essere un qualsiasi altro dipendente della scuola) per impostare la bacheca scolastica.

1. Creazione e cancellazione delle classi dentro le quali verranno inseriti i diversi file.
2. Caricamento e cancellazione dei file dentro l'access point. I file devono necessariamente essere di una particolare classe.
3. Registrazione degli indirizzi MAC dei cellulari dei genitori. L'applicazione deve fornire ai genitori un numero che sarà successivamente utilizzato per inviare un messaggio, tramite cellulare, all'access point e quindi registrare il cellulare. Inoltre l'operatore deve poter eliminare o inserire manualmente i singoli MAC.

4. Aggiornare i dati inseriti nell'anno precedente al nuovo anno in corso.
5. Visualizzazione dei log.

2.6 Realizzazione interfaccia HTML

Il mio compito è stato quello di creare un'interfaccia HTML che consentisse le operazioni indicate nella sezione precedente automatizzandole completamente, operazioni altrimenti complesse e che richiedono una discreta conoscenza del dispositivo. Inoltre sono state implementate le funzionalità quali la creazione delle regole di invio per il programma `obexsender` e la configurazione automatica dello stesso programma. Tramite l'interfaccia HTML sono state implementate le seguenti funzionalità:

1. Gestione dei file nell'access point (sono i file che il genitore avrà a disposizione).
2. Gestione delle regole di invio dei file nel file `obexsender.conf`.
3. Gestione delle classi.
4. Gestione degli indirizzi MAC per il riconoscimento dei dispositivi.
5. Creazione di un report che visualizza le varie operazioni di update dei file avvenuti.

Nelle sezioni successive verranno spiegati i singoli punti in modo dettagliato.

2.6.1 Obiettivi

Vengono di seguito elencati, in modo schematico, i diversi obiettivi voluti conseguire nello sviluppo della bacheca elettronica.

- Facilità di utilizzo, l'interfaccia deve essere semplice e intuitiva.
- Deve poter consentire l'update di file dentro l'access point.

2.6. REALIZZAZIONE INTERFACCIA HTML

- La configurazione dell'`obxsender` deve essere automatica ogni volta che viene inserito un nuovo file.
- I MAC dei cellulari e i file vengono divisi tra le varie classi.
- Deve essere realizzata una semplice procedura per la registrazione dei cellulari dei genitori.
- Deve essere realizzata una procedura che consente in modo agevole di aggiornare i dati quando inizia un nuovo anno.

2.6.2 Dettagli implementativi

La realizzazione dell'interfaccia HTML ha necessitato la creazione di pagine HTML dinamiche e che eseguissero delle operazioni sui file. Per gli access point 3021 e gli access Server A possono essere scritti solamente programmi in C, C++ e script Shell; é possibile anche realizzare script in Perl installando software aggiuntivo a quello dato di default nel dispositivo. In base a questi fattori si è deciso di utilizzare lo standard CGI (Common Gateway Interface) per la creazione dinamica delle pagine HTML. I diversi script CGI sono stati realizzati come script Shell. Questa scelta è stata dettata da diversi fattori come la facilità e la velocità di scrittura del programma rispetto ai programmi come C e C++. Si è deciso, invece, di non utilizzare il Perl per non dover installare software aggiuntivo all'access point che verrebbe utilizzato solamente per un'applicazione.

2.6.3 Gestione dei file nell'access point

Tramite un'interfaccia html si possono inviare o cancellare file nella cartella `/usr/local/obxseder/files/nomeClasse`, dove `nomeClasse` è la classe a cui appartiene il file. Durante l'inserimento il file viene rinominato con la seguente sintassi:

```
classe.nomeFile.estensione
```

Comunque l'utente non vedrà mai il nome modificato, ma solamente quello inserito.

2.6.4 Gestione delle regole di invio

Viene gestita la scrittura e la cancellazione delle regole di invio per il programma `obexsender`. In modo automatico ogni volta che viene inserito un file viene automaticamente inserito alla fine del file `/etc/obexsender.conf` la regole di invio scritte nella seguente formula:

```
reply {
    keyname NomeFile
    file nomeFile
}
```

La cancellazione di un file comporta automaticamente la cancellazione della regole di invio. In questo modo l'utente non deve modificare manualmente il file `obexsender.conf`. Questa regola serve per inviare a un cellulare un determinato file quando l'access point riceve un messaggio con la stringa identificativa `NomeFile`. Il messaggio viene inviato automaticamente quando viene scelto il file da scaricare tramite l'applicazione per cellulare.

2.6.5 Gestione degli indirizzi MAC

Vengono visualizzati tutti gli indirizzi mac dei dispositivi dei genitori di una determinata classe inoltre sono possibili le operazioni di cancellazione, generazione codice identificativo e trasferimento.

Visualizzazione della lista dei MAC viene eseguita leggendo il file `mac-Dettagli.txt`, nel quale. quando viene inviato il messaggio di conferma di autenticazione del mac, vengono salvati tutti i dati di un mac: nome e cognome possessore, data inserimento, numero mac e classe. La sintassi del file è la seguente `classe/indirizzomac/nome/cognome/data` la data è nella forma `giorno.mese.anno`, dati scritti in forma numerica.

Cancellazione indirizzo MAC cancella un indirizzo mac dal file `classi.txt` e dal file `macDettagli.txt`.

Generazione codice identificativo , dopo aver inserito nome e cognome del proprietario del dispositivo, genera un numero a quattro cifre, che viene incrementato dopo ogni inserimento. Il numero viene salvato nel file `FileCodici.txt` con la sintassi di ogni linea è `codice/nome/-cognome/classe` Questa operazione serve per configurare il riconoscimento automatico del MAC del cellulare. L'utente deve inviare il codice identificativo all'access point per salvare il MAC del proprio cellulare e quindi effettuare l'operazione di registrazione.

trasferimento sposta l'indirizzo mac nella classe successiva o nella classe precedente. Occorre che la classe di partenza abbia il nome nella forma `LetteraNumero` (es. A1, B3, C4) ed inoltre che esista la classe di destinazione successiva o precedente (es. classe di partenza A1 trasferimento alla classe successiva deve esistere la classe A2). Questa operazione serve per gestire la presenza di possibili studenti ripetenti.

2.6.6 Gestione delle classi

La gestione delle classi permette la creazione, l'eliminazione e il trasferimento di tutti i dati all'anno successivo.

- La creazione di una classe crea una cartella con il nome inserito dall'utente in `/usr/local/obexsender/files/`. Viene concesso l'inserimento di qualsiasi nome ma affinché le funzioni di trasferimento dei mac e delle classi siano possibili bisogna inserire il nome della classe nella forma `LetteraNumero` (es. A1, B2, D4).
- L'operazione di cancellazione cancella la cartella della classe con anche tutti i suoi file all'interno, tutte le regole di replay di una determinata classe dal `/etc/obexsender.conf` e cancella tutti le informazioni dei mac dai file `FileCodici.txt`, `macDettagli.txt` e `classi.txt`.

- L'operazione di trasferimento sposta la classe all'anno successivo, in pratica la classe A1 viene trasferita alla classe A2. Per eseguire questa operazione il nome della classe deve essere nella forma letteraNumero, inoltre la classe di destinazione non deve essere già presente. L'operazione rinomina la cartella della classe incrementando di uno il numero del nome (es. A1 diventa A2). Tutti i file della classe vengono rinominati per essere della nuova classe (se A1.file.estensione diventa A2.file.estensione). Vengono aggiornati i file `FileCodice.txt`, `macDettagli.txt` e `classi.txt`, in modo che i mac appartengano alla nuova classe. Le regole di reply presenti in un determinato file vengono aggiornate.

2.6.7 Creazione di un report

Viene visualizzato un report che evidenzia tutti i file presenti nelle diverse classi. Per ogni file vengono visualizzati il numero di invii avvenuti con successo e il numero di invii falliti. Inoltre è possibile sapere tutti i dettagli dei diversi invii ovvero sapere data e ora di invio e l'utente che l'ha effettuata. Ciò viene realizzato controllando il file `obexsender.log`, quindi se il file deve essere scritto in memoria usando la specifica opzione del `obesender.conf` inoltre se il file viene cancellato o modificato i dati dipendendo da esso vengono anch'essi modificati.

2.6.8 Layout dell'interfaccia HTML



Figura 2.1: Schermata html per la gestione dei file



Figura 2.2: Schermata html per la gestione delle classi

Classi

- A1
- A2
- B2
- B3
- D1

Modelli cellulari, classe A1

Generazione codice
Nome: Nome
Cognome: Cognome
Numero generato **0013**

Mac	Nome	Cognome	Giorno.Mese.Anno	Operazione	Trasferisci
44:56:87:a2:3b:12	A	B	17.08.2010	Elimina	Successivo / Precedente
89:90:00:ab:a2:11	C	D	17.08.2010	Elimina	Successivo / Precedente

Inserimento

Nome

Cognome

Gestione manuale

[Indietro](#)

Figura 2.3: Schermata html per la gestione degli indirizzi MAC

Capitolo 3

Realizzazione di un programma per la sincronizzazione

3.1 Introduzione

Si è voluto realizzare un programma che sincronizzi diversi access point tramite l'utilizzo di una connessione Bluetooth. Per sincronizzazione si intende quel processo in cui i diversi access point si inviano tra di loro le liste degli indirizzi MAC dei dispositivi che hanno già ricevuto quel file che l'access point deve inviare. Quando l'access point viene impostato per l'invio di file in modalità broadcast l'access point memorizza i MAC di tutti i dispositivi verso i quali è già stato inviato il file. Questo è necessario per evitare che un dispositivo mobile riceva più e più volte lo stesso file. Tramite l'invio delle liste dei dispositivi già serviti e il blocco dell'invio verso questi si possono utilizzare più access point, coprendo quindi un'area maggiore, ma funzionando come fosse un unico dispositivo. Questa funzionalità è già presente nel software dato in dotazione con l'access point ma necessita di un collegamento ethernet. Col metodo sviluppato in questa tesi, invece, non è necessario alcun collegamento basta che i due access point siano uno dentro nel raggio di azione del Bluetooth dell'altro.

3.2 Obiettivi

Di seguito vengono elencati gli obiettivi che si vogliono conseguire nella realizzazione del software per la sincronizzazione degli access point.

- Sviluppare un programma per la ricerca degli access point vicini e funzionanti per creare una rete mesh.
- L'access point deve inviare la lista dei MAC dei cellulari, verso i quali ha già inviato il file, a tutti gli AP presenti nelle vicinanze.
- L'access point deve aggiornare la sua lista dei dispositivi bloccati, cioè quei dispositivi cui è impedito l'invio di file, tramite i MAC inviati dagli altri access point.
- L'aggiornamento della lista dei dispositivi bloccati deve essere abbastanza veloce da impedire che un utente, muovendosi tra i diversi access point, riceva più volte lo stesso file.
- Il programma di sincronizzazione deve essere semplice.
- Il programma deve essere comandato inviando semplici messaggi di testo con il cellulare.

3.3 Pianificazione del lavoro

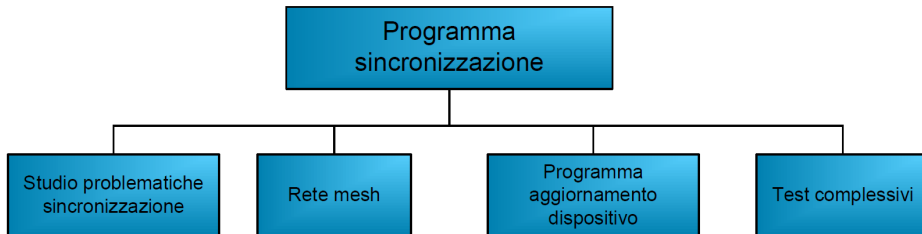


Figura 3.1: WBS generale del lavoro svolto

Come evidenziato in figura 3.1 il lavoro è stato suddiviso in questo modo perchè rappresenta la reale struttura logica del problema e perchè si sono voluti realizzare i due programmi (programma per la realizzazione della rete mesh e sincronizzazione dei dispositivi) indipendenti uno dall'altro.

Lo studio dei problemi di sincronizzazione è stato necessario a causa della scarsa documentazione fornita assieme al dispositivo e dei dubbi sorti sull'effettiva possibilità di realizzare il progetto. La fase di test è stata, invece, separata dal resto dello sviluppo e condotta a posteriori per le effettive difficoltà nella disposizione di diversi access point. Durante lo sviluppo dei singoli programmi si è lavorato solamente con un access point per verificare che il programma fosse corretto. Nella fase di test, una operazione lunga e complessa, invece, dei programmi sono stati testati costruendo una rete di access point reali.

I capitoli successivi spiegheranno in dettaglio il lavoro e i risultati ottenuti.

3.4 Studio delle problematiche di sincronizzazione

Nella prima fase dello sviluppo sono state studiate e risolte le due principali problematiche dell'intero progetto: come trovare i dispositivi della BlueGiga adiacenti a un'access point e come impedire al programma `obexsender`, conoscendo il MAC del dispositivo, l'invio di messaggi in broadcast.

Un'altra scelta effettuata in questa fase è stato decidere come implementare il programma, con quale linguaggio scriverlo e in che modo gestire l'invio e la ricezione dei diversi messaggi. L'intera fase di sviluppo si è conclusa progettando la struttura generale del programma.

3.4.1 Ricerca degli access point adiacenti

Quando un access point viene acceso deve rilevare gli access point presenti nelle vicinanze per decidere verso quali dispositivi deve inviare la sua lista di dispositivi bloccati. Questa operazione può essere facilmente eseguita tramite il comando `inquiry` che rileva tutti i dispositivi mobili con il Bluetooth attivato nelle vicinanze. Sfortunatamente questo comando non funziona correttamente perchè non sempre riesce a rilevare tutti i dispositivi funzionanti. Per risolvere questo problema basta rieseguire l'operazione di ricerca; per cui nella stesura del programma si è deciso di rieseguire la procedura 3 volte scandendo la ricerca ogni 30 secondi. Sfortunatamente questa procedura non è sufficiente, perchè può capitare che sia impossibile inviare file a un dispositivo Bluetooth perfettamente rilevato con il comando `inquiry`. Quindi l'unico metodo per avere la certezza che sia possibile inviare un file a un dispositivo, è inviarlo ed aspettare un messaggio di risposta che certifichi che il messaggio sia stato inviato con successo.

3.4.2 Strategie per bloccare l'invio dei messaggi

Il problema nasce dal fatto che non è stata realizzata dagli sviluppatori una funzionalità specifica per bloccare l'invio di un file verso un determinato indirizzo MAC che non prevedeva il reset dell'`obexsender` (il programma che gestisce l'invio dei file tramite Bluetooth). Il problema è stato risolto usando le funzioni per il ripristino dello stato dell'access point dopo un'interruzione di corrente. Queste funzioni permettono di salvare in un file i MAC dei dispositivi già serviti in un file e, successivamente, di utilizzare questo file per ripristinare lo stato precedente, caricandolo nella memoria del dispositi-

3.4. STUDIO DELLE PROBLEMATICHE DI SINCRONIZZAZIONE

tivo e bloccare l'invio dei file. Questo metodo, perfettamente funzionante, presenta però un grave difetto perché, se l'operazione viene eseguita con una frequenza troppo elevata, c'è il rischio di bruciare fisicamente la memoria del dispositivo. Per evitare ciò si è programmato l'access point in modo che il dispositivo esegua l'operazione di upload solamente una volta ogni tre minuti, un intervallo sufficientemente lungo pensando che gli access point possono essere posti a una distanza di oltre 50 metri e che un dispositivo mobile ci mette un tempo superiore ai 20 secondi per iniziare una fase di comunicazione. Il valore definitivo dell'intervallo, comunque, dovrà essere impostato durante i test conclusivi dell'intero progetto nei quali si potrà capire meglio il comportamento dell'intero programma.

3.4.3 Dettagli implementativi

Per la realizzazione del programma si è sfruttato il programma **obexsender** che gestisce l'invio e la ricezione di messaggi verso altri dispositivi. Questo, inoltre, permette di eseguire un programma quando viene ricevuto un messaggio contenente una particolare stringa identificativa. Attraverso **obexsender** è bastato realizzare i programmi che creassero i messaggi da inviare tra i vari dispositivi e che gestissero le operazioni di salvataggio e di caricamento dei dispositivi già bloccati. Questa particolare scelta implementativa è stata dettata dalla facilità e dalla semplicità nell'impostare ed utilizzare il programma **obexsender**. Non utilizzandolo sarebbe stato necessario scriverne uno dedicato solamente per la gestione dell'invio e la ricezione dei file per la sincronizzazione dei dispositivi. Per fare questo è necessario utilizzare i comandi del protocollo **obex**, compito non semplice e neppure veloce, che necessita di una discreta conoscenza della tecnologia Bluetooth. La scelta fatta non è sicuramente la migliore dal punto di vista dell'efficienza, ma i test effettuati hanno dimostrato che il programma svolge il suo compito senza causare particolari rallentamenti dell'access point che lo esegue. I diversi sotto-programmi sono stati realizzati come script in **Shell** perché non

sono eseguite operazioni computazionalmente pesanti e si lavora solamente su file di testo.

3.4.4 Struttura del programma

Per facilitare la progettazione e soprattutto la fase di test, si è deciso di scomporre il programma principale in due sotto-programmi tra loro indipendenti: `startSincro` e `sincroBlue`. Il primo programma ha il compito di trovare tutti gli access point presenti nelle vicinanze. I MAC degli access point rilevati vengono memorizzati nel file `/usr/local/obxsender/bin/-dispmac.txt` che viene utilizzato nelle fasi successive. Il secondo sotto-programma invia a tutti gli access point adiacenti la lista dei MAC dei dispositivi mobili già serviti. Questa suddivisione è stata decisa anche per consentire all'utente, se avesse già eseguito il programma `startSincro` e la disposizione degli access point non fosse cambiata, di eseguire solamente il programma `sincroBlue`.

3.5 Realizzazione della rete mesh

3.5.1 Pianificazione del lavoro

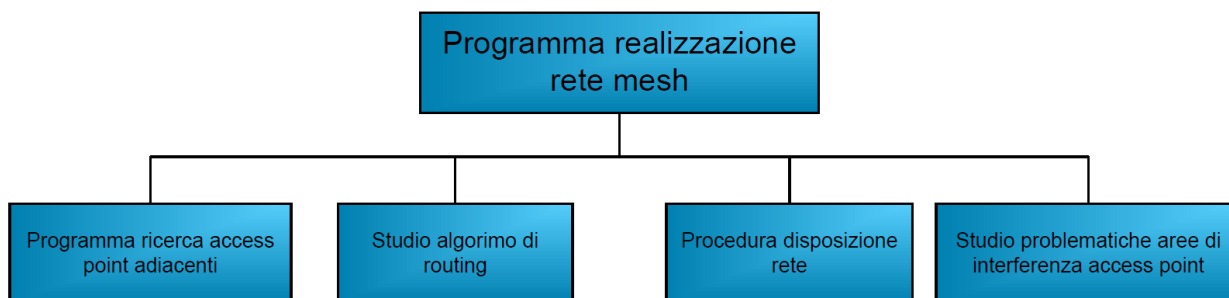


Figura 3.2: WBS della realizzazione della rete mesh

In 3.2 viene mostrato come è stato strutturato il lavoro. Nelle successive sezioni saranno spiegate dettagliatamente le diverse parti del programma.

3.5.2 Ricerca dei dispositivi adiacenti

In questa sezione viene mostrata la struttura del programma che ricerca i dispositivi adiacenti BlueGiga attivi nelle vicinanze. Nella spiegazione del modulo si farà riferimento sempre a due access point diversi: A e B. A sarà il dispositivo che inizia la fase di ricerca e quindi trova B, ovviamente alla fine della procedura anche B avrà trovato A. Per migliorare la comprensione viene inserito in 3.3 l'activity diagram dell'algoritmo di ricerca .

Il grafico rappresenta le operazioni principali che vengono svolte dai tre script Shell `ricercaDisp`, `rispondiRic` e `confermaRisp`, i quali sono installati in ogni access point e che poi verranno eseguiti in caso di necessità. La procedura di ricerca viene attivata solamente in un access point che inizia la ricerca di tutti i dispositivi attivi presenti nelle vicinanze della BlueGiga. I dispositivi della BlueGiga sono identificati dalle prime tre coppie di numeri dell'indirizzo MAC, 00:07:80, quindi l'identificazione di tali dispositivi è particolarmente semplice. L'access point che ha effettuato la ricerca (access point A), invia a tutti gli access point trovati un messaggio, se questo dispositivo non è già presente nella lista dei dispositivi già trovati e quindi non è presente nel file `/usr/local/obexsender/bin/dispmac.txt`. Il messaggio inviato è un file di testo con nome `fileEcho.txt` con le seguenti informazioni:

```
EchoRicercaAp
MACA: indirizzo dell 'access point A
Ora e date in cui è stato creato il messaggio
```

`EchoRicercaAp` è la stringa di riferimento che serve al programma `obexsender` per identificare che il messaggio è stato inviato da un altro access point e che quindi inizia l'esecuzione dello script `rispondiRic`. Lo script `rispondiRic`, eseguito nello script B dopo aver memorizzato il MAC di A, se non già presente invia ad A un messaggio di conferma. Il messaggio inviato da B è chiamato `fileRisp.txt` e contiene le seguenti informazioni:

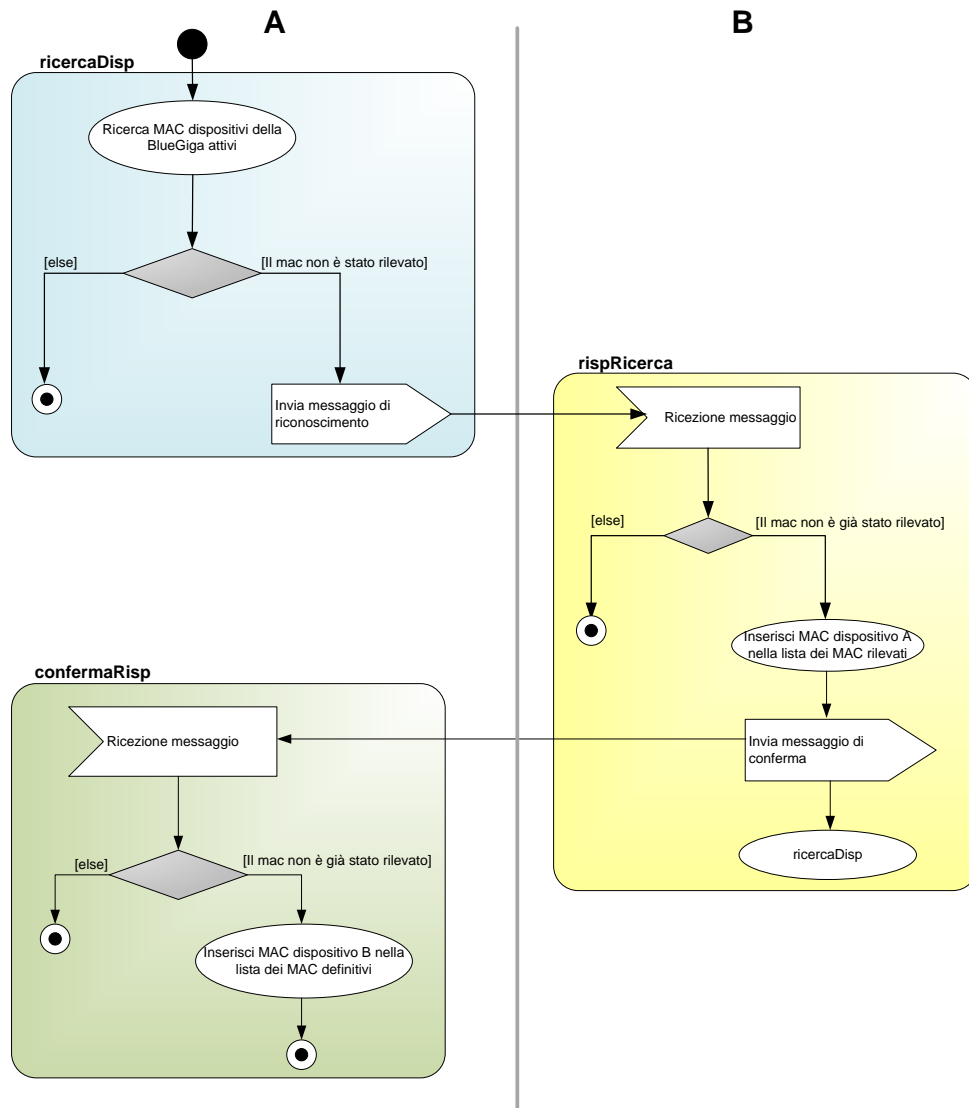


Figura 3.3: Activity diagram della procedura di ricerca dei dispositivi

3.5. REALIZZAZIONE DELLA RETE MESH

```
EchoRispostaRicercaAP
MACA: indirizzo MAC del dispositivo A
MACB: indirizzo MAC del dispositivo B
"Ora e data in cui è stato creato il messaggio"
```

`EchoRispostaRicercaAP` è la stringa di riferimento per il programma `obxsender` per identificare, anche in questo caso, il tipo di messaggio ricevuto. L'access point B, dopo aver inviato il messaggio di risposta, inizia esso stesso a cercare gli access point attivi nelle vicinanze a meno che, ovviamente, questa procedura non sia già in esecuzione. L'access point A, quando riceve il messaggio di conferma inviato da B, può finalmente riconoscerlo come access point adiacente, funzionante per poter essere utilizzato per creare la rete mesh di sincronizzazione.

La procedura descritta non sempre funziona in modo corretto. Un access point (come spiegato nella sezione delle problematiche affrontate) che esegue la ricerca di altri dispositivi attivi nelle vicinanze ha un'alta probabilità di non trovare tutti gli access point perfettamente funzionanti. Questo problema è stato risolto iterando la procedura di ricerca tre volte e aspettando, dopo ogni iterazione 30 secondi. Durante i test effettuati utilizzando questo metodo, la ricerca ha sempre trovato tutti gli access point veramente attivi. L'unica possibilità che un dispositivo non venga rilevato è che tutti i dispositivi adiacenti non lo vedano in tre diverse ricerche, quindi una probabilità davvero minima.

3.5.3 Realizzazione di una procedura per la disposizione della rete mesh

Quando si vuole realizzare una rete mesh, il problema principale per l'utente è trovare la posizione dove installare gli access point per coprire l'area maggiore e nello stesso tempo fare in modo che i diversi access point possano comunicare fra loro. Per risolvere questo problema basterebbe conoscere in modo preciso con quali access point possa comunicare un dispositivo. Per

fare questo si è utilizzato il programma descritto nella sezione precedente, modificandolo in modo tale da renderlo comandabile tramite cellulare. La procedura realizzata funziona in questo modo: l'utente invia un comando contenente la stringa `startRicerca` tramite Bluetooth verso un access point. Quando l'access point riceve il messaggio, esegue il programma `sincroFind` ed invia all'utente la lista dei codice hardware dei dispositivi che riesce a vedere con la potenza in dB del segnale. In base a questi dati l'utente può decidere se modificare la posizione degli access point. Quando viene eseguito il programma `sincroFind` vengono anche salvati i MAC dei dispositivi trovati quindi, eseguendo `sincroBlue` su ognuno dei dispositivi, si riesce a posizionarli correttamente. Nel caso che un access point venga spostato la procedura deve essere rieseguita.

Per facilitare l'impostazione delle rete è stato anche creato un programma che trova i dispositivi adiacenti per tutti gli access point della rete. L'utente manda un'unico messaggio verso un access point, il quale inizia la fase di ricerca dei dispositivi. Quando un access point riceve il messaggio riesegue a sua volta la procedura di ricerca e così via, fino a quando viene identificata tutta la rete. Il programma implementato si chiama `sincroFindDisp` e, per attivarlo, l'utente deve inviare un messaggio con la stringa `startFindDisp`.

3.5.4 Studio dell'algoritmo di routing

La fase successiva è la creazione di una politica di invio dei messaggi cioè, in linguaggio più tecnico, la ricerca dell'algoritmo di routing più idoneo. La scelta della migliore politica è stata fatta tra l'algoritmo di spanning tree e una semplice politica di flooding (invio tutto a tutti). Non sono stati considerati altri algoritmi perchè è impossibile trovare un algoritmo migliore dello spanning tree, essendo l'algoritmo ottimale, e perchè è impossibile trovare un algoritmo più semplice di un flooding, visto che non esegue nessun controllo prima dell'invio dei dati. Dopo una fase di studio preliminare si è deciso di utilizzare l'algoritmo di flooding, i motivi sono i seguenti:

3.5. REALIZZAZIONE DELLA RETE MESH

- La rete mesh arriverà difficilmente a comprendere più di 4 o 5 dispositivi.
- La semplicità nella realizzazione dell'algoritmo.
- L'algoritmo, essendo rindondante, gestisce meglio i guasti i.e. un'interruzione di corrente.
- C'è la sicurezza che gli access point ricevano la lista dei dispositivi bloccati dai dispositivi vicini al più ogni tre minuti. Tramite la creazione di un spanning tree, invece, c'è la possibilità che dispositivi adiacenti si inviino messaggi con tempi superiori ai tre minuti.

L'algoritmo di flooding, comunque, presenta il problema che un access point riceve più volte lo stesso messaggio da dispositivi diversi. Nella figura 3.4 viene mostrata la situazione in cui B riceve lo stesso messaggio sia da A e, dopo tre minuti, da C. Per risolvere questo problema si è deciso di migliorare

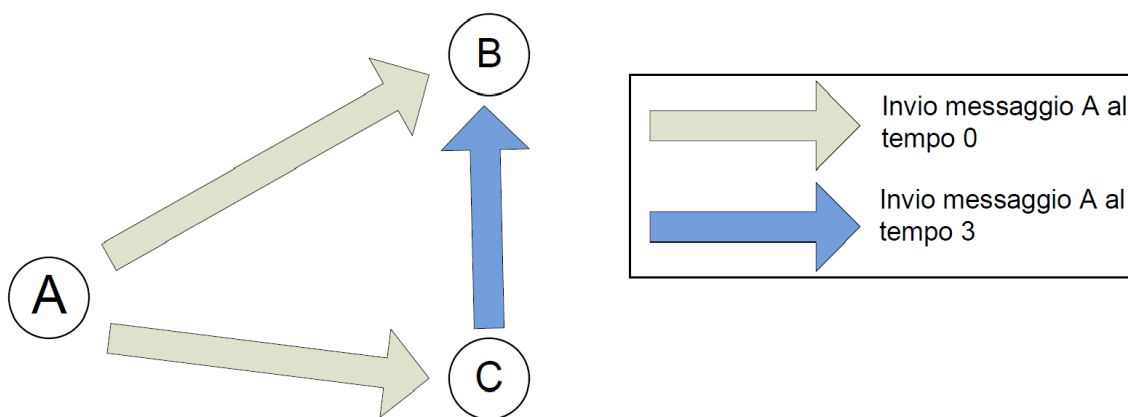


Figura 3.4: Situazione problematica con 3 nodi

l'algoritmo facendo in modo che tutti i messaggi inviati da un access point contengano anche la lista dei destinatari del messaggio. In questo modo non viene evitato il problema precedente, ma solo ritardato. Nella figura ?? si vede che D riceve lo stesso messaggio, inviato originariamente da A, sia da B che da C. Si è comunque scelto di usare questa soluzione perchè le reti

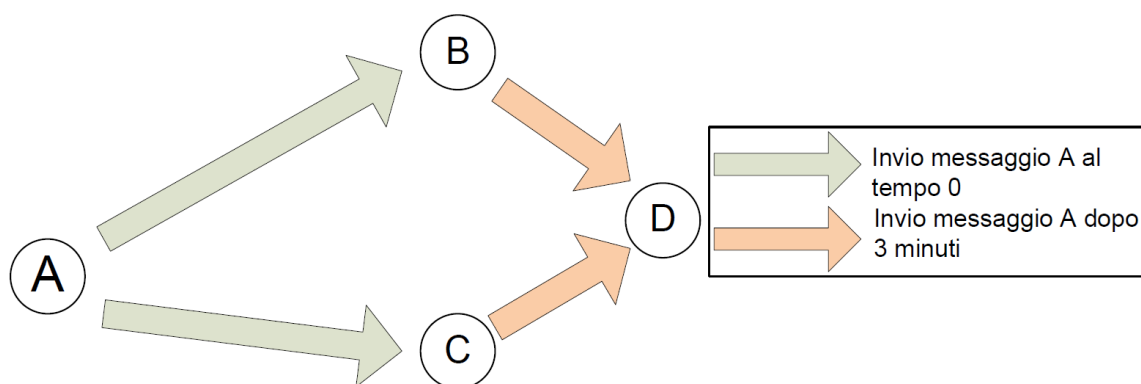


Figura 3.5: Situazione problematica con 3 nodi

ipotizzate sono molto piccole e perchè, la rete rappresentata in figura, non è reale. La rete mesh deve coprire la più vasta area possibile, quindi è molto difficile che un utente posizioni gli access point come mostrato in figura ???. Si consideri, inoltre, che i messaggi inviati dagli access point ogni 3 minuti sono molto leggeri, contengono infatti solamente qualche decina di indirizzi MAC, quindi, anche nel caso sfortunato che vengano inviati dei doppioni, la situazione non è molto problematica.

3.5.5 Studio sulle problematiche delle aree di interferenza degli access point

Per funzionare correttamente la rete mesh non deve avere più di un access point che riesca a comunicare con un cellulare. Se questo avviene, un dispositivo potrebbe ricevere contemporaneamente due messaggi. Per evitare questo problema gli access point devono essere posti a una distanza maggiore della copertura Bluetooth dei cellulari (circa 25m), ma non così elevata da impedire la comunicazione tra di loro. Per mancanza di tempo questi test non sono stati portati completamente a termine e non è stata realizzata una procedura per risolvere questo problema.

3.6. REALIZZAZIONE DI UN PROGRAMMA PER LA SINCRONIZZAZIONE



Figura 3.6: Situazione di una rete mesh funzionante

3.6 Realizzazione di un programma per la sincronizzazione

3.6.1 Pianificazione del lavoro

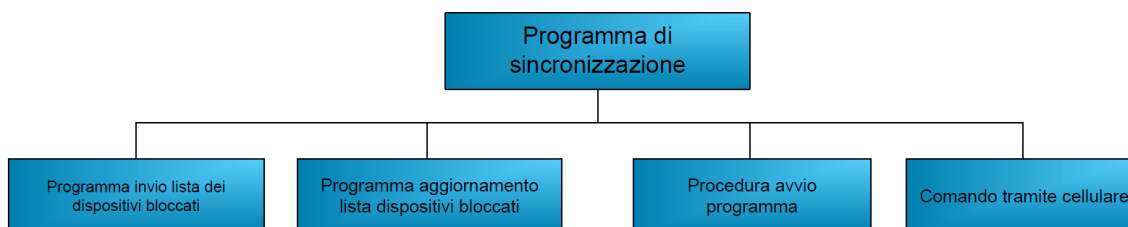


Figura 3.7: WBS realizzazione procedura di sincronizzazione

3.6.2 Invio e aggiornamento della lista dei dispositivi bloccati

Viene di seguito descritto l'algoritmo, implementato nello script `sincro-Blue`, utilizzato per aggiornare ed inviare la lista dei MAC dei dispositivi mobili. Il programma utilizza tre file:

- `/var/lib/obexsender/blocklist.dump` questo è il file dove vengono salvati i MAC dei dispositivi già serviti dal programma `obexsender`. Questo file viene aggiornato lanciando il comando `service obexsender saveblock` e può essere caricato in memoria per impedire l'invio dei file verso i dispositivi mobili tramite il comando `service obexsender loadblock`.
- `/var/lib/obexsender/oldblocklist` nel quale vengono salvati tutti i MAC dei dispositivi mobili già inviati verso gli altri access point.
- `/var/lib/obexsender/newblocklist` in questo file è contenuta la lista dei nuovi MAC dei dispositivi mobili serviti dall'access point. La lista viene creata trovando i MAC presenti nel file `blocklist.dump`, ma non presenti nel file `oldblocklist`.

Per facilitare la comprensione viene inserito l'activity diagram dello script `sincroBlue`, ???. Il programma `sincroBlue` esegue la seguente procedura: il branch con la condizione `lista MAC già inviata verso l'access point` sta a significare che viene controllato il non inserimento, nella lista dei MAC da inviare, anche i MAC inviati dal mittente che ha mandato la stessa lista e che non venga neanche inviata verso access point di cui si sa che hanno già ricevuto tale lista. Quest'ultima condizione viene eseguita controllando ogni singolo messaggio ricevuto da un altro access point. La sintassi di un messaggio contenente la lista dei MAC è la seguente:

```
MAC_DESTINATARI
MDnn:nn:nn:nn:nn:nn:nn
MDnn:nn:nn:nn:nn:nn:nn
MDnn:nn:nn:nn:nn:nn:nn
MAC_BLOCCATI
mm:mm:mm:mm:mm:mm
mm:mm:mm:mm:mm:mm
mm:mm:mm:mm:mm:mm
```

Le stringhe di caratteri `MDnn:nn:nn:nn:nn:nn:nn` si riferiscono agli indirizzi MAC verso i quali è mandato il messaggio. Invece le stringhe `mm:-`

3.6. REALIZZAZIONE DI UN PROGRAMMA PER LA SINCRONIZZAZIONE

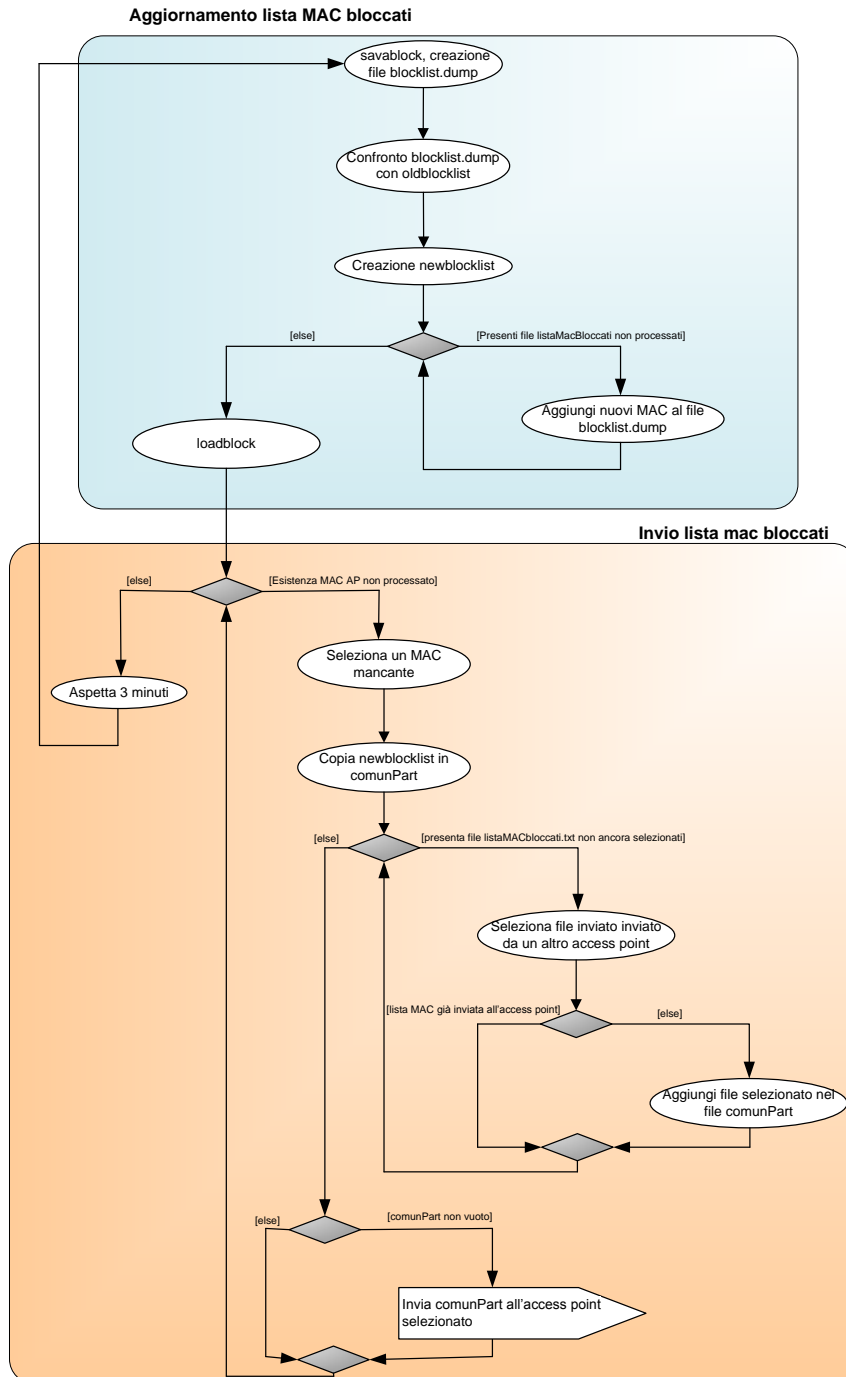


Figura 3.8: Activity diagram rappresentante il programma sincroBlue

`mm:mm:mm:mm:mm` si riferiscono agli indirizzi MAC inviati verso gli access point. Quindi confrontando il MAC dell'access point verso il quale mandare il messaggio con gli indirizzi dei MAC destinatari, si decide facilmente come comporre i messaggi per evitare di eviatare informazioni ridondanti. Nel diagramma non viene esplicitato, per non rendere troppo complessa la rappresentazione, cosa avviene nel caso un messaggio non venga inviato correttamente. La lista dei MAC da inviare viene memorizzata all'interno del file `listaMACnonInviati.txt` nella cartella `/usr/local/obexsender/bin/APs/nn:nn:nn:nn:nn:nn` dove `nn:nn:nn:nn:nn:nn` si riferisce all'indirizzo MAC dell'access point adiacente. Nella stessa cartella, nel file `numeroInviNonRiusciti.txt`, viene anche salvato il numero di volte consecutive che l'invio non ha funzionato. Quando il numero di invii consecutivi non riusciti supera le 3 volte, viene considerato che l'access point ricevente è fuori servizio e quindi il suo indirizzo MAC viene eliminato dalla lista dei MAC definitivi. Il numero di tre tentativi prima di eliminare l'access point è stato deciso tramite una serie di test.

3.6.3 Procedura d'avvio del programma

Dopo la fase di ricerca degli access point attivi inizia la fase di sincronizzazione. Questa fase può essere attivata da un particolare access point il quale inizia ad inviare a tutti gli access point rilevati (si intende che il loro MAC è stato scritto nel file `/usr/local/obexsender/bin/macDef.txt`) il messaggio per avviare il programma di sincronizzazione. Ogni access point che riceve il messaggio di attivazione lo reinvia a tutti gli altri access point rilevati, fuorchè al mittente. L'access point che ha ricevuto il messaggio aspetta 30 secondi e quindi avvia anch'esso il programma di sincronizzazione. Il programma che avvia la fase di sincronizzazione è `startSincro`, mentre il programma che esegue la vera e propria procedura di sincronizzazione è `sincroBlue`. Questa procedura presenta un difetto: se ci sono più di due access point che compongono la rete c'è un'altissima probabilità che un access point riceva due volte il messaggio di attivazione del programma di sincronizzazione. Per

3.6. REALIZZAZIONE DI UN PROGRAMMA PER LA SINCRONIZZAZIONE

evitare che il programma venga attivato più di una volta si è aggiunto un controllo nella fase iniziale che verifica se l'applicazione non sia già in esecuzione. Quando l'access point viene acceso si crea un file `Nome del file` nel quale viene memorizzato se il programma di sincronizzazione non è già stato avviato. Quando il programma di avvio della fase di sincronizzazione viene avviato, invece, viene scritto nel file `Nome file` che il programma è in esecuzione. Questa particolare procedura è stata progettata con l'intento di eliminare i possibili errori che nascerebbero nel caso un access point si spegnesse inaspettatamente.

3.6.4 Comando tramite cellulare

Come fase conclusiva dello sviluppo del programma di sincronizzazione si è creata un'interfaccia per comandare il programma tramite l'invio di messaggi di testo con il Bluetooth. I comandi sono stati provati solamente con il cellulare Sony Ericsson W350; per cellulari di altre marche e modelli sarebbe necessario eseguire ulteriori test o modificare leggermente la procedura perchè vi è rischio concreto che i messaggi di testo inviati abbiano un formato differente. L'implementazione è stata molto semplice: è bastato creare una serie di regole di ricezione, scritte nel file `/etc/obexsender.conf`, che eseguissero dei semplici script. Nella tabella sottostante sono identificati, nella prima colonna, le stringhe che devono contenere i messaggi mentre, nella seconda colonna, le operazioni che vengono eseguite dall'access point.

Stringa	Operazione
startSincro	avvia programma di sincronizzazione
reset	esegue il hardrestart dell'obexsender
hardreset	esegue il hardrestart dell'obexsender e vengono cancellati i file <code>blocklist.dump</code>
resetSincro	riavvia il programma di sincronizzazione solamente per un dispositivo
stopSincro	chiude il programma di sincronizzazione

3.7 Test

3.7.1 Pianificazione del lavoro

In questa sezione vengono spiegati come sono stati condotti i test per testare il programma e ricercare eventuali bug. I test, che verranno spiegati nelle sezioni successive, sono stati ripetuti fino a quando le modifiche apportate al programma hanno fatto sì che rispondesse in modo adeguato agli obiettivi prefissati. Questa sezione, quindi, può essere letta in due modi: sia come quella dei test, se si guardano come sono stati eseguiti, sia come quelle delle funzionalità offerte dal programma, se si guardano i risultati raggiunti. Nelle seguenti sezioni verranno spiegati come sono stati eseguiti i diversi test.

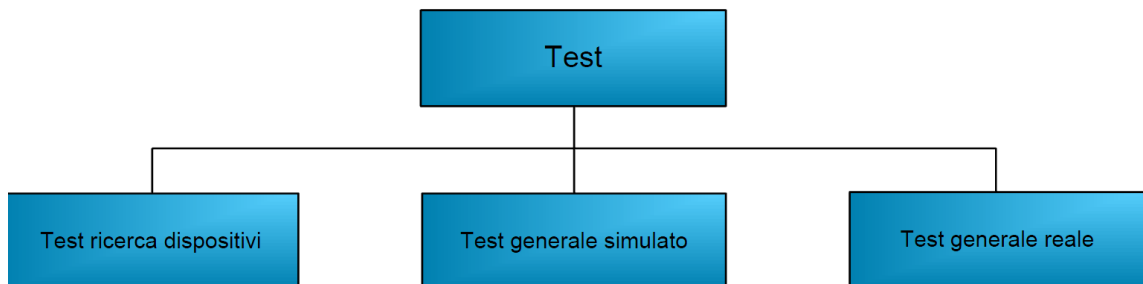


Figura 3.9: WBS dei test effettuati

3.7.2 Rete utilizzata

Per eseguire i test sono stati utilizzati tre access point: un modello A3201 e due server A. Per facilitare i test si è impostata la rete Bluetooth per avere una massima copertura di 10 metri. Per visualizzare l’invio e la ricezione di dati è stato utilizzato un cellulare, modello Sony Ericsson W350.

3.7.3 Test di ricerca dei dispositivi

Obiettivi

In questo test si è voluta provare la funzione di ricerca su una rete vera. L'operazione di ricerca è stata comandata tramite cellulare e doveva restituire a questo la lista dei codici hardware dei dispositivi visti e i dB dei segnali rilevati.

Realizzazione

La realizzazione è stata molto semplice: dopo aver installato il software necessario è stato eseguito il programma di ricerca, prima con tutti i dispositivi aperti e poi spegnendo a turno i diversi access point fino a spegnerli tutti tranne quello che ha effettuato la ricerca. Per ogni fase di test venivano controllati i dispositivi trovati e i file di log realizzati dal programma.

Problematiche affrontate

In questo test non sono stati riscontrati particolari problemi. Da menzionare solamente la scoperta di diversi errori causati dalla presenza di tre antenne Bluetooth all'interno degli access server A. Il problema era che venivano rilevati tre diversi access point e quindi inviati tre diversi messaggi di ricerca perché, a tre antenne, corrispondono tre MAC diversi. Il problema è stato risolto scegliendo solamente un MAC per ogni singolo dispositivo ed usando solo quello.

3.7.4 Test generale simulato

Obiettivi

Si è voluto provare il programma di sincronizzazione analizzando in modo preciso le operazioni di aggiornamento della lista dei dispositivi bloccati e l'invio dei messaggi di invio. Si è cercato di capire quando un messaggio

venisse inviato, quando la lista dei MAC venisse bloccata, ma soprattutto che non venissero inviate informazioni ridondanti.

Realizzazione

La realizzazione del test è stata un'operazione complessa e molto lunga. Il primo problema è stato quello di capire come poter creare le liste da inviare e da aggiornare dei MAC bloccati non potendo, per ovvi motivi, poter utilizzare decine di cellulari. Per supplire a questa mancanza si è deciso di modificare il comando `service obexsender saveblock` (salva la lista dei MAC dei dispositivi a cui è impedito l'invio di file) aggiungendo le seguenti righe allo script `/etc/init.d/obexsender`

```
/usr/local/obexsender/bin/randomMAC "Stringa di 2 lettere"  
cat macRandom.txt >> /var/lib/obexsender/blocklist.dump
```

incui viene lo script `randomMAC` che genera un numero casuale di MAC casuali, tra 0 e 15, e gli inserisce nel file `blocklist.dump`. I MAC generati presentano le ultime due cifre dell'indirizzo uguali a quelle date; questo è necessario per riconoscere successivamente da quale dispositivo è stato inviato un MAC. Il programma di sincronizzazione `sincroFind` è stato modificato per eseguire l'operazione di sincronizzazione 8 volte e quindi chiudersi. Ad ogni iterazione il programma è stato modificato per generare i log dei MAC generati, file ricevuti, messaggi inviati e molte altre informazioni, necessarie per valutare l'effettivo funzionamento del programma. Al termine di un'esecuzione del programma si sono controllati a mano i vari log dei dispositivi confrontandoli con il file `blocklist.dump` di ogni dispositivo per capire quando veniva inviato un file e ricevuto e capire se erano state eseguite delle operazioni ridondanti.

Conclusioni

Il programma provato ha risposto a tutti gli obiettivi proposti. Deve essere aggiunto, comunque, che la rete utilizzata è formata da 3 dispositivi

dove tutti vedono tutti. Sarebbe necessario eseguire test più approfonditi cambiando la disposizione della rete e utilizzando più access point.

3.7.5 Test generale reale

Obiettivi

Questo test ha gli stessi obiettivi del test precedentemente sviluppato con la modifica che oltre a MAC generati casualmente vengono serviti dei veri cellulari. Si è voluto vedere se il programma, gestendo dei dispositivi reali, funzionasse in modo corretto, equivalente al test precedente.

Realizzazione

La procedura utilizzata è identica al test precedente ma è stata modificata la disposizione della rete. Sono stati utilizzati solo due access point posizionati in modo tale da evitare l'invio simultaneo di messaggi verso un cellulare. Per avere la certezza che un access point avesse negato l'invio dei file a un cellulare si è controllato il file di log del programma `obexsender` nel quale viene riportato per quali dispositivi è bloccato l'invio dei messaggi. Il programma è risultato funzionare correttamente e non sono stati rilevati problemi o errori. Per non rendere la realizzazione troppo lunga è stato utilizzato solamente un cellulare, che comunque basta per valutare se le modifiche apportate al file `blocklist.dump` hanno avuto effetto.

Conclusioni

Il programma si è rilevato funzionare correttamente e si è provato che non vi è alcuna differenza tra il test simulato e il funzionamento reale. Tramite questo test, inoltre, si è accertato che i tempi inseriti all'interno del file `blocklist.dump` sono gestiti in modo corretto.

CAPITOLO 3. REALIZZAZIONE DI UN PROGRAMMA PER LA
SINCRONIZZAZIONE

Capitolo 4

Conclusioni

In questa tesi sono stati sviluppate delle applicazioni per degli access point Bluetooth. Il primo lavoro sviluppato è stato la realizzazione di un interfaccia HTML per la configurazione di una bacheca scolastica realizzata tramite i dispositivi Bluetooth. Come secondo lavoro è stata realizzata un applicativo che crea una rete mesh tra access point per sincronizzarli. Il programma di sincronizzazione serve per poter gestire diversi access point come un unico dispositivo. Si è riusciti a realizzare un applicazione funzionante che non necessita nessun tipo particolare di configurazione, è solamente necessario inviare il comando di avvio e di stop tramite il cellulare. L'applicazione comunque necessita ulteriori sviluppi come realizzare un'applicazione per cellulare per facilitare e autorizzare solamente il personale abilitato a comandare la rete. Per come sono stati pensati i comandi, un'utente qualsiasi sapendo i codici dei programmi potrebbe fermare e riavviare a piacere i dispositivi. Come ulteriore sviluppo si potrebbe migliorare l'algoritmo di ricerca iterato che risulta troppo lento. Sono invece necessari, inoltre, ulteriori test per verificare se si può diminuire l'intervallo tra due operazioni di aggiornamento e gli intervalli fra le diverse ricerche.

Appendice A

Manuale dell'interfaccia HTML per la bacheca scolastica

A.1 Installazione

L'installazione dell'interfaccia html è una procedura non automatizzata quindi deve essere prevalentemente svolta a mano. Di seguito vengono descritte le operazioni da compiere per installare il software.

A.1.1 Collegamento dispositivo

L'access point deve essere collegato a un computer tramite cavo ethernet. Successivamente, bisogna aspettare qualche minuto, si deve lanciare l'applicazione `WrapFinder2.exe` e avviare l'operazione di ricerca dei dispositivi premendo il bottone `find devices`.

A.1.2 Installazione applicazione

Dall'applicazione `WrapFinder2.exe` installare il software `ServerBacheca` premendo il bottone `Upload software`. Selezionare il `Server Bacheca` dalla cartella applicazione e premere il tasto OK e l'applicazione viene installata. Accertarsi di aver inserito il nome utente e la password del dispositivo. Per

rendere operativa l'applicazione bisogna inserire le seguenti righe alla fine del file `/etc/obxsender.conf`

```
reply {  
    keyword richiestaInfo  
    exec /usr/local/obxsender/bin/ServerBacheca  
    file Info.txt  
}
```

A.1.3 Installazione interfaccia

Copiare tutti i file contenuti nella cartella `html` nella cartella `/var/www/html/`. Successivamente bisogna copiare tutti i file contenuti in `cgi-bin` nella cartella `/var/www/html/cgi-bin`, se la cartella non è presente bisogna crearla. Attenzione il file `index.html` deve essere sovrascritto, se sono state apportate delle modifiche al file originario tali modifiche andranno perse. Per eseguire queste operazioni si possono utilizzare due metodi. Il primo metodo consiste nell'utilizzare il programma `setup` tramite l'interfaccia web. Per accedere alla pagine di `setup` basta cliccare due volte sopra la riga che mostra dove è stato trovato il proprio access point, si dovrebbe aprire il proprio browser internet di default con caricata la pagina iniziale di configurazione del dispositivo. Se la pagina non viene aperta automaticamente basta inserire l'indirizzo IP dell'access point è indicato nella riga dove è visualizzato il proprio dispositivo come indirizzo nel proprio browser internet. Accedendo nel menu alla sezione `setup`, inserito il nome utente e la password, bisogna entrare nella sezione `setup - Advanced - browse all files` dove c'è l'opzione per l'inserimento dei file. Il secondo metodo è l'utilizzo di un programma per la navigazione tramite protocollo `ssh`, un possibile programma è `winScp` per windows. Il passo successivo è concedere i privilegi di esecuzione per gli script `cgi`. Questa operazione può essere eseguita lanciando il comando

```
chmod +x /var/www/html/cgi-bin/*.cgi
```

da un terminale `ssh` collegato all'access point.

A.2 Utilizzo interfaccia HTML

A.2.1 Accesso Al programma

Per accedere al programma, bisogna collegarsi tramite un comune browser internet alla pagina di impostazione dell'access point. Per accedervi basta inserire il numero dell'indirizzo IP della macchina. Nel menu compaiono il link alla bacheca elettronica. Nella bacheca elettronica vengono visualizzate i diversi menu: Gestione classi, registrazione utenti, gestioni file, log.

A.2.2 Gestione classi

Serve per poter inserire nuove classi o eliminare quelle vecchie. Nella parte superiore della pagine viene visualizzato un elenco di tutte le diverse classi. Le classi sono le cartelle presenti all'indirizzo `/usr/local/files/`. A lato del nome di ogni cartella è presente un tasto che consente l'eliminazione della cartella. **ATTENZIONE** l'eliminazione della cartella comporta anche l'eliminazione definitiva di tutti i file e di tutte le regole di invio dei file. Il programma, comunque, richiede la conferma di tale operazione. A fondo pagina è presente un riquadro che consente l'inserimento di nuove classi. Inserendo il nome della classe nel riquadro e premendo il tasto **aggiungi** viene creata una nuova classe, in pratica viene creato una cartella all'indirizzo `/usr/local/files/` con il nome dato. Il nome dato può essere di qualsiasi tipo, ma la lunghezza è limitata a 15 caratteri. A fianco del tasto **elimina** è presente il tasto **trasferisci**. Tramite questa funzionalità è possibile spostare tutti i file con le regole di invio e i MAC dei genitori alla classe successiva se il nome della classe è composta da una lettera seguita da una cifra. Esempio la classe A1 (prima A) diventerebbe la classe A2 (seconda A) mantenendo tutti i file e tutti i MAC. La funzione non funziona se è già presente la classe di destinazione (A1 non viene trasferita se esiste la classe A2). Questa funzionalità serve per aiutare l'utente a trasferire i file al termine dell'anno scolastico.

A.2.3 Gestione file

In questa pagina vengono gestite le varie operazioni dei file delle classi. La pagina è divisa in due frame (sotto-pagine). Nel frame sinistro è presente l'elenco delle diverse classi, premendo su una di esse nel frame destro vengono visualizzati i dati dei diversi file della cartella. I file vengono visualizzati come un elenco dove il nome più a destra è il nome del file seguito dalla data di immissione del file e per ultimo vi è un tasto elimina che permette la cancellazione del file. Premendo il tasto **elimina** viene cancellato il file ed inoltre viene eliminata la regola di invio dal file `obexsender.conf`, il file quindi non verrà più distribuito. Sotto l'elenco c'è un riquadro per l'inserimento dei nuovi file premendo sul tasto **sfoglia** viene aperto un browser dei file che permette la scelta del file che si vuole che l'access point distribuisca. **ATTENZIONE** non è consentito l'invio di file aventi nomi con spazi, ed inoltre inviando file contenti diversi punti l'estensione del file non viene visualizzata in modo corretto. L'inserimento di un nuovo file aggiunge automaticamente la regole di invio nel file `obexsender.conf` ed inoltre il dispositivo viene automaticamente riavviato, quindi le modifiche sono subito attivate. L'inserimento di file di dimensioni di pochi MB impiega molto tempo, anche qualche minuto. **ATTENZIONE** non vengono garantite le funzioni del programma se vengono eseguite le operazioni di update dei file o modificate le regole di invio di modo manuale o tramite il programma di `setup`, quindi è caldamente consigliato di non eseguire nessuna operazione manualmente, ma affidarsi interamente al programma.

A.2.4 Registrazione utenti

Tramite questa pagina si accede alla registrazione del cellulare. Bisogna inserire obbligatoriamente il nome e il cognome e quindi premere il tasto **genera**. Quindi viene visualizzato un codice numerico. Questo numero serve per la registrazione del cellulare, inviando il numero tramite il cellulare verso l'access point si registra il mac del cellulare del genitore. A lato di ogni

A.2. UTILIZZO INTERFACCIA HTML

MAC visualizzato c'è il tasto **elimina** che cancella il MAC registrato. Oltre al tasto **elimina** sono presenti anche due parole **successivo** e **precedente**. Premendo il tasto **successivo** trasferisce il MAC della linea nella classe successiva, se questa esiste e il nome della classe è del tipo lettera seguita da una cifra. Il tasto **precedente** trasferisce il MAC nella classe precedente. Queste funzionalità servono per gestire la presenza degli eventuali studenti bocciati in una classe. In fondo alla pagina c'è un link **gestione manuale** che apre una pagina dove la gestione degli indirizzi è gestita in modo manuale. In questa pagina si possono inserire i MAC dei cellulari dei genitori conoscendo direttamente il MAC del cellulare, evitando quindi la fase di registrazione.

A.2.5 Report

Viene visualizzata la lista dei file presenti nella diverse classi e per ogni file viene visualizzato il numero d'invii avvenuti con successo e o d'invii falliti. A fianco del numero di invii c'è un tasto **dettagli** premendolo si visualizza un elenco che indica la data, ora dell'invio dei dati, e l'utente che ha ricevuto il file.

Appendice B

Manuale del programma di sincronizzazione

B.1 Installazione

Per eseguire l'installazione l'access point deve essere collegato tramite il cavo ethernet a un computer, l'operazione risulta essere complessa e deve essere eseguita completamente manualmente. Di seguito vengono descritti i passi per installare il software.

1. Impostare il programma obexsender in modo tale che salvi i file ricevuti nella cartella `/root/`. Per fare questo bisogna inserire nella sezione del menu del setup HTML: Setup-iWrap-Profiles-OBJP alla voce Optional parameters for server la seguente stringa in un'unica linea:

```
---bdaddr $b ---prefix $b-$P- ---fork '/bin/cp $$t  
/root/$$p$$d-$$T
```

Quindi premere il tasto `save` e successivamente il tasto `reset`.

2. Copiare tutti gli script dalla cartella Script nella cartella di installazione nella cartella `/usr/local/obexsender/bin` nell'access point. Questa operazione può essere eseguita tramite l'interfaccia HTML oppure utilizzando il comando `scp` da terminale. Dopo questa operazione bisogna

rendere eseguibili tutti gli script eseguendo il comando dal terminare ssh.

```
chmod +x /usr/local/obexsender/*
```

3. Aggiungere le regole di risposta del programma obexsender. Bisogna aggiungere alla fine del file `/etc/obexsender.conf` il contenuto del file `Impostazioni obexsender.text` che è nella cartella di installazione.
4. Aggiungere al file `/etc/rc.d/rc.local` la stringa:

```
echo "Prog sincro chiuso $( date ) >> /root/file_accensione
```

5. Spegner e riaccendere l'access point.

B.2 Disposizione della rete

Dopo aver installato il software necessario bisogna posizionare gli access point in modo da coprire l'area maggiore. Di seguito viene descritta la procedura da seguire.

1. Posizionare fisicamente gli access point nelle posizioni volute. Ricordarsi che gli access point dispongono di una copertura limitata a circa 100 metri che diminuisce in presenza di barriere architettoniche, quindi posizionare gli access point a una distanza (queste distanze sono solo indicative) superiore dei 50 metri ma inferiore ai 100 metri.
2. Accendere tutti gli access point della rete.
3. Inviare a ogni singolo access point un messaggio di testo con la stringa `startRicerca` tramite una connessione Bluetooth. Dopo circa 3 minuti l'access point invia al cellulare i codice identificativi degli access point rilevati attivi nelle vicinanze con la potenza del segnale. Il codice identificativo può essere letto sull'adesivo incollato sul fondo dell'access

B.3. AVVIARE IL PROGRAMMA

point. Nel caso il codice non fosse leggibile può essere trovato lanciando il comando `wrapid` collegando il dispositivo tramite connessione ssh. Quest'operazione può essere compiuta inviando contemporaneamente più messaggi con più cellulari.

4. Controllare che ogni access point venga rilevato da almeno un altro dispositivo. Se per caso un access point non viene rilevato riposizionare gli access point e quindi rieseguire il punto 2. Questa operazione può essere anche compiuta per cercare una nuova disposizione della rete per coprire un'area maggiore.

B.3 Avviare il programma

Per accendere al programma è necessario inviare un messaggio con la stringa `startSincro` a un access point qualsiasi. In questo modo il programma si avvia in tutti gli access point della rete.