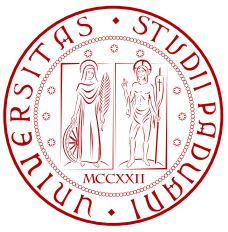

UNIVERSITÀ DEGLI STUDI DI PADOVA
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
CORSO DI LAUREA IN INGEGNERIA INFORMATICA
TESI DI LAUREA

STUDIO, PROGETTAZIONE, SVILUPPO DI UN SISTEMA DI ACCESS POINT BLUETOOTH PER IL PROXMITY MARKETING E ALTRI SERVIZI AVANZATI

RELATORE: Ch.mo Prof. Filira Federico

LAUREANDO: *Campanaro Daniele*

Padova, 27 Aprile 2010

Indice

Sommario

La presente tesi descrive la mia esperienza di tirocinio svoltasi presso *ne-t by Telerete Nordest*, azienda di ICT che opera nel Padovano, che recentemente si sta espandendo oltre i confini della provincia. L'obiettivo del tirocinio è lo studio di Access Point e degli Access Server di un particolare produttore per realizzare un sistema in grado di inviare contenuti informativi/pubblicitari e personalizzati in diversi formati adattati al particolare dispositivo. Dallo studio di casi reali si sono realizzati moduli standard, scrivendo un manuale interno per permettere di usufruire delle funzioni utili al Business anche ai non esperti.

Capitolo 1

INTRODUZIONE

Questa tesi rappresenta un diario di bordo della mia esperienza durante il tirocinio a Telerete Nordest Srl. Nei sei mesi da Ottobre a Marzo mi sono occupato di netBlue, un progetto già avviato all'interno dell'azienda, che permette di inviare messaggi utilizzando la specifica industriale Bluetooth. Nel dettaglio mi è stato affidato lo studio dei nuovi apparati adottati in azienda e la realizzazione di applicazioni avanzate per l'invio di contenuti informativi o pubblicitari a dispositivi mobile. Come richiesto dall'azienda e in accordo con il ch.mo ing. Filira in questa tesi non viene fatto riferimento esplicito all'azienda che ci ha fornito gli apparati hardware per segretezza industriale. Nei primi mesi del mio tirocinio mi sono concentrato sulle funzionalità di base di questi apparati, testando a fondo le possibilità messe a disposizione dal sistema operativo. Successivamente mi sono impegnato nello sviluppo di applicazioni avanzate utilizzando un ambiente di sviluppo che permette di utilizzare linguaggi ad alto livello. Gli input maggiori sono arrivati dal settore commerciale con il quale ho collaborato per tutta la durata del tirocinio per stabilire sia gli obiettivi del mio lavoro, ma anche per fornire al cliente finale un prodotto che fosse in grado di erogare un servizio adeguato. Ho avuto inoltre la fortuna di poter testare, nello stesso periodo, alcune applicazioni realizzate; la mostra di Zaha Hadid e Polesine Innovazione sono due clienti che hanno scelto di pubblicizzarsi in questa modalità che prende il nome di Proximity Marketing [1].

Nell'ultima parte del mio tirocinio, infine mi sono concentrato sulle modalità di iterazione tra i dispositivi mobile e i cellulari. É stata realizzato un programma da installare nei cellulari per gestire automaticamente la connessione e permettere di fornire un' applicazione per lo scambio dati tra i device e i cellulari.

1.1 Contesto

La possibilità di inviare contenuti informativi a dispositivi portatili ha aperto la strada nuovi interessanti servizi. Il proximity marketing o mercato di prossimità è solo una delle possibilità che si basano su questa tecnologia. Con questi termini si vuole indicare l'invio di materiale pubblicitario a cellulari che si trovino in una particolare zona, delimitata dal punto di vista geografico [2]. Le possibilità di personalizzare il contenuto pubblicitario sono già molto spinte; invio di file in formati avanzati come mp3, video, jpg, contenuti in java e realizzazione di semplici applicazioni sono features in grado di realizzare delle campagna pubblicitarie molto accattivanti. L'utente stesso interrogando l'access point o il server può richiedere delle informazioni. Gli scenari sono molti: interrogazione di una pensilina sull'autobus da prendere per arrivare in un determinato luogo, contenuti turistici scaricabili sul telefonino in prossimità di un luogo di particolare interesse. Già alcune aziende, anche nel suolo italiano, hanno iniziato a interessarsi a questa tecnologia, siamo nella fase denominata: "early innovation", i mezzi messi a disposizione non sono ancora stati esplorati a sufficienza, ma c'è molto interesse. Mentre in altri paesi come gli Stati Uniti e la Spagna ci sono cause legate al disturbo e all'invadenza del mezzo e sono in studio disegni di legge per regolarizzare questo tipo di servizio in Italia non accade nulla di tutto questo [3]. Il progetto in cui si inserisce il mio lavoro si vuole tutelare sviluppando dei sistemi ad hoc per evitare la lesione della quiete di chi non voglia ricevere il contenuto. Per esempio un aspetto da affrontare è il *pairing* [4] tra il cellulare e l'access point; attraverso di un programma da istallare nel cellulare si può velocizzare lo scambio di messaggi per velocizzare la comunicazione. I problemi di natura tecnologici sono evidentemente legati alla diversità sia HW sia SW dei vari modelli di cellulari. Tuttavia non è l'unico problema da risolvere: alcuni cellulari, pur utilizzando la specifica Bluetooth implementano particolari soluzioni atte a proteggere i propri clienti da contatti Bluetooth indesiderati (BlackBerry e Iphone) e alcuni di essi non sono in grado di ricevere formati avanzati, per questo i contenuti vanno devono esser realizzati utilizzando formati adeguati ai cellulari (per esempio per i video lo standard attuale è il 3gp).

1.2 Obiettivi

L'obiettivo del tirocinio è lo studio degli Access Point e degli Access Server di un particolare fornitore (che come detto non vengono citati direttamente) per

la realizzare un sistema in grado di inviare ad apparecchi mobili (cellulari, palmari, smartphone) contenuti informativi/pubblicitari in diversi formati adattati al particolare dispositivo. Le modalità di scambio sono principalmente due: una prima attiva; dove sarà l'access point a inviare per primo un file a un dispositivo in prossimità, e una passiva, dove sarà l'utente a esprimere una preferenza o esigenza conoscitiva per ricevere il contenuto desiderato. I due modi sono implementabili in un'unica applicazione per rendere più efficace l'esperienza. Attraverso un contatto con l'ufficio vendite è richiesto lo studio di casi reali per realizzare di moduli standard da utilizzare per soddisfare le esigenze dei diversi clienti, scrivendo un manuale interno che permetta di usufruire delle funzioni utili al Business anche ai non esperti In parallelo si sono ricercate altre soluzioni, sia HW sia SW in un'ottica di riduzione dei costi e d'aumento delle performance, con una particolare attenzione alla possibilità di riutilizzare il SW già scritto.

1.3 Progettazione

Il mio lavoro è stato portato avanti come una serie di mini-progetti di lunghezza variabile, per il raggiungimento dell'obiettivo finale. La gestione dei progetti ha aiutato ad anticipare i problemi che si sarebbero potuti verificare, e aiutare a organizzare gli appuntamenti in base alle esigenze dei miei colleghi e soprattutto a richiedere con sufficiente anticipo le risorse, sia SW che HW di cui ho avuto bisogno settimana dopo settimana. Durante il tirocinio mi sono trovato a sia collaborare con personale dell'azienda sia a dover interagire con entità esterne alla quale l'azienda era legata. Arrivare a questi appuntamenti preparato adeguatamente è una vera necessità, è fondamentale esprimere le proprie esigenze in modo chiaro e preciso, sia per riceve aiuto, sia per trasmettere le conoscenze acquisite. L'oggetto del mio studio è una tecnologia ancora poco approfondita all'interno dell'azienda ma che ha permesso di realizzare servizi a basto costo. Proprio per questo motivo tutto il mio operato deve essere rintracciabile, ossia si ha la necessità di risalire alla storia, al know how e all'ubicazione di tutto ciò che ho prodotto in modo che possa essere riutilizzato anche dopo la fine del mio tirocinio: test, schede tecniche e manuali interni sono le soluzioni che ho scelto di utilizzare per mantenere traccia delle mie analisi.

1.4 Strumenti utilizzati

- Open Office: Editor di testo open source, usato per la stesura dei manuali.

- GanttProject: Strumento indispensabile per il Project Management.
- Start UML 5.0: Permette la creazione di diagrammi UML utili per la progettazione del software.
- Gimp: Uno tra i più famosi editor di immagini, usato per l'ottimizzazione di alcuni contenuti.
- Audacity: Programma per la gestione di file sonori
- Emicsoft Converter: Programma che permette la conversione di filmati in formato 3gp, standard video per i cellulari.
- WrapFinder: Softare per la ricerca di apparati nella LAN a cui si è connessi.
- Dev C++: Ambiente di sviluppo per i linguaggi C e C++
- Putty: un tra i più famosi Hyper Terminal
- TexMaker: Editor per Latex

1.5 Modalità di esecuzione dei test

In parallelo con la stesura del manuale si è sentita l'esigenza di registrare in modo efficace i test realizzati; i test redatti in questo modo diventano tracce delle operazioni necessarie per abilitare in modo corretto le funzioni del device, e in caso di altri acquisiti una valida base per la realizzazione di un test per il corretto funzionamento. Specifiche dei test:

- Il titolo deve essere esplicativo.
- Codice identificativo univoco.
- Devono essere presenti le risorse sia HW che SW necessarie.
- Devono essere indicati data ed esecutore del test.
- L'obbiettivo deve essere chiaro.
- La procedura deve essere chiara.
- Devono essere chiari i risultati.
- Deve essere chiaro cosa i risultati comportino.

- Il test deve essere ripetibile.

Il test si divide in 6 parti:

1. Apetto da testare
2. Progettazione della procedura
3. Esperimento
4. Raccolta dei dati
5. Trattamento
6. Analisi

Ogni test deve essere eseguito per un determinato obiettivo: è controproducente in un unico test tentare di verificare tutte le funzionalità e le routine. Per capire a fondo il funzionamento di una particolare feature e trovare conferma alle proprie ipotesi è più conveniente realizzare molti piccoli test di un unico complesso. Altrettanto importante è la progettazione della procedura. Cosa devo realizzare? Cosa devo misurare? Quali sono i dati salienti che possono confermare o smentire la mia tesi? Quali sono i tempi e i costi del test? E' conveniente realizzarlo? Quali sono le risorse che sono necessarie per il corretto svolgimento dei test? Prima di iniziare una qualsiasi operazione è necessario rispondere a tutte queste domande per non perder tempo inutilmente e sprecare risorse. La procedura deve essere scritta tenendo in considerazione il livello di conoscenza tecnologico di chi andrà a leggerla per consultazione e di chi andrà a ripeterla. La procedura deve svolgersi rigorosamente come è stato progettato, seguendo la scaletta passo-passo. Nel caso durante l'esperimento accada qualcosa d'inaspettato l'anomalia deve essere registrata e successivamente analizzata. Un esperimento che ha come risultati dati imprevisti può essere più utile di molti test completati senza intoppi. I dati dovranno essere precisi e attendibili, a tale scopo è necessario che ci sia un'adeguata attenzione al loro reale significato e un'adeguata cura nella loro trascrizione ed elaborazione. I dati forniscono informazioni attendibili e utili, se la loro raccolta avviene con:

- Chiarezza (deve essere chiaro lo scopo per cui avviene eseguita la raccolta dati, per evitare di raccogliere dati inutili, o errati).
- Facilità (il supporto per la raccolta deve essere semplice e chiaro).

- Completezza (è necessario pensare in dettaglio quali siano i dati e le reazioni del sistema da registrare, per evitare di accorgersi che mancano dati importanti).

Il trattamento dei dati è altrettanto importante, devono essere conservabili in modo permanente, non va dimenticato che tali test rappresentano una vera e propria risorsa per l'azienda. Infine l'analisi dei dati permette di ottenere il massimo delle informazioni possibili, valutare necessità di realizzare altri test. Per fortuna i dati che sono stati principalmente di duplice natura:

1. Qualitativi, in altre parole, misurabili con strumenti e procedure riferibili ai sistemi metrologici internazionali (numeri di cellulari serviti, tempo trascorso tra una trasmissione e l'altra.).

2. Oggettivi, ossia dotati di una o più proprietà non contestabili (il messaggio è arrivato, nel log è possibile trovare una determinata informazione formatta in un certo modo...).

I test realizzati non sono riportati in questa tesi come richiesto dal Telerete.

1.6 L'azienda

In questa sezione presento brevemente l'azienda nella quale ho svolto il mio tirocinio.

1.6.1 Ne-t by Teleretenordest s.r.l.

Ne-t by Telerete Nordest srl è una società di servizi nelle telecomunicazioni e nell'ICT che ha lo scopo di aiutare l'adozione e l'impiego delle nuove tecnologie per rendere le città sicure e a misura d'uomo. La sua azione si esplica in attività di progettazione, realizzazione ed esercizio di sistemi per la sicurezza e sorveglianza, la connettività e l'accesso ai servizi pubblici e privati, con specifica attenzione al cittadino e agli enti dedicati al suo servizio [5]. I servizi offerti spaziano in diversi campi delle tecnologie per la città digitale:

- Gestione della MAN cittadina in fibra e wireless, pluriconnettività, satcom, server farm, security e disaster recovery, postazioni cellulari GSM, UMTS, DVHB, videosorveglianza

- Progettazione e integrazione di tecnologie per la mobilità (AVM, tram, ZTL)

- Call center e servizi di prenotazione monumenti e manifestazioni
- Portali e applicazioni informatiche di supporto all'E-Government e per le aziende
- Fornitura hardware di supporto

Telerete Nordest applica il principio del System Integration, ha quindi le competenze nella gestione di progetti complessi che sono indispensabili per realizzare sistemi completi di:

- Networking
- Connettività
- Servizi di web call/contact center
- E-service
- Info-mobilità
- Domotica
- E-government
- Integrazione di applicazioni.

Ne-t fornisce inoltre servizi di consulenza per ottimizzare i processi organizzativi e servizi a supporto dell'intero ciclo di vita della soluzione. Con una struttura di oltre 60 collaboratori, è in grado di assicurare assistenza post-vendita e servizi di manutenzione ai clienti in modalità 7 giorni su sette, 24 ore su 24.

1.6.2 La mission

Fornire una gamma di servizi completa che, attraverso l'utilizzo delle tecnologie più avanzate, rispondano alle necessità del tessuto urbano e delle aziende, sia pubbliche che private che in esso operano. Porsi come interlocutore privilegiato per chiunque abbia la necessità di ottenere in tempi rapidi risposte concrete a esigenze di natura tecnologica, fornendo inoltre servizi accessori con il valore aggiunto della professionalità espressa dai singoli collaboratori. Affiancare gli Enti con il proprio know-how nello sforzo di offrire la maggiore accessibilità possibile ai servizi, sia attraverso l'uso di media diversi sia attraverso la connettività diffusa, con un'attenzione particolare alla tutela della sicurezza dei cittadini.

1.6.3 Aspetto societario

I soci di Telerete Nordest sono:

- APS Holding s.p.a., proprietaria del 51% delle quote
- Pronet tecnologies, proprietaria al 38%
- Serenissima Infracom Italia, con una quota pari a l'8%
- Camera di commercio di Padova proprietaria del restane 3%

1.6.4 Serivizi offerti

L'azienda si caratterizza per offrire servizi molto diversificati, ai suoi clienti e alle aziende partner offrendo un ambiente di lavoro intellettualmente molto vivace. Al centro dell'attenzione di tutto l'operato dell'azienda c'è la diffusione di tecnologia che migliori la qualità dei servizi e della vita nel territorio Patavino.

1.6.5 Tassonomia dei servizi

- Progetti di integrazione tecnologica in ambito urbano: progettazione ed implementazione di tecnologie urbane e progetti integrati, tra cui la rete del metrobus di Padova (nell'ambito del progetto SAE - Sistema di Ausilio all'Esercizio)
- Videosorveglianza, per conto di Carabinieri, Polizia Municipale e Questura:
  - Progetto Padova città sicura - Via Anelli
  - Progetto Padova città sicura - Palazzo Moroni
  - Consorzio Padova Ovest
  - Pensiline alle fermate del metrobus padovano
  - Stadio Euganeo
  - ZTL (Zone a Traffico Limitato)
- Installazione infrastrutture e gestione di rete per servizi di connettività wireless:
- Padova WiFi
- Unipd WiFi

- Monselice WiFi
- Fornitura di connettività: MAN cittadina (protocollo HiperLAN), apparecchiature tra cui hotspot e antenne WiFi, gestione di postazioni hositing per antenne di telefonia mobile, gestione della rete in fibra ottica e disaster recovery.
  - Servizi informatici:
  - Sviluppo ed installazione software
  - Sviluppo website, servizi ed applicazioni web
  - Servizi avanzati di gestione anagrafe animale
- Call center: servizi informativi per clienti, servizi di CRM, assistenza e Help Desk, booking di eventi culturali
- Campagne di eventi nel padovano
- Portali e applicazioni informatiche di supporto all'E-Government e per le aziende
- Fornitura hardware di supporto

1.6.6 Progetto in azienda

Il mio tirocinio si inserisce nel progetto net-Blue; in questo capitolo viene presentato il suo stato all'inizio del mio tirocinio. Il progetto ha lo scopo di sviluppare soluzioni che sfruttino la tecnologia Bluetooth per offrire ai propri clienti un canale veloce e diretto di comunicazione personale per l'invio di contenuti a cellulari.

Il primo Hardware impiegato è stato un access server dotato di 4 antenne e di numerosi led di stato di un fornitore differente [7]. Dal punto di vista HW è molto simile all'access server dell'attuale fornitore, ma la differenza maggiore risiede nel firmware. Infatti il firmware del device precedente non permette funzionalità avanzate di default, e il primo progetto è stato quello di realizzare un'applicazione in grado di gestire una campagna Bluetooth in modo più efficace, completo e profondo.

Il firmware preinstallato utilizza dei file di configurazione e applicazioni in linguaggio python per la gestione degli invii attraverso il Bluetooth. Python è un linguaggio di programmazione ad alto livello, rilasciato pubblicamente per la prima volta nel 1991 con licenza Open Source, supporta diversi paradigmi di

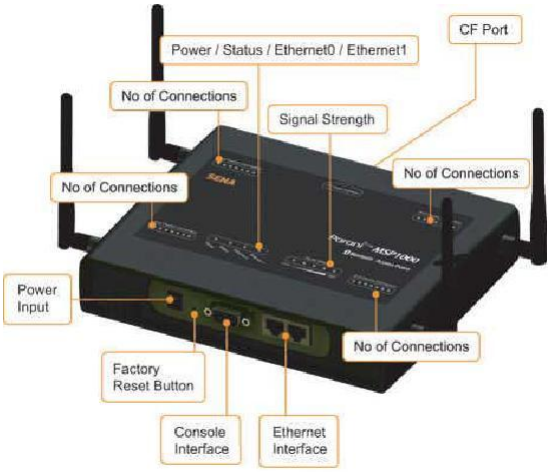

Figura 1.1: Access Point fornitore precedente

programmazione, come quello object-oriented (con supporto all'ereditarietà multipla), quello imperativo e quello funzionale, ed offre una tipizzazione dinamica forte.

Python è un linguaggio pseudocompilato: un interprete si occupa di analizzare il codice sorgente (semplici file testuali con estensione .py) e, se sintatticamente corretto, di eseguirlo. In Python, non esiste una fase di compilazione separata (come avviene in C, per esempio) che generi un file eseguibile partendo dal sorgente [8].

In dettaglio questi si sono gli obiettivi raggiunti dal precedente lavoro:

1. Il sistema deve offrire tre tipologie di invio file:
   (a) Invio Singolo: si invia un file alla volta ai dispositivi Bluetooth rilevati.
   (b) Invio Multiplo: si inviano tutti i file della tranche corrente contemporaneamente ai dispositivi Bluetooth rilevati.
   (c) Test: modalità utilizzabile in fase di ricerca sui dispositivi Bluetooth.
2. Per ogni file deve essere presente una grey-list, dove all'interno si inseriscono i MAC-Address dei dispositivi che:
   (a) hanno ricevuto correttamente il file
   (b) hanno rifiutato il file.
3. Il sistema deve mettere a disposizione degli utenti i seguenti file di log:
   (a) Log_file: vengono salvate tutte le informazioni riguardanti le varie attività del sistema

(b) discovered: vengono salvate le informazioni (data, ora, MAC-Address, nickname e tipologia del dispositivo) riguardanti i dispositivi rilevati dall'attività di discovery.

(c) received: vengono salvate le informazioni riguardanti l'invio di un file ad un dato dispositivo Bluetooth (data, ora, file, risultato dell'invio, MAC-Address del dispositivo e nickname).

4. Il sistema deve dare la possibilità di creare più tranche (gruppi) di invio file

Questi obbiettivi sono stati raggiunti realizzando uno script e realizzando una interfaccia web intuitiva [9]. Successivamente Telerete ha ritenuto opportuno munirsi dei prodotti sviluppati da un altro fornitore. Il mio tirocinio si è concentrato sullo studio di questi nuovi apparati, che come vedremo, hanno potenzialità maggiori.

Capitolo 2

BLUETOOTH PROXIMITY MARKETING

2.1 Introduzione

Il Proximity Marketing è una tecnica che, mediante l'utilizzo di strumenti e metodi specifici, veicola informazioni e stabilisce contatti con individui e strutture poste nelle immediate vicinanze del mittente [1]. Esempi di proximity marketing sono i cartelloni pubblicitari e le insegne luminose dei negozi. Il marketing di prossimitià *NON* è legato a doppio filo con il bluetooth, così come molti pensano erroneamente e così come è stato erroneamente scritto su Wikipedia. Il bluetooth entra in gioco come strumento di proximity marketing perché espande la prossimità geografica dell'attore e moltiplica il numero di contatti a buon fine che si riescono ad ottenere in un tempo dato. Il Bluetooth Proximity Marketing è una strategia pubblicitaria che invia messaggi e contenuti multimediali direttamente ai telefoni cellulari degli utenti presenti all'interno di una particolare area. La filosofia adottata dal Proximity Marketing sposa la necessità di arrivare sempre più in prossimità ed in contatto con il consumatore, dandogli maggiori informazioni.

2.2 Come funziona

Una soluzione di Proximity Marketing richiede la presenza di un Access Point Bluetooth che distribuisca contenuti multimediali ai telefonini presenti nell'area di copertura. Gli Access Point ricercano continuamente i cellulari degli utenti ed iniziano le trasmissioni per interagire con loro. Il consumatore riceverà un

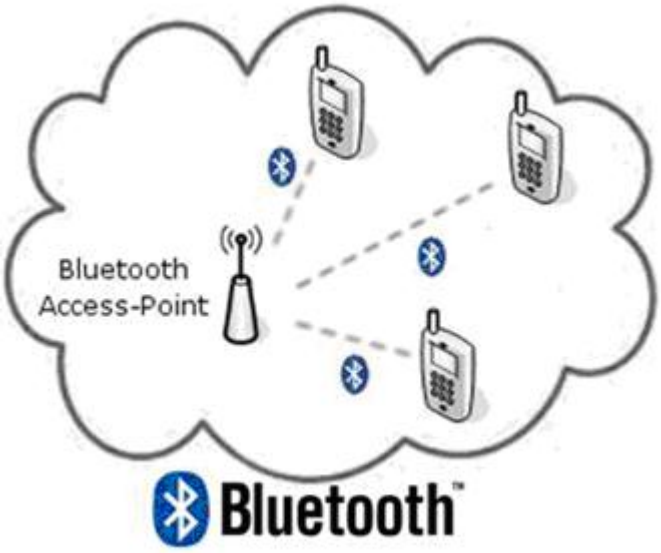

Figura 2.1: Rete Bluetooth

messaggio personalizzato di notifica sul proprio telefono, per l'autorizzazione al collegamento Bluetooth. Questo messaggio garantisce all'utente di essere avvisato prima di ricevere uno o più messaggi ed al tempo stesso lo protegge dallo Spam.

2.3 Privacy

Nell'ambito del Proximity/Bluetooth Marketing, la Comunità Europea ha emanato una serie di normative riguardanti la privacy dei potenziali bersagli della campagna in atto (art. 4 del d.lgs. 196/03 del 30 giugno 2003 [10] e News Letter Garante della privacy N. 308 del 17 giugno 2008 [11]). Nel momento in cui l'Access Point lo intercetta per l'invio del messaggio (ogni dispositivo Bluetooth, ha un identificativo unico), sia che l'utente accetti il download, sia che lo rifiuti, l'identificativo Bluetooth viene memorizzato ed entra a far parte di una graylist (che sarà, eventualmente, gestita in un server remoto). Il sistema sarà quindi poi in grado di riconoscere il cellulare Bluetooth e non tenterà di inviare nuovamente lo stesso messaggio, evitando azioni di spamming. Le informazioni registrate per questo scopo non permettono di conoscere in nessun modo l'identità del possessore, né i suoi contatti in rubrica; ciò che è registrato è solamente in codice macchina, una stringa alfanumerica generata dal produttore in base agli elementi di cui è formato il cellulare. La ricezione dei messaggi inoltre, è possibile solo se Bluetooth è attivo. É quindi necessaria un'azione pro-attiva da parte dell'utente affinché tale comunicazione sia possibile. Ogni ricezione di una tranche di

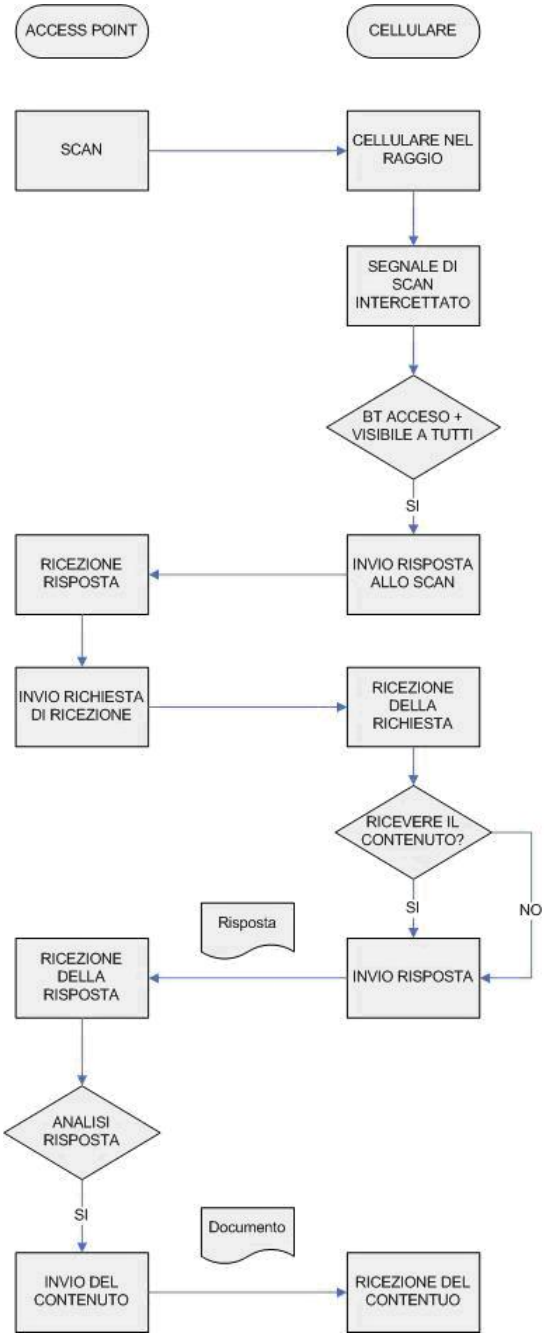

Figura 2.2: Proxy di comunicazione Bluetooth

messaggi richiede all'utente un'autorizzazione. Azione pro-attiva e richiesta di autorizzazione fanno sì che tale metodo di advertising sia pienamente rispettoso della normativa europea in tema di privacy.

2.4 Perchè usare il Bluetooth?

Le principali soluzioni di Proximity Marketing utilizzano Bluetooth come tecnologia di connettività wireless. Questa tecnologia, infatti, è particolarmente indicata per il trasferimento dati a corto raggio ed è ampiamente diffusa tra i terminali mobili degli utenti.

2.4.1 Vantaggi

Il Bluetooth è la tecnologia di connettività wireless più utilizzata nell'ambito del Proximity Marketing principalmente per i seguenti motivi:

- Efficacia
- Innovatività dello strumento
- Costi molto bassi
- Multimedialità del messaggio
- Possibilità di Interazione fra mittente e destinatario

Inoltre non è da sottovalutare che Bluetooth è la tecnologia di connettività wireless, a corto raggio, più diffusa tra i telefoni cellulari, smartphone e palmari. Secondo un indagine del 2007 oltre il 89% degli italiani ha un cellulare, che la spesa media per il download di contenuti digitali supera i 15 euro al mese [12] e che i contenuti sono offerti gratuitamente, per gli inserzionisti interessati al Proximity Marketing, l'opportunità di veicolare la pubblicità su questa soluzione in espansione è fortemente invitante

2.4.2 Efficacia

Il trasferimento di uno o più contenuti, verso un telefono cellulare o uno smartphone, richiede che l'utente accetti un messaggio di notifica che compare sullo schermo del suo dispositivo (permission-based message). Questo messaggio di notifica garantisce all'utente di essere avvisato prima di ricevere uno o più contenuti,

ed al tempo stesso lo protegge da eventuale SPAM indesiderato. In quest'ottica, l'utente riceve i contenuti se e solo se:

- Bluetooth è attivo sul dispositivo mobile
- Bluetooth è in modalità visibile a tutti o raggiungibile a seconda del modello.
- L'utente accetta il messaggio di notifica.

In genere, una strategia di Proximity Marketing dovrebbe essere adottata in particolari contesti e con le seguenti condizioni:

- Numero sufficiente di clienti dotati di cellulare con tecnologia Bluetooth (fiere, concerti, ecc..).
- Luoghi aperti o con limitate barriere fisiche (per l'ottimale trasmissione dei dati).
- Possibilità di pre-avvisare i clienti della campagna in programma in modalità audio/video.

Quest'ultima condizione è vincolante in quanto il cliente potenziale bersaglio della campagna deve fornire il suo assenso per la ricezione dei dati. Ciò avviene con più probabilità se sono state diffuse informazioni circa:

- L'identità dell'azienda committente e che ha realizzato la campagna.
- Le misure di sicurezza prese per tutelare la privacy (bisogna rassicurare i clienti che non ci sono rischi per nè la privacy nè per il supporto hardware nell'accettare il messaggio).

Infine il successo di una campagna di Proximity Marketing dipende fortemente dall'interesse che riesce a suscitare nei confronti dell'utente finale. Un contenuto multimediale, contenente un messaggio (pubblicitario o informativo), viene recapitato con successo all'utente se e solo se quest'ultimo ha i servizi Bluetooth attivati, visibile a tutti ed accetta il messaggio di notifica. In quest'ottica, è fondamentale informare l'utente della campagna in corso ed invitarlo/invogliarlo ad attivare Bluetooth ed accettare il messaggio di notifica. É di fondamentale importanza che il cliente percepisca come vantaggioso ricevere tale contenuto; infatti, è necessario convincere che la ricezione di un messaggio è un'opportunità, e non

una seccatura, i risultati miglior, infatti,i sono stati registrati dove la ricezione del messaggio forniva uno sconto o ad un gadet. Questo tipo d'informazione avviene solitamente per mezzo di cartellonistica, appositamente studiata, da installare nell'area coperta dal segnale dei trasmettitori Bluetooth.

2.5 Access point o Adattatore USB

In commercio sono disponibili adattatori USB Bluetooth, delle chiavette USB in grado di inviare contenuti attraverso questa tecnologia. Di norma, queste soluzioni hanno un costo minore, ma permettono di realizzare campagne semplici, come l'invio di un singolo messaggio. Un access point invece è un dispositivo in grado di gestire campagne tecnicamente più complesse e in grado di creare una vera e propria infrastruttura di rete capace di soddisfare esigenze più complesse rispetto alla soluzione precedente. Inoltre un access point è in grado di funzionare stand alone, ossia, una volta programmato funziona senza la necessità di essere collegato ad un computer, ed durante il mio tirocinio ho sviluppato un prototipo che non necessita l'alimentazione da rete, e che raggiunge un'autonomia di 72 ore, per ulteriri dettagli si consulti il capitolo 8.

2.6 Produttori ed Erogatori di Servizi

Nella prima fase del tirocinio mi sono occupato della ricerca di aziende che fornissero servizi basati su BT. Attraverso la distinzione tra "produttori", "erogatori di servizi", si è costruita una base di conoscenza riguardo quali fossero le possibilità e su quale fossero gli standard che il mio lavoro avrebbe dovuto raggiungere. I produttori di apparati Bluetooth che ho trovato nella mia ricerca sono 15, di cui 5 sono stati scartati per motivi legati al prezzo o alla natura del servizio offerto. A queste 10 società è stato inviato un questionario per approfondire la loro conoscenza, purtroppo la maggior parte di queste offrono solamente un servizio chiavi in mano, senza offrire la possibilità di personalizzare la macchina al livello necessario per la creazioni di applicazioni avanzate. Ciò riduce i produttori veri e propri a solamente poche aziende, che offrono servizi molto simili tra loro. Tra tutte le azide possiamo definirne tre tipi:

- Solamente Produttori
- Produttori ed Erogatori di Servizi

- Solamente Erogatori di Servizi

Nel primo gruppo ci sono le aziende di maggior interessere, ossia quelle che forniscono solamente HW e nessun servizio aggiuntivo. Si tratta di veri e propri produttori che assemblano l'access point e forniscono il firmware per poterlo utilizzare. Il livello del firmware di solito è molto basso, da una shell testuale fino a delle semplici pagine html con le quali è possibile gestire in modo molto limitato i servizi. La maggior parte di questi fornitori, a pagamento, forniscono una piattaforma per la gestione dei contenuti e della campagna per permettere anche ai non esperti di poter utilizzare features avanzate; tuttavia, dalla mia analisi, affrontare il costo della piattaforma non è un investimento corretto; infatti, non sono aggiunte vere e proprie funzioni, ma solo un'interfaccia più user-friendly. D'altra parte alcune aziende, come quella scelta daTelerete, forniscono dei tool di programmazione che permettono di usare linguaggi ad alto livello, come il C o il C++, per la programmazione di soluzioni avanzate. Tenuto conto delle limitazioni della campagna pubblicitaria, imposta dal firmware, e delle possibilità messe a disposizione dal contesto descritto, la possibilità di utilizzare un linguaggio di programmazione ad alto livello è diventato un requisito indispensabile. La possibilità di usare tali linguaggi apre a scenari del tutto inesplorati dalle precedenti esperienze realizzate; campagne veramente intelligenti in grado di interagire con i cellulari degli utenti. Alcune di questi produttori, offrono la possibilità di realizzare una brandizzazione, limitata, del prodotto; vengono offerte colorazioni diverse e l'applicazione del proprio logo aziendale sull'hardware. La licenza si basa sull'acquisto della macchina comprensiva della componente software.

Nel secondo gruppo, "Produttori ed Erogatori di Servizi", sono presenti le aziende che oltre a produrre un HW offrono le applicazioni per la gestione di tale software e la realizzazione della campagna, o del progetto di marketing. I software in questione offrono servizi e prestazioni superiori ai firmware prima descritti, e permettono la gestione di tutta la campagna anche a chi non ha una preparazione tecnologica. Solo una azienda offre la possibilità di una gestione della macchina prodotta al di fuori del suo software, mentre le altre si sono mostrate, come era logico immaginarsi, discutere di tale aspetto solo a trattative avanzate. Le licenze sono nella formula: acquisto/noleggio dell'access point, utilizzo della macchina. Infine il terzo tipo di aziende, al quale appartiene anche Teleretenordest srl, sono quelle aziende che comprano la componente HW, per poi offrire un servizio chiavi in mano al cliente. Viene offerta una realizzazione della campagna in tutti i suoi aspetti, dalla progettazione fino alla realizzazione e alla stesura di statistiche per la valutazione del successo della campagna stessa. E' stato importante investire

molto tempo studiando queste aziende per capire quale fosse il benchmark con cui confrontarsi, e possibilmente superare, per scoprire quali siano i luoghi e le esperienze più interessanti dove sviluppare una campagna che questo tipo. Ciò che è emerso è una situazione di poca propensione all'innovazione, con sviluppi di campagne molto semplici, e con un livello di interattività con il cliente nullo. Per ulteriori dettagli si consulti l'appendice.

2.7 Scenari d'utilizzo

- Fiere ed Eventi
  - Per l'organizzatore della fiera può essere interessante fornire la mappa, e dei memorandum per dei convegni che si svolgeranno. Potrebbe inoltre vendere gli spazi pubblicitari creati da questa tecnologia
  - Gli standisti potrebbero munirsi di questa tecnologia per distinguersi dagli altri ed esercitare un'operazione pubblicitaria nel breve medio raggio per catturare nuovi clienti
- Centri Commerciali: invio di messaggi riguardo all'orario d'apertura, manifestazioni, offerte promozionali di negozi, sconti o saldi.
- Pub, locali pubblici: clientela target perfetta, giovane e aperta alle nuove tecnologie, i contenuti più appetibili sono **vcalendar** per appuntamenti organizzati dal locale, coupon con sconti
- Concerti: come nel caso precedente, buona clientela alla quale viene offerto di scaricare la canzone in formato mp3, foto e video del backstage
- Negozio di giochi: sono offerti in formato java delle demo dei giochi acquistabili all'interno del negozio
- Turismo: posizionati nella hall di un hotel, in un monumento, l'access point è in grado di inviare audio guide, video, descrizioni dell'offerta turistica della città o la storia del monumento a che si sta osservando
- Museo: è già possibile realizzare una vera e propria guida virtuale attraverso l'invio di filmati o file mp3
- Stazioni di servizio e autogrill: luoghi ad altissima concentrazione di persone, possono essere inviati contenuti pubblicitari/turistici, oppure riguardo al traffico, o il meteo.

- Trasporto pubblico: è possibile richiedere gli orari di una particolare linea e in immediato futuro, interrogare l'access point su eventuali ritardi del bus che si attende
- Congressi: nei paesi asiatici è già di moda la diffusione via bluetooth di credenziali, biglietti da visita dotati di QR-Code, scaletta del congresso e tante altre informazioni utili e non volatili per l'utente finale.
- Cinema: durante l'attesa della proiezione, o in coda alla cassa è possibile riceve trailer, sfondi per il cellulare, suonerie mp3 dei film in programmazione.
- Studi medici: durante l'attesa può essere offerto una semplice applicazione ludica, alcune curiosità o altro.

2.7.1 Esempi di Proximity Marketing

Vengono ora presentati due progetti: il primo è un esempio di proximity marketing, studiato per avvicinare i clienti ad uno stand [13], mentre il secondo vuole mostrare come sia possibile utilizzare il canale Bluetooth per fornire contenuti utili e interessanti [14].

2.7.2 Stand ISUZU al Salone di Ginevra 2007

Progetto: Installazione postazione mediasender e diffusione video promozionale realizzato da Nikita s.r.l..

Obiettivo del progetto: L'obiettivo del progetto è la realizzazione di un'istallazione di Proximity Marketing via Bluetooth presso l'area ISUZU alle manifestazioni future, con lo scopo di attirare visitatori, suscitare interesse intorno al prodotto ISUZU, comunicare innovatività e creatività dei progetti, nonché valorizzare il prodotto videogioco realizzato per il Motorshow.

Sistema: L'installazione per ISUZU Italia è stata posta all'interno dello stand della manifestazione (Salone dell'auto 2007 di Ginevra); questa consisteva in un Totem ambientato nella scenografia dello stand, il quale ha funzionato da supporto per il box trasmettitore del segnale Bluetooth. A corredo all'interno dello stand, diversi cartelli avvisano l'utente che in quell'area viene spedito ai cellulare dotati di Bluetooth (abilitato) un contenuto omaggio da parte di ISUZU Italia. L'utente accetta di ricevere il messaggio e incuriosito si avvicina allo stand chiedendo informazioni, stabilendo così un contatto volontario e interessato con il personale ISUZU presente in loco.

Fase 1

L'antenna inizia l'invio del contenuto multimediale; in primo luogo trova tutti i cellulari nel suo raggio d'azione.

Fase 2

Il visatatore si aggira nei pressi dello stand, tra gli spazi espositivi.

Fase 3

Il cellulare viene catturato dal segnale, e il visitatore, previa una accettazione volontaria, riceve il messaggio pubblicitario.

Fase 4

Il processo fin qui descritto viene eseguito su diversi cellulari; attraverso un effetto domino si crea un flusso di persone

Fase 5

A questo punto lo spazio espositivo è frequentato da persone interessate al prodotto.

2.7.3 Bluemotors

Progetto: installazione di un sistema di access point nel circuito di Aci Vallelunga realizzato da Aci Informatica e Cyber Computer.

Obiettivo del progetto: Inviare messaggi multimediali sui telefonini del pubblico e degli addetti ai lavori presenti negli eventi del polifunzionale impianto di Campagnano di Roma. I messaggi istituzionali riguardano il calendario degli eventi, l'agenda, le classifiche e i tempi, le foto.

Sistema: Il sistema è costituito da una serie di access point distribuiti nell'area dell'autodromo collegati attraverso una rete LAN e comandati attraverso un CMS. Il sistema viene alimentato con i dati della telemetria e con contenuti caricati prima e durante il gran premio.

2.7.4 Perché il Bluetooth non è spam

Ad occhi inesperti può sorgere il dubbio che il Proximity Marketing non sia che un altro modo per fare Spam selvaggio, ma così non è. Innanzitutto che cosa è lo SPAM? Lo Spam è l'invio indifferenziato ed illimitato di messaggi a carattere prevalentemente pubblicitario e si attua tramite posta elettronica, newsgroup, link non attinenti inseriti in blog, forum, directory (tecnica chiamata Spam linking) [2]. Lo Spam si caratterizza per:

1. L'invio di massa. Praticare SPAM significa inviare in contemporanea, centinaia e centinaia di messaggi ad inconsci soggetti che inconsapevolmente si ritrovano le proprie caselle di posta elettronica ma anche directory e forum intasati da messaggi indesiderati. Nello SPAM il destinatario del messaggio, di fatto inconsapevole, è dunque un soggetto passivo perchè non ha la possibilità di scegliere se ricevere o meno quel messaggio. Salvo quindi che non abbia preventivamente adottato misure idonee a filtrare i messaggi indesiderati (filtri antiopa) si vede costretto ad inutili perdite di tempo per la loro cancellazione. Il problema poi è forse ancor più grave quando i messaggi spam sono recapitati nella propria casella di posta elettronica che seppur virtuale è pur sempre un luogo strettamente privato e personale che di forza è violato. Da qui l'invasività.

2. L'anonimato. I messaggi SPAM nella quasi totalità delle volte sono ANONIMI ossia inviati con fantasiosi nickname che non permettono in alcun modo di risalire al vero mittente.

3. La decontestualizzazione. I messaggi SPAM sono del tutto decontestualizzati e risultano totalmente privi di interesse per il ricevente nel caso di posta elettronica o non attinenti agli argomenti trattati in caso di link spamming. Di qui il termine spazzatura.

4. Disponibilità di liste di nominativi. É ormai noto quanto diffusa sia la pratica non del tutto legale di raccogliere indirizzi email per comporre lunghissime liste di nominativi vittime di questa odiosissima pratica

Caratteristiche diametralmente opposte al marketing di prossimità. Certo se si pensa ad un congresso con centinaia di persone munite di telefono cellulare a sua volta dotato, nella maggior parte dei casi, di tecnologia bluetooth, possiamo anche in questo caso parlare di invio di massa a potenziali fruitori del nostro servizio ma in questo caso i presupposti sono completamente diversi e basati su:

1. Identificazione del mittente. Il marketing di prossimità prevede l'invio di messaggi che non sono assolutamente anonimi: ogni messaggio inviato ha un mittente chiaro ben definito ed identificabile con un ente promotore

2. Richiesta di accettazione. Quando si invia un messaggio nell'ambito di una campagna di prossimità avvalendosi del bluetooth, il destinatario è raggiunto da una domanda di ricezione messaggio del tipo "Vuoi ricevere un messaggio da nomeazienda?" In questo momento il destinatario ha la piena consapevolezza delle sue azioni: diventa attore attivo dell'iniziativa in corso potendo scegliere se rifiutare il messaggio oppure accettarlo e cominciare ad interagire. Ricerche hanno dimostrato che, fermo restando l'iniziale predisposizione positiva dovuta all'identificazione chiara del mittente che in un certo qual modo abbassa le naturali barriere difensive proprie di ogni individuo, quanto più il mittente è famoso ossia è un noto brand, tanto più si è orientati ad accettare il messaggio e dare un seguito all'iniziale segnale di ricezione.

3. Attinenza o contestualizzazione. Quando si inviano messaggi bluetooth in una campagna di marketing di prossimità i contenuti non sono mai decontestualizzati, anzi al contrario, una campagna di marketing di prossimità nasce proprio per sostenere eventi o iniziative in corso in un determinato contesto circoscritto, in cui chi è presente ha tutto l'interesse a ricevere quel tipo di messaggio.

Capitolo 3

BLUETOOTH

3.1 Introduzione

Bluetooth è una specifica industriale per reti personali senza fili (WPAN: Wireless Personal Area Network). Fornisce un canale di comunicazione, economico e sicuro per scambiare informazioni tra dispositivi diversi attraverso una frequenza radio sicura a corto raggio. Tale sistema prevede la ricerca di dispositivi entro un raggio di qualche decina di metri; tali dispositivi coperti dal segnale, sono messi in comunicazione tra di loro. I dispositivi, come detto, possono essere, ad esempio, palmari, telefoni cellulari, personal computer, portatili, stampanti, fotocamere digitali, console per videogiochi.

3.2 Aspetti Generali

Questo standard è stato progettato con l'obiettivo primario di ottenere bassi consumi, un corto raggio di azione e un basso costo di produzione per i dispositivi compatibili. Lo standard doveva consentire il collegamento wireless tra periferiche come stampanti, tastiere, telefoni, microfoni, ecc. a computer o PDA o tra PDA e PDA. I telefoni cellulari che integrano chip Bluetooth sono venduti in milioni di esemplari e sono abilitati a riconoscere e utilizzare periferiche Bluetooth in modo divincolarsi dai cavi. BMW è stato il primo produttore di autoveicoli a integrare tecnologia Bluetooth nelle sue automobili in modo da consentire ai guidatori di rispondere al proprio telefono cellulare senza dover staccare le mani dal volante [15].

3.2.1 Cenni Storici

La storia di questo appellativo ci rimanda indietro nei secoli. Harald Bluetooth era il nome di un antico re vichingo, vissuto tra il 940 e il 981 D.C.. L'impresa di Harald Bluetooth fu l'unione di due paesi tanto diversi tra loro, come Danimarca e Norvegia, sotto un unico regno, con lui a capo. Inoltre Harald portò a termine la completa cristianizzazione dei due paesi. In tal modo il nome Bluetooth è rimasto a significare l'unione di due realtà diverse, ma con la volontà di collaborare per un futuro migliore [2] [1]. Così Bluetooth è stato preso come simbolo per la nuova tecnologia, che porterà a unire differenti dispositivi, anche molto diversi tra loro, in un'unica grande rete senza fili. Anche il simbolo del Bluetooth rimanda all'antico re vichingo.

Figura 3.1: Logo Bluetooth

Consiste, infatti, nei due simboli celtici raffiguranti la H e la B, le iniziali di Harald Bluetooth. Le basi dell'odierno Bluetooth le getta Ericsson nel 1994. L'azienda svedese iniziò una ricerca per trovare una nuova interfaccia di comunicazione a onde radio a basso costo, tra i cellulari e i diversi accessori, che soppiantasse l'IrDA. L'idea era quella che un minuscolo ricevitore radio immesso sia nel cellulare sia nell'eventuale dispositivo da unire, avrebbe sostituito i vari fili usati all'epoca. Dopo circa un anno dal progetto iniziale, lo sviluppo della nuova tecnologia entrò nella fase operativa e il lavoro passò alla sezione ingegneristica dell'Ericsson. Il progetto iniziale, di fornire nuovi accessori per telefonia mobile a onde radio, fu soppiantato da un progetto più ambizioso, la creazione di una vera e propria nuova tecnologia per far comunicare qualsiasi tipo di dispositivo a breve distanza. Ericsson capì l'importanza degli studi effettuati e decise di allargare il gruppo di sviluppo ad altri partner. Si arriva quindi al 1998, con la formazione del SIG (Special Interest Group) insieme a Nokia, Intel, IBM e Toshiba [16].

Inizialmente Ericsson ha fornito un importante contributo per quanto riguarda la tecnologia radio. Toshiba e IBM hanno lavorato per sviluppare un protocollo comune per l'integrazione del Bluetooth all'interno di dispositivi portatili. Nokia si è occupata della trasmissione dati e del software e Intel della progettazione dei nuovi chip necessari.

Gli obiettivi iniziali del gruppo erano aiutare lo sviluppo di una nuova tecnologia a onde corte per far comunicare a breve distanza, creando uno standard aperto a tutte le aziende che ne volessero far parte. Già nel mese di aprile del 1999 il consorzio contava ben 600 membri. Nel mese di luglio dello stesso anno uscirono le prime specifiche tecniche del neonato Bluetooth. Da quel momento varie versioni del Bluetooth sono state ratificate dal SIG, tutte rispondenti ai requisiti d'interoperabilità e armonizzazione della banda. Obiettivo principale del SIG è appunto garantire il perfetto funzionamento di apparati Bluetooth costruiti da differenti aziende.

L'interoperabilità dei dispositivi Bluetooth è il punto di partenza per garantire il successo della nuova tecnologia; è quindi ovvio che sia il primo punto d'interesse del SIG. Inoltre è stata decisa l'adozione dei Profili, ossia una serie di differenze sulla produzione di dispositivi dotati di Bluetooth ad interagire tra loro. Ad esempio difficilmente una stampante dovrà interagire con un monitor, di conseguenza i Profili da inserire in ambo i prodotti si riducono ed aiutano a semplificarne lo sviluppo.

3.2.2 Frequency-Hopping Spread Spectrum

La tecnologia Bluetooth è basata su Frequency- Hopping Spread Spectrum, si tratta di una tecnica di trasmissione radio usata per aumentare la larghezza di banda di un segnale; consiste nel variare la frequenza di trasmissione a intervalli regolari in modo pseudo casuale attraverso un codice prestabilito.

L'unico modo di ricevere correttamente la trasmissione è di conoscere la sequenza esatta dei salti di frequenza e disporre di un ricevitore adatto a seguirli; altrimenti ciò che si ottiene sono solo dei frammenti sparsi senza alcun significato coerente. Questo porta a due capacità fondamentali della FHSS:

1. Un certo grado di segretezza della trasmissione

2. Una buona immunità ai disturbi, soprattutto da parte di altre trasmissioni interferenti

Inoltre la FHSS consente una buona resistenza al multipath fading perché il ricevitore ottiene per primo il segnale diretto; i segnali riflessi seguono un percorso più lungo e quando arrivano finalmente al ricevitore, quest'ultimo ha già effettuato il salto di frequenza, in questo modo i segnali riflessi non sono presi in considerazione e non possono interferire nella comunicazione.

3.2.3 Tipologia di rete – Piconet

Si dice Piconet una rete formata da dispositivi collegati tramite Bluetooth

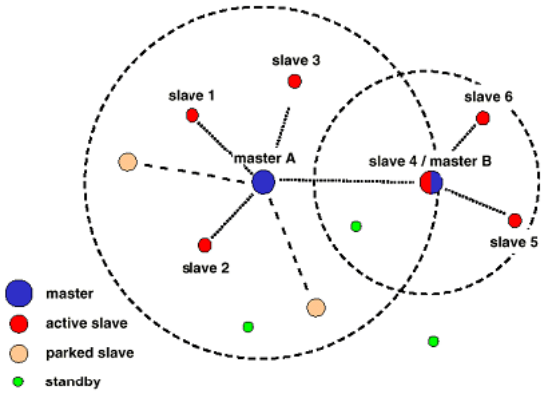

Figura 3.2: Piconet Bluetooth

Per stabilire una connessione tra due (o più) dispositivi, i dispositivi devono prima selezionare un canale e successivamente collegarsi a esso. Tale processo è denominato come "forming a piconet", creare una piconet. Le Piconet si servono della topologia Master/Slave: ogni piconet ha un master e uno o più slave. Un dispositivo Bluetooth può simultaneamente partecipare a più piconet, o come master o come slave. Un dispositivo non può, però, essere master in più di una piconet. Ciò è dovuto al fatto che l'indirizzo hardware del master definisce il canale Bluetooth, e quindi la piconet. Un gruppo di piconet, tra le quali esiste una connessione, è chiamato scatternet.

3.3 Aspetti Tecnici

Similmente all'architettura OSI, Bluetooth specifica un approccio a livelli nella sua struttura protocollare. Differenti protocolli sono utilizzati per differenti applicazioni. Indipendentemente dal tipo di applicazione, però, lo stack protocollare Bluetooth porta sempre all'utilizzo dei livelli data-link e fisico. Non tutte le appli-

cazioni usano tutti i protocolli dello stack Bluetooth; infatti, esso è rappresentato su più livelli verticali, al di sopra dei quali c'è un'applicazione specifica.

3.3.1 Architettura OSI

La figura 3.3 rappresenta l'architettura di rete OSI (Open System Interconnection), la quale definisce una suddivisione delle funzionalità di rete in sette livelli, con uno o più protocolli che realizzano il funzionamento assegnata a un dato livello.

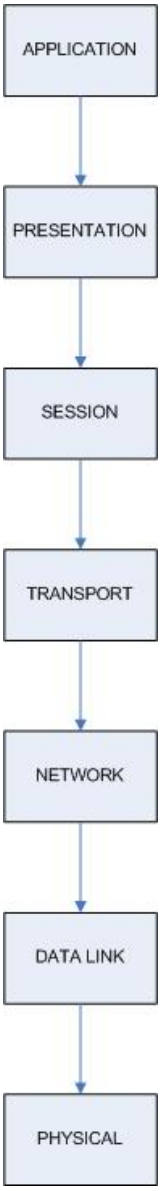

Figura 3.3: Architettura OSI

Physical
Il livello fisico definisce aspetti elettrici e meccanici, relativi l'interfacciamento verso uno specifico media fisico di trasmissione dei dati. L'implementazione di questo livello include, oltre alla costruzione dell'hardware di rete specifico, lo sviluppo dei driver software che consentono il controllo del dispositivo di comunicazione.

Data Link
Il livello di collegamento ha il compito di gestire un circuito di comunicazione tra due punti e di garantire una comunicazione priva di errori. Ha quindi i compiti di gestire l'accesso al canale fisico, creare una connessione, verificare l'integrità dei messaggi ricevuti, eseguire il framing (suddivisione dei messaggi in blocchi di lunghezza standard), garantire la sequenza dei dati trasmessi.

Network
Il livello di rete indirizza messaggi, stabilisce il percorso tra i nodi di comunicazione, nel caso nodo sorgente e destinazione non siano adiacenti, instrada i messaggi attraverso nodi intermedi, controlla il flusso di messaggi tra i nodi.

Transport
Il livello di trasporto fornisce il controllo end-to-end di una sessione di comunicazione. Una volta determinato il percorso, il livello di trasporto ha il compito di garantire uno scambio di dati affidabile (affidabile a livello di singolo messaggio) e consequenziale, indipendentemente dai sistemi comunicanti coinvolti e dalla loro ubicazione in rete.

Session
ll livello di sessione controlla aspetti dipendenti dalla sessione di comunicazione (insiemi complessi di messaggi), osservando i servizi offerti dal livello di trasporto e le funzioni logiche dipendenti dal sistema operativo ospitante.

Presentation
Il livello di presentazione traduce e converte dati scambiati, secondo formati comprensibili per gli utenti

Application
Il livello di applicazione supporta funzionalità utente specifiche e applicazioni software complesse quali condivisione di risorse, trasferimento di file ed accesso a database via rete.

3.3.2 Bluetooth Stack Protocol

Similmente all'architettura OSI, Bluetooth specifica un approccio a livelli nella sua struttura protocollare. Differenti protocolli sono utilizzati per differenti ap-

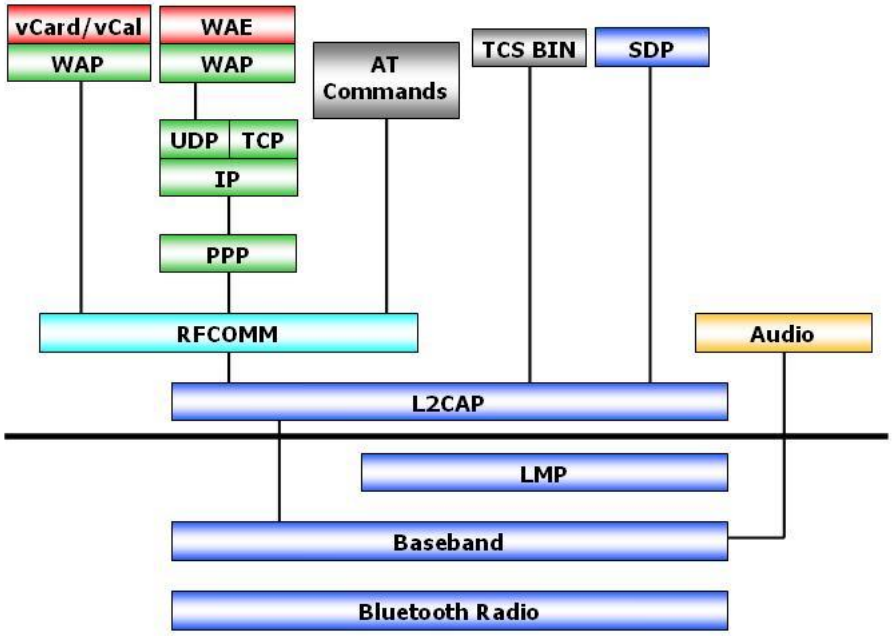

Figura 3.4: Bluetooth Protocol Stack

plicazioni. Indipendentemente del tipo di applicazione, però, lo stack protocollare Bluetooth porta sempre all'utilizzo dei livelli data-link e fisico. Non tutte le applicazioni usano tutti i protocolli dello stack Bluetooth, infatti, esso è rappresentato su più livelli verticali, al di sopra dei quali c'è un'applicazione specifica.

Bluetooth Radio

Lo strato Bluetooth Radio è lo strato più basso della specifica Bluetooth. Definisce i requisiti del dispositivo Bluetooth che opera nella banda ISM di 2.4GHz.

Caratteristiche del trasmettitore

I dispositivi sono divisi in tre classi:

1. Classe 1: 20 dBm max power output, range 100m
2. Classe 2: 4 dBm max power output, range 10m
3. Classe 3: 0 dBm max power output, range 10cm

Il modulo radio del dispositivo Bluetooth usa una modulazione GFSK (Gaussian-Frequency Shift Keying).

Caratteristiche del ricevitore

Il ricevitore deve avere un livello di sensibilità per il quale il bit-error rate sia pari a 0.1

3.3.3 Bluetooth Baseband

Lo strato Baseband corrisponde allo strato Physical dell'architettura OSI. Gestisce il canale fisico e fa da collegamento per altri servizi come la correzione d'errore, selezione della sequenza di hopping e Bluetooth security. Il protocollo Baseband è implementato come Link Controller, il quale lavora con il link manager per effettuare routine di livello come connessione e controllo di potenza. Inoltre, gestisce pacchetti, collegamenti sincroni e asincroni e ricerca i dispositivi Bluetooth nell'area.

Canale fisico

Il trasmettitore Bluetooth opera nella banda disponibile (globalmente) di 2400 MHz ISM (Industrial Scientific Medicine). In molti paesi il range di tale banda di frequenza è 2400.0 - 2483.5 MHz. La sequenza di hop è unica per ogni piconet ed è determinata dal Bluetooth device address (BD_ADDR) del master; la fase nella sequenza di hop è determinata dal clock Bluetooth del master. Il canale è diviso in time-slot, dove ogni slot corrisponde ad una frequenza di hop RF.

Collegamento fisico

Baseband gestisce due tipi di collegamento: SCO (Synchronous Connection- Oriented) e ACL (Asynchronous Connection-Less).

Canali logici

Bluetooth ha cinque canali logici che possono essere usati per trasferire informazioni differenti. I canali LC (Link Controller) e LM (Link Manager) sono usati nel livello di collegamento, mentre i canali UA (User Asynchronous), UI (User Isochronous) e US (User Synchronous) sono usati per trasmettere informazioni asynchronous, isosynchronous e synchronous. Quattro possibili tipi di indirizzo possono essere assegnati alle unità Bluetooth:

- BD_ADDR: Bluetooth Device Address. Ad ogni dispositivo è assegnato un indirizzo unico di 48 bit. Questo è diviso in un campo LAP (Lower Address Portion) di 24 bit un campo NAP (Non-signficants Address Portion) da 16 bit e un campo UAP (Upper Address Portion) da 8 bit

- AM_ADDR: Active Member Address. É un numero da 3 bit ed è valido solo finchè lo slave è attivo nel canale

- PM_ADDR: Parked Member Address. É un member address di 8 bit che distingue gli slave in modalità PARK. Il PM_ADDR è valido solo finché lo slave è in park mode.

72 bits	54 bits	0-2745 bits
Access Code	Header	Payload

Tabella 3.1: Pacchetto Bluetooth

8 o 16 bits	0-2712 bits	16 bits
Payload Header	Payload	Data CRC

Tabella 3.2: Dettaglio del Payload

- AR_ADDR: Access Request Address. É utilizzato da un parked slave per determinare lo slot nella finestra d'accesso nel quale è permesso inviare un messaggio d'access request. É valido solo finché lo slave è in parked mode e non è necessariamente unico.

Tipo di pacchetto

Sono definiti 13 tipi diversi di pacchetto (per lo strato Baseband). Tutti gli strati più alti usano questi pacchetti per comporre PDU (Protocol Data Unit) di alto livello. I pacchetti sono ID, NULL, POLL, FHS, DM1; questi sono definiti per entrambi i collegamenti SCO e ACL. DH1, AUX1, DM3, DH3, DM5, DH5 sono definiti solo per collegamenti ACL. HV1, HV2, HV3, DV sono definiti solo per collegamenti SCO.

Pacchetti

Questa sezione introduce la struttura del pacchetto Bluetooth e i differenti tipi di data packet (pacchetti per trasmissione dati) usati per la comunicazione in un collegamento ACL. I pacchetti sono formati da: access code, packet header, payload header e payload. Attualmente sono definiti 4 tipi di pacchetto: Packets (sia ACL sia SCO), Single slot, ACL3 e ACL4. Ogni tipo di pacchetto ha un diverso livello di correzione d'errore, protezione e differente grandezza del payload. Il formato del pacchetto generico è diviso in 3 segmenti L'Access Code è usato per scoprire la presenza di un pacchetto e indirizzarlo verso un dispositivo specifico. Il campo Header Packet contiene informazioni di controllo associate al pacchetto, come, ad esempio, l'indirizzo dello slave al quale è destinato il pacchetto stesso. Infine, Payload contiene l'informazione del messaggio. Il campo Payload di tutti i pacchetti ACL è diviso in Payload Header, Payload Data e il campo Cyclic Redundancy Check (CRC); lo si può vedere nella tabella 3.2.

Link Manager Protocol (LMP)

Effettua il setup del collegamento, autenticazione, configurazione del collegamento e altri protocolli. Scopre altri LM remoti e comunica con loro tramite il Link Manager Protocol (LMP). Per eseguire il suo ruolo di service provider, il LM fa uso dei servizi del sottostante Link Controller (LC). Sostanzialmente, il Link Manager Protocol è formato da un certo numero di PDU (protocol data units) trasmessi da un dispositivo ad un altro, determinato da AM_ADDR presente nel campo header del pacchetto. I LM-PDU sono sempre spediti come pacchetti single-slot e l'header del payload è quindi un byte. Pacchetti DM1 sono usati per trasportare LM-PDU ad eccezione del caso in cui un collegamento SCO, che usa pacchetti HV1, sia presente e la lunghezza del contenuto sia inferiore di 9 byte. In questo caso vengono usati pacchetti DV. Segue una lista di LM-PDU. Questi possono essere obbligatori (Mandatory) o opzionali (Optional).

General Response

(M) LMP_accepted, LMP_not_accepted

Questi PDU sono usati come messaggi di risposta ad altri PDU in molte procedure differenti

Authentication

(M) LMP_au_rand, LMP_sres

La procedura di autenticazione è basata su uno schema challenge-response. Il verifier spedisce un messaggio LMP_au_rand, che contiene un numero casuale (detto challenge), al richiedente. Il richiedente calcola una risposta, la quale e una funzione del challenge, del BD_ADDR del richiedente e una chiave segreta. La risposta viene spedita al verifier, il quale controlla se la risposta è corretta. Il calcolo corretto della risposta di autenticazione richiede che due dispositivi condividano una chiave segreta.

Encryption

((O) LMP_encryption_mode_req,LMP_encryption
_key_size_req, LMP_start_encryption_req, LMP_stop_encryption_req

Se è stata eseguita almeno un'autenticazione, allora si possono spedire messaggi criptati. Se il master vuole che tutti gli slave della piconet usino gli stessi parametri di cifratura, deve diffondere una chiave temporanea (K master) e far diventare tale chiave la chiave di collegamento (link key) corrente per tutti gli slave nella piconet, prima che inizi la cifratura

LMP Version

Il layer LMP prevede richieste sulla versione del protocollo LM. Il dispositivo che ha ricevuto la richiesta risponde con tre parametri: VersNr, CompId e

Sub-VersNr.

- VersNr specifica la versione di LMP di cui il dispositivo è fornito.
- CompId è usato per tener traccia di possibili problemi con i layer Bluetooth più bassi. Tutte le compagnie che creano un'unica implementazione del Link Manager hanno il loro CompId.
- La stessa compagnia è anche responsabile dell'amministrazione e della tutela del Sub-VersNr. Inoltre, è raccomandato che ogni compagnia debba avere un unico SubVersNr per ogni implementazione di RF/BB/LM.

Name Request
(M) LMP_detach
LMP supporta la name request (richiesta del nome) di un altro dispositivo Bluetooth. Il nome è associato al dispositivo ed è formato da un massimo di 248 byte codificati secondo lo standard UTF-8. Il nome viene diviso in uno o più pacchetti DM1.

Detach
(M) LMP_detach
La connessione tra due dispositivi Bluetooth può essere chiusa, in qualsiasi momento, dal master o dallo slave. Un parametro (che ne sta ad indicare il motivo) viene incluso nel messaggio per informare gli altri utenti sul "perchè la connessione sia stata chiusa".

Sniff Mode
(O) LMP_sni_req, LMP_unsni_req
Per entrare in modalità sniff, master e slave negoziano un intervallo di sniff, T_sniff, e un offset, D_sniff. L'offset determina la durata del primo sniff slot; poi, gli altri sniff slot seguono ad intervalli periodici di periodo T_sniff. Quando il link è in modalità sniff, il master può solo iniziare una trasmissione nello sniff slot. Due parametri controllano l'attività d'ascolto dello slave. Il parametro sniff_attempt determina per quanti slot temporali lo slave deve restare in ascolto, altrimenti non riceverà il pacchetto con il suo AM_ADDR. Il parametro sniff timeout determina per quanti slot addizionali lo slave deve restare in ascolto se continua a ricevere solo pacchetti contenenti il suo AM_ADDR.

Park Mode
(O) LMP_park_req, LMP_unpark_PM_ADDR_req, LMP_unpark_BD_ADDR_req, LMP_set_broadcast_scan_window, LMP_modify_beacon
Se uno slave non richiede di far parte del canale, ma vuole rimanere sincronizzato

(FHsynchronized), può essere posto in park mode. In questa modalità, il dispositivo rivela il proprio AM_ADDR e si re-sincronizza svegliandosi all'istante detto di beacon (segnalazione) alla fine dell'intervallo di beacon. L'intervallo di beacon, un offset di beacon e un flag (che indica come calcolare il primo istante di beacon) determinano il primo beacon instant. Dopo di questo, gli istanti di beacon si susseguono periodicamente agli istanti predeterminati dall'intervallo di beacon. All'istante di beacon, lo slave in park mode può essere riattivato dal master. Tutti i PDU spediti dal master agli slave in park mode, vengono trasmessi in broadcast. Quando uno slave si trova in park mode, gli viene assegnato un PM_ADDR unico, che verrà usato dal master per cambiare lo stato dello slave (in park mode).

Power Control

(M) LMP_max_slot, LMP_max_slot_req

Il numero di slot (temporali) utilizzati da un dispositivo può essere limitato. Un dispositivo permette ad un altro remoto l'utilizzo di un numero massimo di slot trasmettendo il PDU LMP_max_slot (indicando il numero massimo di slot come parametro). Ogni dispositivo può richiedere di utilizzare un numero massimo di slot trasmettendo il PDU LMP_max_slot_req fornendo il numero massimo di slot come parametro. Dopo una nuova connessione, come risultato di una procedura di page scan, masterslave switch, unpark, il valore di default è di 1 slot. Vengono usati due PDU per il controllo di pacchetti multi-slot. Questi PDU possono essere trasmessi in qualunque momento, dopo, però, che sia stato completato il setup della connessione.

Link Supervision

(M) LMP_supervision_timeout

Ogni link Bluetooth è provvisto di un timer, utilizzato per la supervisione del link stesso. Il timer viene utilizzato per rilevare i cosiddetti link loss causati da dispositivi che si spostano fuori dal range d'azione, o da dispositivi che terminano la comunicazione (power-down), o altri casi simili. Una procedura LMP viene usata per settare il valore del supervision timeout.

Connection Establishment

(M) LMP_host_connection_req, LMP_setup_complete

Quando un dispositivo vuole creare una nuova connessione (Page State) che coinvolge strati che stanno sopra il LM, trasmette il messaggio LMP_host_connection_req. Quando il dispositivo, dall'altra parte, riceve tale messaggio, l'host viene informato della, incoming connection. Il dispositivo remoto può accettare o rifiutare la richiesta, trasmettendo LMP_accepted o LMP_not_accepted. Se

LMP_host_connection_req viene accettata, allora vengono invocate procedure LMP di sicurezza (pairing, authentication, encryption). Se un dispositivo non invoca alcuna procedura di sicurezza durante l'instaurazione della connessione, allora questo trasmette il messaggio LMP_setup_complete. Quando entrambi i dispositivi hanno trasmesso il messaggio LMP_setup_complete, il primo pacchetto può essere spedito, in un canale logico che non sia LMP.

Test Mode

LMP ha dei PDU per testare la certificazione (certification) e la conformità (compliance) di Bluetooth Radio e Bluetooth Baseband..

Host Controller Interface (HCI)

HCI mette a disposizione un'interfaccia di comandi per il baseband controller e il link manager, oltre all'accesso allo stato dell'hardware e ai registri di controllo. In sostanza, questa interfaccia fornisce un metodo uniforme per l'accesso alle potenzialità del Bluetooth baseband.

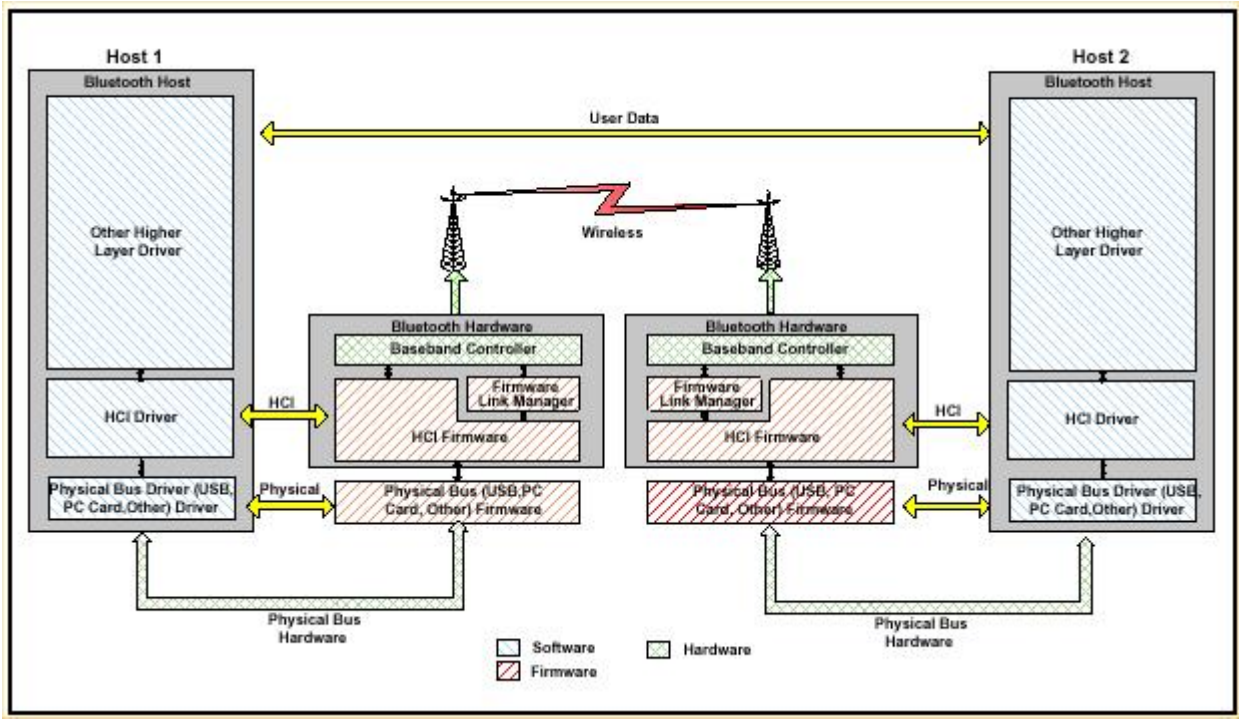

Figura 3.5: HCI Functional Entities

HCI Functional Entities

Divisione in tre parti:

1. HCI Firmware (location: Host Controller)
   HCI Firmware implementa i comandi HCI Commands per l'hardware Bluetooth, tramite l'accesso ai baseband commands, link manager, registri di stato hardware e registri di controllo.

2. HCI Driver (location: Host)
   HCI Driver, situato nell'host (es. entità software). L'host riceve una notifica asincrona di HCI events; questi sono usati per informare l'host quando sta "accadendo qualcosa". Quando l'host si accorge che ha avuto luogo un evento, analizza l'event packet per determinare quale sia l'evento.

3. Host Controller Transport Layer (location: Intermediate Layers)
   HCI Driver e Firmware comunicano tramite l'Host Controller Transport Layer, cioè una definizione di alcuni layer che ci possono essere tra l'HCI Driver nell'host system e l'HCI Firmware nell'hardware Bluetooth. Questi strati intermedi, l'Host Controller Transport Layer, forniscono la capacità di trasferire dati senza sapere niente sui dati trasferiti. Diversi HCL possono essere usati, tre dei quali sono stati inizialmente definiti per il Bluetooth: USB, UART, RS-232. L'host riceve una notifica (asincrona) di HCI events indipendentemente dall'HCTL usato.

HCI Commands

HCI mette a disposizione un metodo uniforme per l'accesso alle potenzialità hardware del dispositivo Bluetooth. Gli HCI Link Commands permettono all'host di controllare il Link Layer Connections. Questi comandi coinvolgono il Link Manager (LM) per scambiare comandi LMP con dispositivi Bluetooth remoti. Sono di sette tipologie diverse:

1. HCI-Specific Information Exchange
   L'Host Controller Transport Layer prevede uno scambio trasparente di informazioni specifiche HCI. Questi meccanismi di trasporto permettono all'host di trasmettere all'Host Controller comandi HCI, dati SCO (Synchronous Connection Oriented) e dati ACL(Asynchronous ConnectionLess). Inoltre, tali meccanismi permettono all'host di ricevere HCI Events, dati di tipo SCO e di tipo ACL, dall'Host Controller. Poiché l'Host Controller Transport Layer prevede uno scambio trasparente di informazioni specifiche HCI, le specifiche HCI precisano il formato dei comandi events e dello scambio dati tra l'host e l'Host Controller.

2. Link Control Commands
   I comandi Link Control permettono all'Host Controller di controllare la connessione con altri dispositivi Bluetooth. Quando i comandi Link Control vengono invocati, il Link Manager controlla come le piconet e le scatternet sono state costituite e come vengono mantenute. Questi comandi informano il Link Manager di creare e modificare le connessioni (del Link layer) con altri dispositivi Bluetooth remoti, eseguire procedure di Inquiry per trovare altri dispositivi nel range d'azione, e altri comandi LMP.

3. Link Policy Commands
   I comandi Link Policy forniscono dei metodi, all'host, per influenzare il modo in cui il Link Manager gestisce la piconet. Quando i comandi Link Policy vengono invocati, il Link Manager controlla ancora come le piconet e le scatternet sono state costituite e come vengono mantenute, in dipendenza di policy parameters (parametri di policy) regolabili/modificablii. Tali comandi modificano il comportamento del Link Manager, che può sfociare in cambiamenti sulle connessioni, del Link layer, con gli altri dispositivi Bluetooth remoti.

4. Host Controller & Baseband Commands
   I comandi Host Controller & Baseband forniscono l'accesso e il controllo di varie capacità dell'hardware Bluetooth. Tali parametri permettono il controllo di dispositivi Bluetooth e delle capacità dell'Host Controller, Link Manager e Baseband.

5. Informational Parameters
   I parametri informativi sono fissati dal costruttore dell'hardware Bluetooth. Questi parametri danno informazioni sul dispositivo e sulle capacità dell'Host Controller, Link Manager e Baseband

6. Status Parameters
   L'Host Controller modifica tutti i parametri di stato. Questi parametri forniscono informazioni sullo stato corrente dell'Host Controller, Link Manager e Baseband. Il dispositivo host non può modificare tali parametri, oltre a resettare certi parametri specifici.

7. Testing Commands
   I comandi di test (Testing Commands) vengono usati per fornire la possibilità di testare varie funzionalità dell'hardware Bluetooth.

Bluetooth-defined Host Controller Transport Layers

1. UART Transport Layer
   L'obiettivo dell'HCI UART Transport Layer è di rendere possibile l'uso del Bluetooth HCI attraverso un'interfaccia seriale tra due UART nello stesso PCB. L'HCI Transport Layer assume che la comunicazione UART sia libera da errori di linea. Gli eventi e i pacchetti dati passano attraverso questo layer, ma il layer non li decodifica.

2. RS-232 Transport Layer
   L'obiettivo dell'HCI RS-232 Transport Layer è di rendere possibile l'uso del Bluetooth HCI attraverso un'interfaccia fisica RS-232 tra il Bluetooth Host e il Bluetooth Host Controller. Gli eventi e i pacchetti dati passano attraverso questo layer, ma il layer non li decodifica.

3. USB Transport Layer
   L'obiettivo dell'Universal Serial Bus (USB) Transport Layer è di poter usare un'interfaccia hardware USB per l'hardware Bluetooth (realizzata in due modi: o come adattatore Bluetooth o integrata nella scheda madre di un PC portatile). È usata una classe di codici specifica per tutti i dispositivi Bluetooth USB. Ciò fa sì che sia caricato il driver stack più adatto.

Logical Link Control and Adaptation Protocol (L2CAP)

Il Logical Link Control and Adaptation Layer Protocol (L2CAP) è situato sopra il Baseband Protocol e risiede nello strato Data Link. L2CAP fornisce servizi dati connection-oriented e connectionless a protocolli di strati superiori, con operazioni di segmentazione e riassemblaggio. L2CAP permette, a protocolli e applicazioni di alto livello, trasmissione e ricezioni di pacchetti dati, aventi lunghezza fino a 64kB.

L2CAP Functional Requirements

L2CAP supporta diversi requisiti importanti di protocollo, che sono ora elencati:

1. Protocol Multiplexing
   L2CAP deve essere in grado di distinguere i protocolli di strati superiori, come il Service Discovery Protocol, RFCOMM e Telephony Control.

2. Segmentation & Reassembly
   A confronto con altri media fisici (wired - cablati), i pacchetti definiti dal

Baseband Protocol hanno grandezza limitata. Ciò limita l'uso efficiente della larghezza di banda per i protocolli di strati superiori, ideati per pacchetti più grandi. I pacchetti più estesi devono necessariamente essere segmentati in molteplici pacchetti Baseband più piccoli. Analogamente, più pacchetti Baseband, ricevuti, possono essere riassemblati in un unico pacchetto L2CAP, seguendo una semplice procedura di "integrity check". Segmentazione e Riassemlaggio (SAR) sono assolutamente necessari in presenza di protocolli che utilizzano pacchetti di dimensioni superiori rispetto alle dimensioni supportate dal Baseband.

3. Quality of Service (QoS)
   Il processo di instaurazione della connessione permette lo scambio di informazioni riguardanti la qualità del servizio (QoS) attesa tra due unità Bluetooth. Ogni implementazione di L2CAP deve monitorare le risorse utilizzate dal protocollo e assicurarsi che i contratti di QoS siano onorati.

L2CAP General Operation

1. Identificatori di canale
   Gli identificatori di canale (CIDs) sono nomi locali che rappresentano un canale logico su un dispositivo. Le implementazioni sono libere di gestire i CID nel modo più adeguato per quella particolare implementazione, provvedendo però a non riusare lo stesso CID per un canale locale L2CAP, usato in connessioni multiple e simultanee, tra un dispositivo locale e alcuni dispositivi remoti. L'assegnazione dei CID è relativa a un particolare dispositivo, e un dispositivo può assegnare i CID indipendentemente dagli altri dispositivi (con l'eccezione di certi CID riservati). Quindi, sebbene lo stesso CID sia stato assegnato a channel end-points (remoti) da diversi dispositivi remoti connessi ad un unico dispositivo locale, tale dispositivo può tuttavia associare univocamente ogni CID remoto a un dispositivo differente.

2. Operazioni tra dispositivi
   I data-channel connection oriented rappresentano una connessione tra due dispositivi, dove un CID identifica ogni end-point del canale. I canali connectionless limitano il flusso dati a una singola direzione. Questi canali sono utilizzati per sostenere un channel group, dove il CID (del dispositivo sorgente) rappresenta uno o più dispositivi remoti. C'è, inoltre, un certo numero di CID riservati per scopi speciali. Il cosiddetto signalling channel è un esempio di canale riservato. Tale canale è utilizzato per creare

e stabilire canali dati connection-oriented e per negoziarecambiamenti alle caratteristiche di questi canali. Un altro CID è riservato per il traffico dati (in entrata) di tipo connectionless.

3. Operazioni tra Layer
   Le implementazioni di L2CAP seguono le seguenti regole generali:

   - Le implementazioni di L2CAP devono trasferire dati tra protocolli a livelli superiori e i protocolli a livelli più bassi.
   - Ogni implementazione deve anche avere un set di signalling commands da usare tra implementazioni differenti di L2CAP.
   - Le implementazioni L2CAP dovrebbero, inoltre, essere pronte ad accettare certi tipi di eventi da layer più bassi e generare eventi verso layer a livello più alto. Come questi eventi siano "passati" tra i layer, è un processo dipendente dall'implementazione specifica

L2CAP State Machine

Questa sezione descrive gli stati dei canali L2CAP connection-oriented. La sezione definisce gli stati, gli event che causano le transizioni di stato e le azioni intraprese in risposta agli eventi. Nell'immagine vengono illustrati gli event e le azioni eseguite da un'implementazione del layer L2CAP. Client e Server rappresentano semplicemente "chi fa la richiesta" e "chi accetta la richiesta" stessa. La convenzione è la seguente:

- L'interfaccia tra due layer (interfaccia verticale) usa il prefisso del layer più basso che offre il servizio al layer posto più in alto, es. L2CA.
- L'interfaccia tra due entità dello stesso layer (interfaccia orizzontale) usa il prefisso del protocollo (aggiungendo una P all'identificativo del layer), es. L2CAP.
- Gli eventi che "provengono da sopra" (che iniziano sopra) sono chiamati Request (Req) e la corrispondente risposta è chiamata Confirm (Cfm).
- Gli eventi che "provengono dal basso" sono chiamati Indication (Ind) e la corrispondente risposta è chiamata Response (Rsp).
- Response, che richiedono ulteriori processi, sono chiamati Pending (Pnd). La notazione per Response e Confirm assume risposta positiva. Risposte negative sono denotate dal suffisso Neg, come ad esempio L2CAP_Connect_Cfm_Neg.

Altre caratteristiche

- Data Packet Format
  L2CAP è basato su pacchetti, ma segue un modello di comunicazione basato su canali. Un canale rappresenta un flusso dati tra due entità L2CAP in dispositivi remoti. I canali possono essere connection-oriented o connectionless. Tutti i campi dei pacchetti utilizzano l'ordinamento dei byte secondo la regola Little Endian.

- Signalling
  Vari comandi di signalling possono essere passati tra due entità L2CAP su dispositivi remoti. Tutti questi comandi sono spediti al CID 0x0001 (signalling channel). L'implementazione di L2CAP deve essere in grado di determinare l'indirizzo BD_ADDR del dispositivo che ha inviato i comandi.

  Molti comandi possono essere inviati in un singolo pacchetto (L2CAP) e i pacchetti sono spediti al CID 0x0001.

- Parametri opzionali di configurazione
  Le opzioni (option) sono dei meccanismi che estendono le possibilità di negoziare differenti requisiti della connessione.

- Primitive di servizio
  L2CAP offre diversi servizi, in termini di primitive di servizio e parametri. L'interfaccia di servizio è richiesta nelle operazioni di testing. Sono incluse primitive per:

  - Connessione: setup, configurazione, disconnessione.
  - Data: read, write.
  - Gruppi (Group): creazione, chiusura, aggiunta/rimozione di un membro.
  - Traffico connectionless: abilitato, disabilitato.

RFCOMM Protocol

Il protocollo RFCOMM emula porte seriali per il protocollo L2CAP. Il protocollo è basato sullo standard ETSI TS 07.10. Solo un subset dello standard TS 07.10 viene utilizzato e sono specificati alcuni adattamenti al protocollo nelle specifiche Bluetooth di RFCOMM.

RFCOMM Overview/Service

72 pin	Circuit Name
102	Signal Common
103	Transmit Data (TD)
104	Received Data (RD)
105	Request to Send (RTS)
106	Clear to Send (TCS)
107	Data Set Read (DST)
108	Data Terminal Ready (DTR)
109	Data Carrier Detect (CD)
125	Ring Indicator (RI)

Tabella 3.3: Circuiti interfaccia RS-232

RFCOMM è un semplice protocollo di trasporto, che emula la porta seriale RS-232 per il protocollo L2CAP. Il protocollo è basato sullo standard TS 07.10, del quale viene utilizzato solo un subset, ed è stata aggiunta un'estensione specifica, nella forma di uno schema di flusso di controllo obbligatorio. RFCOMM riesce a mantenere fino a 60 connessioni simultanee tra due dispositivi. Il numero di connessioni che possono essere usate contemporaneamente in un dispositivo Bluetooth dipende dall'implementazione specifica.

- Device Types
  Esistono due tipi di dispositivi a cui RFCOMM deve provvedere:
  Tipo 1: End-point di comunicazione, come computer e stampanti.
  Tipo 2: Dispositivi che fanno parte della sezione di comunicazione, ad esempio, i modem.

  Sebbene RFCOMM non faccia distinzione tra questi due tipi di dispositivo, provvedere a entrambi incide nel protocollo RFCOMM stesso. L'informazione trasferita fra due entità RFCOMM è stata definita in modo da essere accettata da entrambi i tipi di dispositivo. Alcune informazioni sono richieste solo dai dispositivi di tipo 2, mentre altre informazioni sono destinate ad essere utilizzate da entrambi. Nel protocollo, non è fatta distinzione tra tipo 1 e tipo 2. Poichè il dispositivo non è a conoscenza del tipo dell'altro dispositivo, nel percorso di comunicazione, ognuno deve passare tutte le informazioni disponibili, specificate dal protocollo.

- Control Signals

- Null Modem Emulation
  RFCOMM è basato sullo standard TS 07.10. Il modo in cui TS 07.10 trasferisce i segnali di controllo RS-232 crea un null modem implicito, nel momento in cui due dispositivi dello stesso tipo sono connessi l'uno all'altro

- Multiple Emulated Serial Ports
  Due dispositivi Bluetooth, che usano RFCOMM nella loro comunicazione, possono aprire "porte seriali emulate multiple". RFCOMM supporta fino a 60 porte (emulate) aperte; comunque, il numero di porte che possono essere usate in un dispositivo è specifico dell'implementazione. Un Data Link Connection Identifier (DLCI) identifica una connessione in corso fra un'applicazione client e un server. Il DLCI è rappresentato da 6 bit, ma il suo range utilizzabile è 2 - 61. Il DLCI è unico per ogni sessione RFCOMM tra due dispositivi.

Adattamenti TS 07.10 per RFCOMM

- Media Adaption
  I flag di apertura e chiusura nel frame 07.10 basic option, non sono utilizzati in RFCOMM. C'è sempre esattamente un frame RFCOMM in ogni frame L2CAP.

- TS 07.10 Multiplexer Startup & Closedown Procedure
  Le procedure di start-up e closedown, non sono supportate. Ogni volta ci deve essere al massimo una sessione RFCOMM tra ogni coppia di dispositivi. Quando si instaura un nuovo DLC, l'entità (che inizia la comunicazione) deve controllare se esiste già una sessione RFCOMM con il dispositivo remoto, e, se è così, sostituisce il vecchio DLC con il nuovo. Una sessione è identificata dal Bluetooth BD_ADDR dei due end-point.

  – Step 1: Procedura di start-up
    Il dispositivo che apre la prima connessione, con la porta seriale emulata, ha la responsabilità, per prima cosa, di instaurare il multiplexer control channel. Ciò implica i seguenti step, dopo dei quali i DLCs per traffico dati, possono essere stabiliti:

    * Instaurare un canale L2CAP verso l'entità RFCOMM di pari livello, tramite l'uso delle primitive di servizio L2CAP.
    * Avviare il multiplexer RFCOMM inviando il comando SABM sul DLCI 0 e attendere la risposta, detta UA response, dall'entità di pari livello.

  – Step 2: Procedura di closedown
    Il dispositivo che chiude l'ultima connessione (DLC), di una particolare sessione, è responsabile della chiusura del multiplexer chiudendo il corrispondente L2CAP channel. Chiudere il multiplexer inviando prima un DISC command frame al DLCI 0 è opzionale, ma è obbligatorio rispondere correttamente al DISC (con un'UA response).

  – Step 3: Link Loss Handling (gestione del Link Loss). Se viene ricevuta una notifica di L2CAP Link Loss, l'entità locale RFCOMM. ha la responsabilità di inviare una notifica "Connection Loss " verso l'entità port emulation per ogni DLC attivo. Quindi, tutte le risorse associate alla sessione RFCOMM possono essere liberate.

- DLCI Allocation with RFCOMM Server Channels
  Considerando che sia l'applicazione client, sia l'applicazione server possono

trovarsi in entrambi i lati di una sessione RFCOMM, con client in entrambi i lati che creano connessioni indipendentemente da tutti gli altri, lo spazio dei valori di DLCI viene diviso tra i due dispositivi comunicanti, usando il concetto di "RFCOMM server channel" e un bit detto direction bit (bit di direzione). Il numero di canali RFCOMM server è un sottoinsieme dei bit nella parte DLCI del campo address del frame TS 07.10. Alle Server Application, che si registrano (ad un servizio) con un'interfaccia RFCOMM service interface, vengono assegnati i numeri di Server Channel. nel range 1 - 30. Per una sessione RFCOMM, al dispositivo che comincia la comunicazione viene dato il valore 1 al direction bit (D=1) e, viceversa, D=0 all'altro dispositivo. Quando viene instaurata una nuova connessione data link su una sessione RFCOMM esistente, il direction bit viene usato insieme al Server Channel per determinare il DLCI da usare per connettersi ad una specifica applicazione. Questo DLCI è successivamente utilizzato per tutti i pacchetti in transito in entrambe le direzioni. Un'entità RFCOMM, che crea un nuovo DLC in una sessione esistente, forma il DLCI, combinando il Server Channel, per l'applicazione, sull'altro dispositivo e l'opposto del suo direction bit per la sessione.

- Multiplexer Control Commands
  Si noti che in TS 07.10, alcuni comandi di Multiplexer Control, riguardanti DLCI specifici, possono essere scambiati sul canale di controllo (DLCI 0) prima che il DLC corrispondente sia stato instaurato.

  - Remote Port Negotiation (RPN) Command
    Il RPN command può essere usato prima che sia stato aperto un nuovo DLC e dovrebbe essere utilizzato ogni qualvolta il settaggio della porta cambia.

  - Remote Line Status (RLS) Command
    Questo comando è usato per indicare lo stato della porta remota.

  - DLC Parameter Negotiation (PN) Command
    É obbligatorio l'utilizzo, di questo comando, per le implementazioni di RFCOMM conformi alle specifiche Bluetooth ver1.1 e successive. Questo comando deve essere usato prima della creazione del primo DLC su una sessione RFCOMM, e chi inizia la comunicazione deve cercare di attivare il controllo di flusso "credit based".

Metodi di Flow Control usati

Il controllo del flusso tra RFCOM e il layer sottostante, L2CAP, dipende dall'interfaccia di servizio supportata dall'implementazione. In aggiunta, RFCOMM ha il suo meccanismo di "controllo del flusso". Di seguito sono descritti i vari meccanismi di flow control:

1. L2CAP Flow Control
   Dipende dal meccanismo di flow control del layer Link Manager nel Baseband; il meccanismo di flow control tra i layer L2CAP e RFCOMM è specifico dell'implementazione.
2. Wired Serial Port Flow Control
   Si divide in due campi:
   (a) Controllo di flusso software che usa caratteri quali XON/XOFF.
   (b) Controllo di flusso hardware che usa circuiti RTS/CTS o DTR/DSR.

   Questi metodi possono essere utilizzati da entrambi i lati di un wired link, oppure possono essere utilizzati in una sola direzione.
3. RFCOMM Flow Control
   Il protocollo RFCOMM prevede due meccanismi di flow control:
   - RFCOMM contiene comandi di flow control che operano sul flusso dati globale tra due entità RFCOMM.
   - Il comando Modem Status è il meccanismo di flow control che opera su singoli DLCI.
4. Port Emulation Entity: Serial Flow Control
   Su dispositivi di tipo 1 alcuni port driver (Port Emulation Entities più RFCOMM) hanno bisogno che siano forniti loro servizi di flow control come specificato dalle API che stanno emulando. Un'applicazione può richiedere un particolare meccanismo di flow control come XON/XOFF o RTS/CTS, e aspettarsi che sia il port driver a gestire il controllo di flusso.

   Su dispositivi di tipo 2, il port driver potrebbe aver bisogno di eseguire il controllo del flusso sulla parte non-RFCOMM del percorso di comunicazione.

   Questo controllo del flusso è specificato tramite parametri di controllo inviati dalla pari (peer) entità RFCOMM (solitamente un dispositivo di tipo 1).

La descrizione del controllo del flusso, in questa sezione, è per port driver di dispositivi di tipo 1.

Siccome RFCOMM ha già il proprio meccanismo di flow control, il port drive non ha bisogno di eseguire il controllo del flusso utilizzando i metodi richiesti dall'applicazione. Nel caso ideale, l'applicazione stabilisce un meccanismo di flow control e assume che il sistema COMM gestisca i dettagli.

Il port driver può, quindi, ignorare, semplicemente, la richiesta e dipenderà dal meccanismo di flow control di RFCOMM. L'applicazione è ora in grado di inviare e ricevere dati, non sapendo che il port driver non esegue il controllo del flusso. Comunque, nel mondo reale si presentano alcuni problemi

- Il port driver basato su RFCOMM sta funzionando in conformità a un protocollo packet-based, dove i dati possono essere bufferizzati da qualche parte nel percorso di comunicazione. Quindi il port driver non può eseguire il controllo di flusso con la stessa precisione del caso wired.
- L'applicazione può decidere in autonomia di applicare il meccanismo di flow control in aggiunta al controllo di flusso richiesto dal port driver

Questi problemi suggeriscono che il port driver deve compiere del lavoro addizionale per eseguire correttamente l'emulazione del controllo di flusso. Di seguito sono riportate le regole base per l'emulazione del flow control:

- Il port driver non dipende solamente dal meccanismo richiesto dall'applicazione, ma utilizza una combinazione di meccanismi di controllo.
- Il port driver deve essere a conoscenza del meccanismo di controllo del flusso richiesto dall'applicazione e comportarsi come nel caso wired, utilizzando il meccanismo richiesto.
- Il port driver deve essere a conoscenza del meccanismo di controllo del flusso richiesto dall'applicazione e comportarsi come nel caso wired, quando vede cambiamenti sui circuiti non-data (hardware flow control) caratteristiche di flow control nei dati in entrata (software flow control).

- Based Flow Control (CBFC) E' una caratteristica obbligatoria che non esiste nelle specifiche Bluetooth RFCOMM 1.0B e precedenti. Quindi, il suo uso è soggetto ad una negoziazione prima della prima instaurazione di DLC. Le implementazioni conformi a queste specifiche devono supportare il CBFC e devono tentare di utilizzarlo in connessioni con altri dispositivi. Quando il Credit Based Flow Control viene utilizzato, entrambi i dispositivi coinvolti in una sessione RFCOMM conosceranno, per ogni DLC, quanti frame RFCOMM l'altro dispositivo è in grado di accettare prima che il buffer, per quel DLC, si riempia. Un'entità, detta sending entity (cioè l'entità che spedisce messaggi), può inviare tanti frame su un DLC quanti ne ha a credito; se il "credit count" raggiunge lo zero, il sender deve fermarsi e aspettare ulteriori crediti dall'entità pari. É però, sempre permesso inviare frame che non contengono user data (campo length = 0) quando è in uso il CBFC. Questo meccanismo opera indipendentemente per ogni DLC e per ogni direzione.

OBEX (OBject EXchange) Protocol

OBEX, abbreviazione di object exchange, è un protocollo di comunicazione che facilita lo scambio di oggetti (binari) tra dispositivi. É mantenuto efficiente dall'Infrared Data Association, ma è stato adottato anche dal Bluetooth Special Interest Group. Sebbene OBEX fu inizialmente progettato per gli infrarossi, è tuttavia utilizzato in associazione a RS-232, USB e WAP. OBEX è un protocollo di sessione e può essere descritto, in modo migliore, come un protocollo HTTP binario. É ottimizzato per link wireless ad-hoc e può essere utilizzato per scambiare qualsiasi tipo di dato, da file generici, a immagini, biglietti da visita (vCard) e oggetti calendario (vCalendar).

Differenze tra OBEX e http

OBEX e HTTP sono simili in quanto a funzioni, poiché un client utilizza un trasporto affidabile per connettersi ad un server e può, poi, richiedere e/o mettere a disposizione oggetti. Ma OBEX differisce in alcuni aspetti importanti:

- Trasporti HTTP utilizza testo leggibile (human-readable text), mentre OBEX scambia informazioni, circa una richiesta o un oggetto, tramite l'utilizzo di triplette type-length-value, formattate in binario.

- Supporto di sessione Le transazioni HTTP sono dette stateless (senza stato); un client http apre una connessione, esegue una singola richiesta, riceve la risposta corrispondente e chiude la connessione. In OBEX, una singola

connessione di trasporto può produrre molte operazioni collegate. Infatti, recenti aggiornamenti, alle specifiche OBEX, permettono una chiusura inaspettata della transazione, la quale può essere ripresa con tutte le informazioni di stato intatte.

Profili Bluetooth

Nell'ottica della interoperabilità, in aggiunta ai requisiti del sistema radio ed ai protocolli, la specifica Bluetooth descrive quelli che vengono chiamati profili. Un profilo è un insieme di definizioni, raccomandazioni e requisiti comuni per la corretta implementazione di un particolare scenario applicativo. Per requisiti si intendono i servizi che devono essere supportati e le procedure atte ad implementare le varie funzioni, ognuna riferita ad un particolare layer dello stack. Ogni profilo, pertanto, può esser visto come una sezione verticale dello stack di protocolli Bluetooth.

Tra i vari profili esistono delle relazioni di dipendenza, nel senso che alcuni di essi usano gerarchicamente altri. Le relazioni fra diversi profili definiti nella versione della specifica Bluetooth 2.0 sono rappresentate in Figura 3.6: un profilo dipendente è rappresentato come contenuto in quello da cui dipende. Si vede perciò piuttosto chiaramente che tutti i profili dipendono direttamente o indirettamente dal Generic Access Profile. Esso definisce le procedure atte ad assicurare l'interoperabilità più elementare, essenzialmente garantendo la capacità di un dispositivo di individuare altri apparecchi nelle vicinanze ed eventualmente connettervisi, o di essere a sua volta individuato e connesso. Tale profilo definisce inoltre i requisiti per le procedure di autenticazione e cifratura delle connessioni. La maggior parte, se non la totalità, degli apparati Bluetooth oggi esistenti si appoggia ai profili già definiti. Ricordiamo, ad esempio, gli auricolari per telefoni cellulari (Headset Profile), l'accesso wireless ad una rete di calcolatori (Personal Area Networking Profile), mouse e tastiere senza cavi (Serial Port Profile, Generic Object Exchange Profile) e l'uso di telefoni cellulari come modem senza fili (Dial-up Networking Profile). Tuttavia, sebbene il Bluetooth SIG (Special Interest Group, il consorzio di aziende direttamente interessate allo sviluppo della tecnologia Bluetooth) si adoperi per aggiungere alla specifica sempre nuovi profili, accade spesso di immaginare e voler realizzare un'applicazione per cui non sia stato previsto alcun profilo specifico. In questo caso, l'unico profilo che occorre rispettare è il Generic Access Profile.

Ecco una lista dei principali profili Bluetooth :

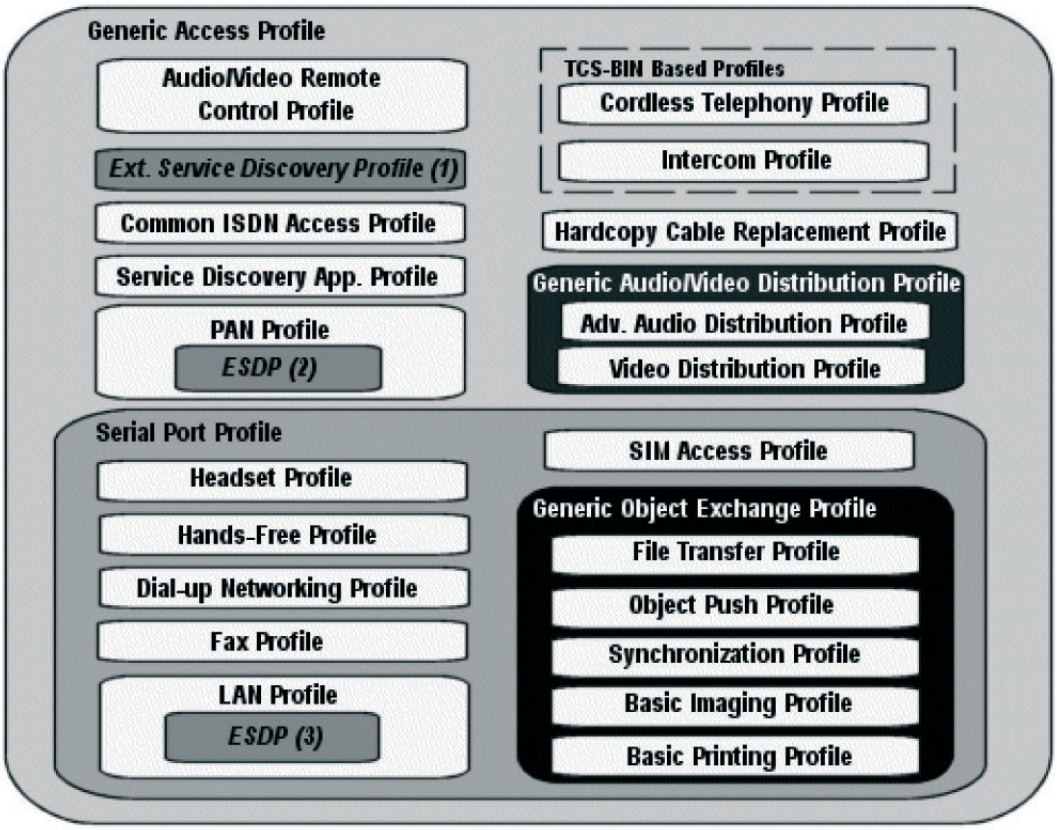

Figura 3.6: Profili Bluetooth

- Advanced Audio Distribution Profile (A2DP): profilo di distribuzione audio avanzata
- Audio Video Remote Control Profile (AVRCP): profilo di telecomando multimediale
- Basic Imaging Profile (BIP): profilo d'infografia di base
- Basic Printing Profile (BPP): profilo di stampa di base
- Cordless Telephony Profile (CTP): profilo di telefonia senza fili
- Dial-up Networking Profile (DUNP): profilo d'accesso alla rete a distanza
- Fax Profile (FAX): profilo di fax
- File Transfer Profile (FTP): profilo di trasferimento di file
- Generic Access Profile (GAP): profilo d'accesso generico
- Generic Object Exchange Profile (GOEP): profilo di scambio di oggetti
- Hardcopy Cable Replacement Profile (HCRP): profilo di sostituzione di hardcopy
- Hands-Free Profile (HFP): profilo mani libere
- Human Interface Device Profile (HID): profilo d'interfaccia uomo-terminale
- Headset Profile (HSP): profilo dell'auricolare
- Intercom Profile (IP): profilo d'intercom (walkie-talkie)
- LAN Access Profile (LAP): profilo d'accesso alla rete
- Object Push Profile (OPP): profilo di invio dei file
- Personal Area Networking Profile (PAN): profilo di rete personale
- SIM Access Profile (SAP): profilo di accesso ad una scheda SIM
- Service Discovery Application Profile (SDAP): profilo di scoperta delle applicazioni
- Synchronization Profile (SP): profilo di sincronizzazione con un gestionario delle informazioni personali (detto PIM per Personal Information Manager).
- Serial Port Profile (SPP): profilo di porta seriale

3.4 Interferenza con il Wi-fi

Lo standard industriale Bluetooth, come detto trasmette sulle frequenze ISM (frequenze libere dedicate ad applicazioni industriali, di ricerca e mediche) a 2.4 Gherz: la stessa banda utilizzata dalla tecnologia Wi-fi

3.4.1 Specifiche del wi-fi

Wi-Fi, abbreviazione di Wireless Fidelity, è un termine che indica dispositivi che possono collegarsi a reti locali senza fili (WLAN) basate sulle specifiche IEEE 802.11. Un dispositivo, anche se conforme a queste specifiche, non può utilizzare il logo ufficiale Wi-Fi se non ha superato le procedure di certificazione stabilite dal consorzio Wi-Fi Alliance (Wireless Ethernet Compatibility Alliance), che testa e certifica la compatibilità dei componenti wireless con gli standard 802.11x (della famiglia 802.11).

Il Wifi trova la sua maggior applicazione nella fornitura di Internet veloce; accessi wi-fi sono disponibili in aeroporti, stazioni ferroviarie, interne cafè sparsi per il mondo. Esistono anche città, gruppi o singoli individui che hanno costruito reti wi-fi adottando un regolamento comune per garantirne l'interoperabilità.

Negli ultimi anni, alcune province e amministrazioni comunali hanno avviato progetti per la realizzazione di reti civiche con tecnologia Wi-Fi. Tipicamente le reti realizzate sono di proprietà pubblica, mentre la loro gestione è affidata ad un concessionario privato. Le reti collegano le pubbliche amministrazioni del territorio locale e forniscono un accesso diffuso alla banda larga in quelle zone in cui gli operatori nazionali non intendono investire per via degli alti costi (es. territori montuosi).

3.4.2 Utilizzo del Wi-fi

Per i bassissimi costi della tecnologia, il wi-fi è la soluzione principale per il digital divide, che esclude numerosi cittadini dall'accesso alla banda larga. Wi-fi è usato da anni in tutto il mondo per portare connettività veloce nelle zone isolate e nei piccoli centri. Negli USA (laddove l'UMTS si è rivelato un fallimento[2]), si è sperimentata anche un'integrazione con la telefonia mobile dove il wi-fi dovrebbe sostituire le vecchie antenne GSM/GPRS/UMTS, con una nuova rete in grado di dare le velocità sperate e i servizi di videotelefonia. Ci sono prospettive di integrare fonia fissa e mobile in un unico apparecchio che con lo stesso numero funzioni da fisso/cordless nel raggio di 300 metri da casa e oltre come un normale

cellulare. Grazie al wi-fi, anche i centri più piccoli hanno la possibilità di accesso veloce ad Internet, pur non essendo coperti da ADSL. In molti sostengono che i dispositivi Wi-Fi sostituiranno i telefoni cellulari e le reti GSM. Nel futuro più prossimo, costituiscono ostacoli a questo fatto: l'impossibilità del roaming e delle opzioni di autenticazione (802.1x, SIM e RADIUS), la limitatezza dello spettro di frequenze disponibili e del raggio di azione del Wi-Fi. Molti operatori, (Vodafone e Orange in primis) iniziano a vendere dispositivi mobili per accedere a internet, che collegano schede wireless dei cellulari e ricevitori wi-fi per trarre benefici da entrambi i sistemi. Ci si attende che in futuro i sistemi wireless operino normalmente fra una pluralità di sistemi radio.

3.4.3 Lo standardIEEE 802.11

EEE 802.11 definisce uno standard per le reti WLAN sviluppato dal gruppo 11 dell'IEEE 802, in particolare in livello fisico e MAC del modello ISO/OSI, specificando sia l'interfaccia tra client e base station (o access point) sia tra client wireless.

Questo termine viene usualmente utilizzato per definire la prima serie di apparecchiature 802.11 sebbene si debba preferire il termine 802.11 legacy.

La famiglia 802.11 consta di tre protocolli dedicati alla trasmissione delle informazioni (a, b, g), la sicurezza è stata inclusa in uno standard a parte, 802.11i. Gli altri standard della famiglia (c, d, e, f, h) riguardano estensioni dei servizi base e miglioramenti di servizi già disponibili. Il primo protocollo largamente diffuso è stato il b; in seguito si sono diffusi il protocollo a e soprattutto il protocollo g.

802.11 legacy

La prima versione dello standard 802.11 venne presentata nel 1997 e viene chiamata 802.1y, specificava velocità di trasferimento comprese tra 1 e 2 Mb/s e utilizzava i raggi infrarossi o le onde radio nella frequenza di 2,4 GHz per la trasmissione del segnale. La trasmissione infrarosso venne eliminata dalle versioni successive dato lo scarso successo. La maggior parte dei costruttori infatti non aveva optato per lo standard IrDA, preferendo la trasmissione radio. Il supporto di questo standard per quanto riguarda la trasmissione via infrarossi è incluso delle evoluzioni dello standard 802.11 per ragioni di compatibilità. Poco dopo questo standard vennero realizzati da due produttori indipendenti delle evoluzioni dello standard 802.1y che una volta riunite e migliorate portarono alla definizione dello standard 802.11b.

802.11b

802.11b ha la capacità di trasmettere al massimo 11Mbit/s e utilizza il Carrier Sense Multiple Access con Collision Avoidance (CSMA/CA) come metodo di trasmissione delle informazioni. Una buona parte della banda disponibile viene utilizzata dal CSMA/CA. In pratica il massimo trasferimento ottenibile è di 5,9 Mbit/s in TCP e di 7,1 Mbit/s in UDP. Metallo, acqua e in generale ostacoli solidi riducono drasticamente la portata del segnale. Il protocollo utilizza le frequenze nell'intorno dei 2,4 GHz.

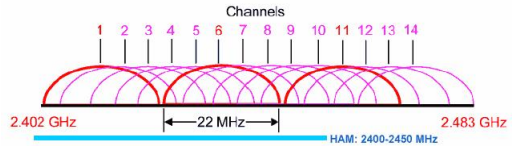

Figura 3.7: Canali Wi-fi

Utilizzando antenne direzionali esterne dotate di alto guadagno si è in grado di stabilire delle connessioni punto a punto del raggio di molti chilometri. Utilizzando ricevitori con guadagno di 80 decibel si può arrivare a 8 chilometri o se le condizioni del tempo sono favorevoli anche a distanze maggiori ma sono situazioni temporanee che non consentono una copertura affidabile. Quando il segnale è troppo disturbato o debole lo standard prevede di ridurre la velocità massima a 5,5, 2 o 1 Mb/s per consentire al segnale di essere decodificato correttamente.

Sono state sviluppate delle estensioni proprietarie che utilizzando più canali accoppiati consentono di incrementare la velocità di trasmissione a scapito della compatibilità con le periferiche prodotte dagli altri produttori. Queste estensioni normalmente vengono chiamate 802.11b+ e portano la banda teorica a 22, 33 o addirittura a 44 Mb/s.

Il primo produttore commerciale a utilizzare il protocollo 802.11b è stata Apple Computer con il marchio AirPort. Il primo produttore per IBM compatibili è stato Linksys.

Canali e compatibilità internazionale

802.11b e 802.11g dividono lo spettro in 14 sottocanali da 22 MHz l'uno come mostrato in figura 3.7. I canali sono parzialmente sovrapposti tra loro in frequenza, quindi tra due canali consecutivi esiste una forte interferenza. I 2 gruppi di canali 1, 6, 11 e 2, 7 e 12 non si sovrappongono fra loro e vengono utilizzati negli ambienti con altre reti wireless. Gli unici canali utilizzabili in tutto il mondo sono

il 10 e 11 dato che la Spagna non ha concesso i canali dall'1 al 9 e molte nazioni si limitano ai primi 11 sottocanali.

3.4.4 802.11a

Nel 2001 venne ratificato il protocollo 802.11a approvato nel 1999. Questo standard utilizza lo spazio di frequenze nell'intorno dei 5 GHz e opera con una velocità massima di 54 Mb/s sebbene nella realtà la velocità reale disponibile all'utente sia di circa 20 Mb/s. La velocità massima può essere ridotta a 48, 36, 24, 18, 9 o 6 se le interferenze elettromagnetiche lo impongono. Lo standard definisce 12 canali non sovrapposti, 8 dedicati alle comunicazioni interne e 4 per le comunicazioni punto a punto. Quasi ogni stato ha emanato una direttiva diversa per regolare le frequenze ma dopo la conferenza mondiale per la radiocomunicazione del 2003 l'autorità federale americana ha deciso di rendere libere secondo i criteri già visti le frequenze utilizzate dallo standard 802.11a.

Questo standard non ha riscosso i favori del pubblico dato che l'802.11b si era già molto diffuso e in molti paesi l'uso delle frequenze a 5 GHz è tuttora riservato. In Europa lo standard 802.11a non è stato autorizzato all'utilizzo dato che quelle frequenze erano riservate all'HYPERLAN; solo a metà del 2002 tali frequenze vennero liberalizzate e quindi si potè utilizzare l'802.11a. Esistono schede dual standard o tri standard in grado di accettare oltre allo standard a anche il b e per le schede tri standard anche il g. Ovviamente esistono anche degli Access point multi standard.

3.4.5 802.11f

Anche chiamato Inter Access Point Protocol (IAPP), è un protocollo di livello applicazione per la gestione di ESS (Extented Service Set), ovvero più reti wireless collegate tra di loro, gestendo l'handover di terminali da una rete wireless all'altra.

3.4.6 802.11g

Questo standard venne ratificato nel giugno del 2003. Utilizza le stesse frequenze dello standard 802.11b cioè la banda di 2,4 GHz e fornisce una banda teorica di 54 Mb/s che nella realtà si traduce in una banda netta di 24,7 Mb/s, simile a quella dello standard 802.11a. è totalmente compatibile con lo standard b ma quando si trova a operare con periferiche b deve ovviamente ridurre la sua velocità a quella dello standard b.

Prima della ratifica ufficiale dello standard 802.11g avvenuta nell'estate del 2003 vi erano dei produttori indipendenti che fornivano delle apparecchiature basate su specifiche non definitive dello standard. I principali produttori comunque preferirono aderire alle specifiche ufficiali e quando queste vennero pubblicate molti dei loro prodotti furono adeguati al nuovo standard.

Alcuni produttori introdussero delle ulteriori varianti chiamate g+ o Super G nei loro prodotti. Queste varianti utilizzavano l'accoppiata di due canali per raddoppiare la banda disponibile anche se questo induceva interferenze con le altre reti e non era supportato da tutte le schede.

Il primo grande produttore a rilasciare schede con le specifiche ufficiali 802.11g fu nuovamente Apple che presentò i suoi prodotti AirPort Extreme. Cisco decise di entrare nel settore acquistando Linksys, e fornì i suoi prodotti con il nome di Aironet.

3.4.7 802.11n

Nel gennaio 2004 IEEE ha annunciato di aver avviato lo studio di un nuovo standard per realizzare reti wireless di dimensioni metropolitane. La velocità reale di questo standard dovrebbe essere di 100 Mb/s (quella fisica dovrebbe essere prossima a 524 Mb/s), quindi dovrebbe essere 5 volte più rapido del 802.11g e 40 volte più rapido dell'802.11b.

Il 19 gennaio 2007 il gruppo di lavoro 802.11 di IEEE ha approvato la Draft 2.0; sulla quale si sono basate le aziende produttrici per rilasciare i loro prodotti della fascia Draft n. Il primo grande produttore a rilasciare prodotti con le specifiche ufficiali 802.11n draft 2.0 fu ancora una volta Apple che presentò i suoi Macbook nella seconda metà del 2006, tutti forniti di serie con dispositivi compliant alla specifica 802.11n, ancora prima della ratifica ufficiale.

Nel marzo 2009 il gruppo di lavoro TGn è arrivato al draft 8.0.

La versione definitiva dello standard è stata approvata l'11 settembre 2009 e la pubblicazione è prevista per l'inizio del 2010.

Tabella Riassuntiva

Utilizzo dei canali Wi-fi

3.4.8 CSMA/CA

Il meccanismo primario QoS in reti 802.11 è evitare le collisioni utilizzando CSMA/CA. Questo è il metodo di rilevamento delle *collisioni*, una collisione avviene

Standard	Frequenza	Velocità di trasferimento (Mbit/s)
802.11 legacy	FHSS, 2,4 GHz, IR	1, 2
802.11a	5,2, 5,4, 5,8 GHz	6, 9, 12, 18, 24, 36, 48, 54
802.11b	2,4 GHz	1, 2, 5.5, 11
802.11g	2,4 GHz	6, 9, 12, 18, 24, 36, 48, 54
802.11n	2,4 GHz, 5,4 Ghz	1, 2, 5, 12, 18, 24, 36, 48, 54, 125

Tabella 3.4: Standard 802.11, frequenze e velocità

Canale	Frequenza (GHz)	Europa	Francia	Spagna	Giappone	US
1	2.412	X			X	X
2	2.417	X			X	X
3	2.422	X			X	X
4	2.427	X			X	X
5	2.432	X			X	X
6	2.437	X			X	X
7	2.442	X			X	X
8	2.447	X			X	X
9	2.452	X			X	X
10	2.457	X	X	X	X	X
11	2.462	X	X	X	X	X
12	2.467	X	X		X	
13	2.472	X	X		X	
14	2.477				X	

Tabella 3.5: Standard 802.11, utilizzo dei canali

quando due stazioni tentano di comunicare contemporaneamente utilizzando lo stesso media, i due messaggi in questo modo diventano illeggibili. CSMA/CA è l'acronimo inglese di Carrier Sense Multiple Access with Collision Avoidance, ovvero accesso multiplo tramite rilevamento della portante che evita collisioni. É un'evoluzione del protocollo CSMA con accorgimenti ulteriori per ridurre le collisioni ed è alternativo al più efficiente CSMA/CD, che non può essere usato nelle reti senza fili a causa della difficoltà di realizzazione di un apparato che possa contemporaneamente trasmettere ed ascoltare sullo stesso canale radio.

Nel momento in cui una stazione vuole tentare una trasmissione ascolta il canale (Listen-before-Transmit). Se il canale è occupato la stazione attiva un timer di durata casuale (detto tempo di backoff) che viene decrementato solo

durante i periodi di inattività del canale. Quando il timer arriva a zero la stazione fa un altro tentativo. Se il canale risulta libero lo prenota ed attende per un certo lasso di tempo. Se il canale continua ad essere libero (non ci sono state altre prenotazioni) trasmette.

Carrier Sense descrive il fatto che un trasmettitore resta in ascolto per un'onda portante prima di tentare di inviare. Cioè, tenta di rilevare la presenza di un segnale codificato da un'altra stazione prima di tentare di trasmettere.

Multiple Access descrive il fatto che più stazioni possono inviare e ricevere sullo stesso supporto. Le trasmissioni con un nodo sono in genere ricevute da tutte le altre stazioni.

3.4.9 Le interferenze

Benchè il Wi-fi (da ora in avanti viene indicato con Wi-fi solamente l'insieme di protocolli a 2.4 Ghz) e Bluetooth non siano competitori, utilizzano la stessa banda ISM(industrial, scientific e medical), da 2.40 2.48 GHz. Il wi-fi utilizza 14 canali con parziale sovrapposizione, con un banda di 22 MHz e il Bluetooth invece salta attraverso 79 canali con banda di 1 Mhz equispaziati.

Come detto precedentemente il sistema Wi-fi utilizza il sistema di csma/ca per evitare le collisioni, la specifica Bluetooth invece non alcun sistema per evitare le collisioni, se avviene una collisione la trasmissione viene interrotta. Nonostante il grande numero e la piccola dimensione dei canali Bluetooth, le collesioni sono relativamente frequenti.

Uno studio del 2005 [17] sposorizzato da EDN (Information, News, and Business Strategy for Electronics Design [18]) ha messo in relazione la posizione delle le antenne e l'efficacia della comunicazione dimostrando che la distanza delle antenne è un fattore non trascurabile.

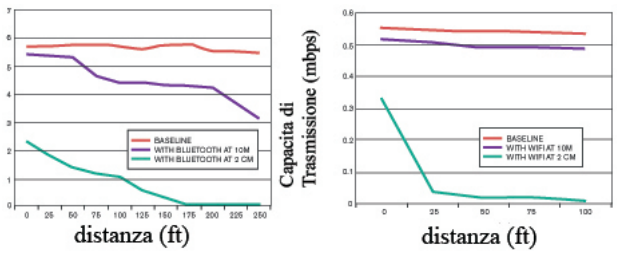

Figura 3.8: Iterazione Bluetooth e Wifi in base alla distanza

Nel caso le antenne siano poste a 2cm l'una dall'altra l'azione di disturbo è molto pesante e il Bluetooth può arrivare a coprire completamente la portante

wi-fi, mentre nel caso le distanze siano maggiori, 10 metri, i disturbi, pur non scomparendo si mitigano.

Per questo motivo uno sviluppo futuro del progetto netBlue è la realizzazione di modulo che permetta agli access point di limitare i canali di trasmissione del segnale in caso di presenza di segnale WI-fi. In teoria limitando la banda utilizzata dal Bluetooth, il Wi-fi, statisticamente, andrà ad occupare la porzione di banda lasciata libera dal Bluetooth, identificandoli come canali più sicuri.

Capitolo 4

GLI APPPARATI UTILIZZATI

4.1 Introduzione

In questo quarto capitolo vengono presentati i device utilizzati, un access point e un access server; sono dispositivi capaci di inviare contenuti multimediali attraverso la tecnologia Bluetooth.
Le funzionalità più elementari sono gestibili attraverso un semplice file di testo molto semplice da configurare, le funzionalità avanzate sono disponibili attraverso l'upload nella macchina di pacchetti in formato proprietario WPK

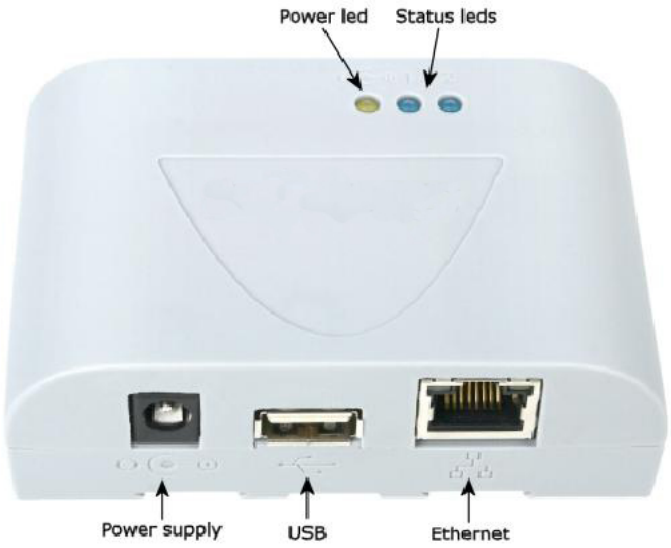

Figura 4.1: Access Point

4.2 Caratteristiche dei device

Si tratta di macchine (AS e AP) basate sul sistema operativo LINUX 2.60, analizzeremo poi i vantaggi di questa scelta. Sono molte le funzioni realizzate in modo automatico dall'apparato stesso, tra cui la ricerca automatica dei dispositivi e la connessione con essi, permettendo al programmatore di concentrarsi su altri aspetti critici. L'invio dei messaggi può essere gestito in modo semiautomatico, feature molto comoda, ma che si è dimostrata essere molto limitativa. La facilità nell'invio dei messaggi viene pagata con una pesante limitazione nelle possibilità di realizzare scenari di una certa complessità. Questo problema viene risolto attraverso la scrittura di eseguibili in linguaggi di alto livello C/C++ residenti nei device stessi. Attraverso un software development kit è possibile una gestione molto più a basso livello del comportamento dell'access point. La comunicazione con i device, a livello fisico, avviene principalmente attraverso un cavo di rete, creando una LAN, attraverso i protocolli HTML, SSH, STP. É possibile anche una comunicazione tramite un' USB, (e solamente per A, attraverso una connessione seriale e Digital I/O) che permette un set di funzioni limitato come vedremo in seguito. Le antenne usate dai dispositivi sono di classe 1 e hanno un raggio di 200 metri, ma mentre, AP dispone di 1 singola antenna, AS dispone di 3 antenne; il che li rende in grado di comunicare con, rispettivamente, fino a 7 e 21 cellulari (Si ricorda che una WLAN Bluetooth ha un limite superiore di 8 nodi).

4.3 Connessione al device

Il primo passo da intraprendere è quello di collegarsi al device, dal punto di vista fisico è sufficiente connettere con un cavo di rete incrociato il device e il computer. Dal punto di vista protocollare abbiamo diverse alternative:

- Interfaccia WWW
- Interfaccia SFTP
- Interfaccia SSH

4.3.1 Connessione WWW

É il modo più semplice e permette l'accesso a tutte le funzioni e opzioni per gestire una campagna pubblicitaria, ma non a tutte le opzioni per la gestione del device. Eccola procedura per questo tipo di connessione:

- Collegare il device e il PC con un cavo Ethernet. Lanciare l'appliacazione WRAPFinder. L'acquisizione dell'indirizzo può durare alcuni secondi, se non compare nulla premere Rescan. Se non dovesse comparire ancora nulla settare l'indirizzo di broadcast a 255.255.255.255.

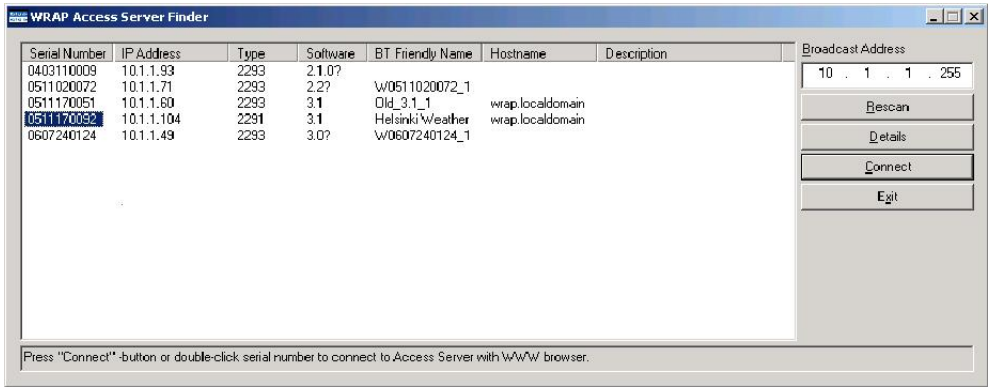

Figura 4.2: Wrap Access Server Finder

- Selezionare il dispositivo d'interesse e premere Connect. L'indirizzo della pagina è l'indirizzo IP assegnato al dispositivo. Si presenterà una interfaccia attraverso la quale sarà possibile configurare l'access point contattare il supporto clienti.

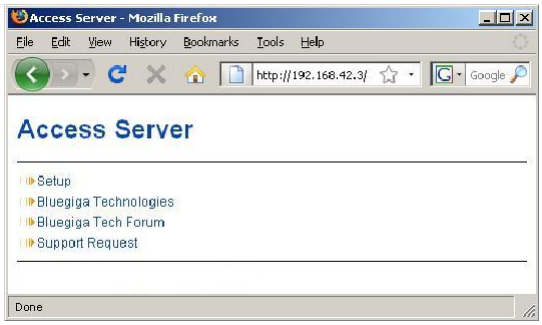

Figura 4.3: Interfaccia web

- Selezionare Setup e immettere la username e la password
- Una volta autenticati si accede alla pagina di setup della macchina

Se i caratteri dovessero risultare troppo piccoli si consiglia di premere simultaneamente ctrl e + (rispettivamente ctrl e -) per raggiungere una migliore visualizzazione.

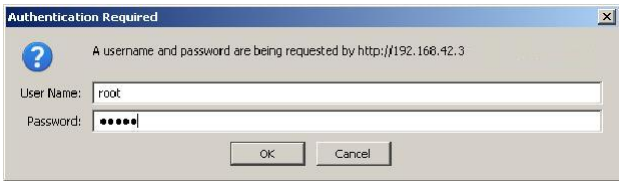

Figura 4.4: Console per l'inserimento della password nell'interfaccia Web

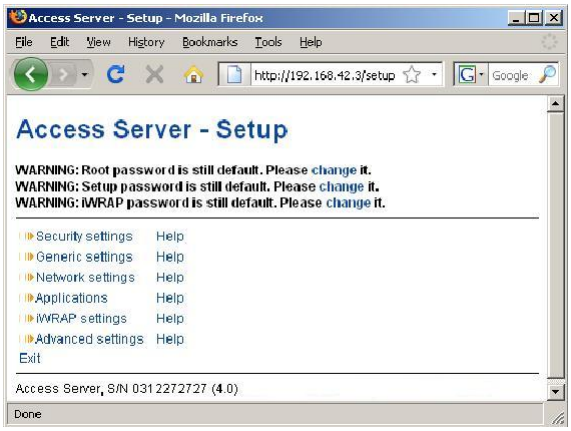

Figura 4.5: Pagina di Setup

4.3.2 Connessione SFTP

É possibile accedere all'access point anche attraverso una connessione via SFTP. Questa interfaccia permette di caricare e scaricare i file con maggiore facilità, permettendo ad utenti esperti di accedere direttamente ai file e agli script di configurazione. Segue la procedura per collegarsi con Winscp432:

- Collegare il dispositivo alla corrente elettrica e al PC attraverso un cavo di rete incrociato
- Attraverso WRAPFinder identificare l'indirizzo IP del dispositivo
- Aprire il browser SFTP.
- Nel campo Host name inserire l'indirizzo IP del device, compilare i campi User name e Password, e lasciare in bianco il campo Private key file.
- Premere Login, o invio

4.3.3 Connessione Seriale (solo AS)

Con questa connessione si accede alla shell testuale della macchina. Non va dimenticato che il firmware dei device è una versione LINUX (2.60) benchè non

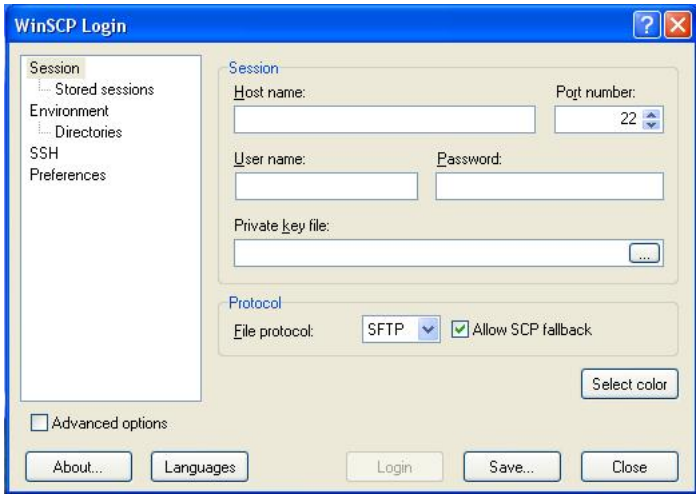

Figura 4.6: WinSCP

siano disponibili tutti i comandi, molti sono utilizzabili (si veda il manuale tecnico per la lista delle busybox applet utilizzabili). Benchè sia possibile configurare una campagna pubblicitaria attraverso l'interfaccia WWW , i pacchetti wpk e la connessione SFTP, è bene ricordare questa possibilità, per lo sviluppo di scenari avanzati. Con il comando setup si attiverà un menù navigabile.

Figura 4.7: Connessione Seriale

4.4 Operazioni preliminari per l'Access Point e l'Access Server

4.4.1 Introduzione

In questa sezione presento le operazioni necessarie per un corretto funzionamento della macchina. Ricordo ancora una volta che questi device sono macchine Linux, quindi la maggior parte dei file sono file di testo di tipo Linux nel formato single Line Fedd (ossia FL, \n). Se tali vengono modificati e risalvati in formato MS-DOS (se si usa per esempio Notepad in ambiente Windows), dove ogni linea finisce con Carriage Return e Line Feed (CR+LF, \r \n) i devices non riusciranno a interpretare correttamente i nuovi file. Per ovviare a questo problema è sufficiente convertire tali file in formato UNIX prima di eseguire l'upload. Se si modificano i file via l'interfaccia WWW o in ambiente LINUX questa operazione non è necessaria.

4.4.2 Pacchetti WPK

Un altra caratteristica saliente degli Access Server e Access Point è la possibilità di aggiungere interessanti features caricando pacchetti in formato proprietario wpk. Questi file hanno il seguente formato:

name-[version].[architecture].wpk

É importante prestare attenzione al campo architecture, infatti mentre alcuni pacchetti sono caricabili sia sulla versione server che sulla versione access point, altri sono dedicati, e l'errato caricamento provoca malfunzionamenti [19].

Architecture	Descrizione
noarch	Sia Access Server che Access Point
lt	Solo Access Point
if	Solo Access Server

Tabella 4.1: Sigle delle architetture dei device

Vediamo ora come installare uno di questi pacchetti, la procedura vale per tutti i pacchetti in questo formato.

4.5 Aggiornamento del Firmware

L'aggiornamento del Firmware è un operazione eseguibile con un pacchetto wpk. Come detto, è necessario prestare attenzione alla versione dell'aggiornamento: per la versione AP gli aggiornamenti sono nella forma:

reflash-x.x-1.lt.wpk

con x.x versione dell'aggiornamento. Per la versione AS è necessario usare il file:

reflash-x.x-1.if.wpk

É importante ricordare che questa operazione CANCELLA OGNI SETTAGGIO, COMPRESI I PACCHETTI CARICATI!!

Ci sono 4 possibilità di aggiornamento:

4.5.1 Aggiornamento via interfaccia WWW

1. Collegarsi all'apparecchio attraverso la procedura illustrata e andare nella pagina:

   Access Point → *Setup* → *Advanced Settings* → *Upload a software update.*

2. Attraverso il pulsante Browse selezionare il file di configurazione (reflash-x.x-1.lt.wpk) e premere Upload.

Figura 4.8: Schermata per l'aggiornamento software.

3. Ora è necessario aspettare 3-4 minuti, se l'operazione si sta svolgendo correttamente i led dovrebbero accendersi da una parte all'altra (12211221, 12344321123 per AS). É importante NON SPEGNERE IL DEVICE.

4. Una volta finita l'operazione i led 1 e 2 resteranno accesi per altri due minuti e il sistema ripartirà.

5. Per valutare il successo dell'operazione collegarsi con l'interfaccia WWW e confrontare a più di pagina la versione del Firmware.

4.5.2 Aggiornamento via USB

1. Accendere il dispositivo e caricare su una chiavetta UBS VUOTA il file reflash-x.x-1.lt.wpk
2. Connettere la chiavetta alla porta UBS del AP.
3. Ora è necessario aspettare 3-4 minuti, se l'operazione si sta svolgendo correttamente i led dovrebbero accendersi da una parte all'altra. É importante NON SPEGNERE IL DEVICE.
4. Una volta finita l'operazione i led 1 e 2 resteranno accesi per circa un minuto e il sistema ripartirà.
5. Per valutare il successo dell'operazione collegarsi con l'interfaccia WWW e confrontare a più di pagina la versione del Firmware.

4.5.3 Aggiornamento via SFTP

Una connessi via SFTP è sufficiente andare nella cartella

$$\setminus\ tmp\ \setminus\ obex\ \setminus$$

e copiare il file e attendere come descritto sopra.

4.5.4 Aggiornamento via shell testuale (SSH o SERIALE)

Usando il comando wpkgd è possibile installare la versione più recente di ogni pacchetto. Il device scaricherà il pacchetto richiesto.

1. É necessario che il device sia collegato sia collegato in rete.
2. Collegarsi alla shell usando un client ssh o il cavo serial
3. In root digitare il comando: <wpkgd install nome_pacchetto>

Se volessimo aggiornare un access server:

[root@wrap root]\$ wpkgd install reflash Downloading
http://update.xxxx.com/as/4.0/lt/reflash-4.0-1.if.wpk

Come detto precedetemente questo tipo di accesso permette delle funzioni avanzate.

1. wpkgd search <nome_pacchetto>: ricerca se il pacchetto è stato installato

2. wpkgd search <keyword> : ricerca un pacchetto contenente la parola chiave
3. wpkgd erase <nome_pacchetto> (wpkgd -e) : cancella un pacchetto
4. wpkgd update <nome_pacchetto>: aggiorna un pacchetto alla versione più recente
5. wpkgd update: aggiorna tutti i pacchetti.
6. wpkgd list (wpkgd -l): fornisce una lista dei pacchetti disponibili per il download, indicando quali sono già installati.

4.5.5 Regolare l'ora

Il device riconosce automaticamente l'ora al momento della connessione e consultabile attraverso il percorso:

Access Point → *Setup* → *Generic*

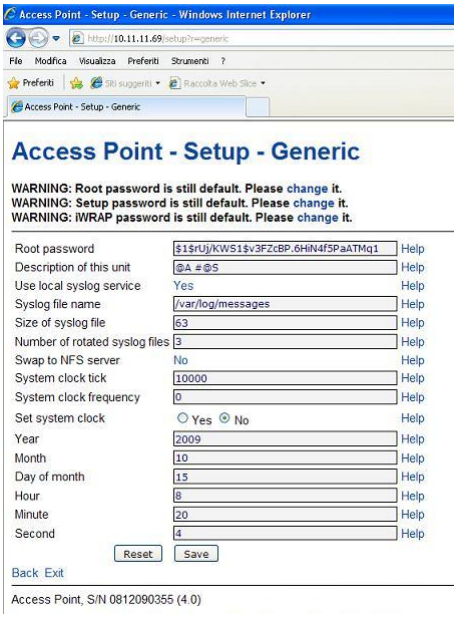

Figura 4.9: Menù di setup, Interfaccia web

Purtroppo il device non riconosce il diverso fuso orario, e non è possibile configurarlo dalla pagina mostrata in figura 4.9, la procedura è molto simile a quella usata per aggiornare il SW.

1. Accedere alla pagina

Access Point → *Setup* → *Advanced Settings* → *Upload a software update*

2. É necessario identificare il giusto file di configurazione da caricare. Dal file zip SOFTWARE RELEASE disponibile nel sito del fornitore, entrare nella cartella Timezone_packets e cercare il file

   tzdata-1.2009f-1.europe.rome.noarch.wpk

3. Caricare il file così trovato.

Naturalmente è possibile settare lì'ora anche attraverso il client SFTP, ma per semplicità si preferisce presentare solamente il modo via interfaccia WWW

4.5.6 Defaul startup application

Per consultare i servizi attivi dopo l'accensione è sufficiente andare alla pagina:

Access Point → *Setup* → *Applications* → *Default startup applications*

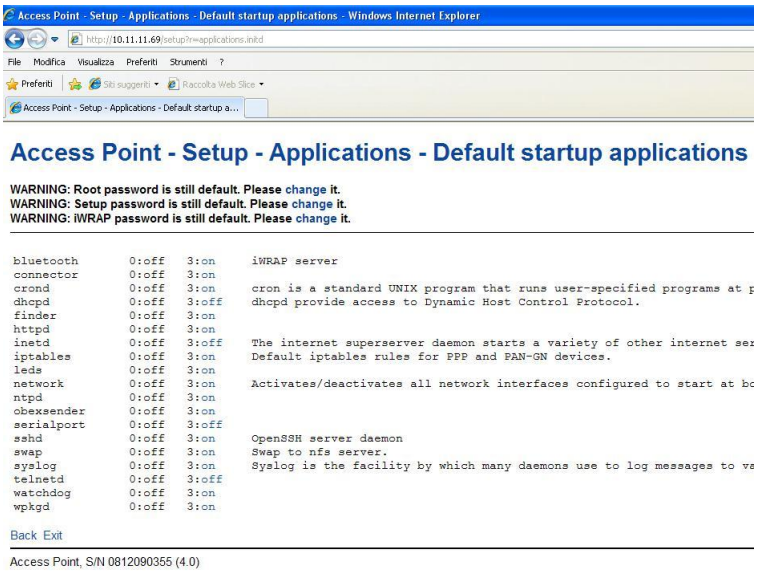

Figura 4.10: Elenco dei servizi attivati all'accensione

4.5.7 Installazione di un dispositivo di memoria esterno

E' possibile collegare una chiavetta USB (fino a 8 GB), o una Compact Flash Card all'Access Point. Le possibilità messe a disposizione dal collegamento sono:

1. Memoria per i file ricevuti e i log

Server:	Descrizione
Bluetooth	iWRAP Server, servizio che gestisce la connettività BT
Connector	Mantiene le connessioni con i specifici cellulari Bluetooth
Crond	Demon che schedula l'esecuzione di prcessi, editabile a var/spoll/cron/crontabs/root
Dhcpd	Demone DHCP che permette la configurazione automatica della rere. Può essere disabilitato attraverso l'interfaccia WWW come vedremo nella sezione SCENARI INTERMEDI per usare un indirizzo IP fisso.
Crond	Demon che schedula l'esecuzione di prcessi, editabile a var/spoll/cron/crontabs/root
Finder	Permette di cercare altri AP o AS connessi alla rete per la realizzazioni di soluzioni avanzatet
Iptables	Non un server, ma una serie di regole di IP filtering
Leds	Non un server, ma uno script che controlla i led all'accensione o dopo il reboot
Network	Non un server, ma uno script di controllo
Ntpd	Demone per il Network Time Protocol (NTP).
Obexsender	ObexSender. Applicazione che invia i contenuti via bluetooth
Openvpn	Soluzione via SSL VPN. Non installata di default
Pppd	Demone per il Point protocollo to Point
Serialport	Servizio per l'abilitazione della presa seriale
Smsgw	SMS gateway
Sshd	Demone SSH
Swap	Script per far partire o stoppare NFS server
Syslog	Demone per la gestione dei log. Configurabile attraverso WWW
Telnetd	Protocollo Telnet. Disabilitato di default per ragioni di sicurezza.Abilitabile dalla shell.
Udhcpcd	Demone DHCP per la configurazione automatica della rete
Watchdog	Sistema di temporizzazione HW per evitare loop infiniti
Wpkgd	Demone per la gestione dei pacchetti wpk. Definito spesso come: Remote management system daemon.
Zcip	Zero configuration networking service

Tabella 4.2: Lista dei servizi selezionabile allo startup

2. Memoria per i contenuti avanzati da inviare.

3. Gestione della campagna via memoria esterna.

Come detto in precedenza la macchina ha in background un ambiente Linux, quindi l'operazione è del tutto simile a istallare una periferica esterna in un PC Linux. Tramite la pagina:

Access Point → *Setup* → *Advanced* → *System startup script*

si accede a una sezione che gestisce la azioni del device dopo l'accensione. Lo script 4.1 permette di connettere una chiavetta Usb al dispositivo.

Script 4.1 ***Mount di una chiavetta USB***

```
mkdir -p /mnt/disk
mount -t vfat /dev/sda1 /mnt/disk
if [ $? != 0 ]; then

fsck.vfat -a /dev/sda1
mount -t vfat /dev/sda1 /mnt/disk
fi
service obexsender restart
```

Copiandolo in questa sezione si ottiene l'effetto desiderato. In caso sia tolta la corrente con la chiavetta ancora in stato mount alcuni file potrebbero corrompersi, il comando fsck.vfat serve proprio a quello. Infine si resetti l'Access point dalla pagina:

Access Point → *Setup* → *Advanced Setup*

con il comando: **Reboot System (confirm)**. Nel caso la chiavetta USB fosse inserita prima di accendere l'AP questo ultimo passo non è necessario.

Per quanto riguarda la nominazione delle chiavette USB, e delle memorie Compact Flash, AP le rinomina con le seguenti regole.

- La prima chiavetta USB inserita dopo un reboot sarà **/dev/sda1** se la chiavetta ha un file system (la maggior parte dei casi).
- La prima memoria Compact Flash prenderà il nome di **/dev/hda1**.
- Nel caso si inseriscano diverse memorie esterne dopo il reboot l'ultima lettera cambierà seguendo l'ordine alfabetico (ad esempio la seconda chiave sarà /dev/sdbb1, la terza chiave sarà dev/sdc1 e così via) [20].

4.6 ObexSender

4.6.1 Introduzione

Il cuore principale del device è "OBEXSENDER" e permette l'invio dei messaggi attraverso la tecnologia BT raggiungibile attraverso il percorso:

$$Access\ Point \rightarrow Setup \rightarrow Applications \rightarrow ObexSender$$

Da questa pagina sono accessibili tutte le funzioni di base, per cui non è necessaria una infrastruttura di rete esterna o usare il Software Development Kit.

4.6.2 ObexSender Configuration

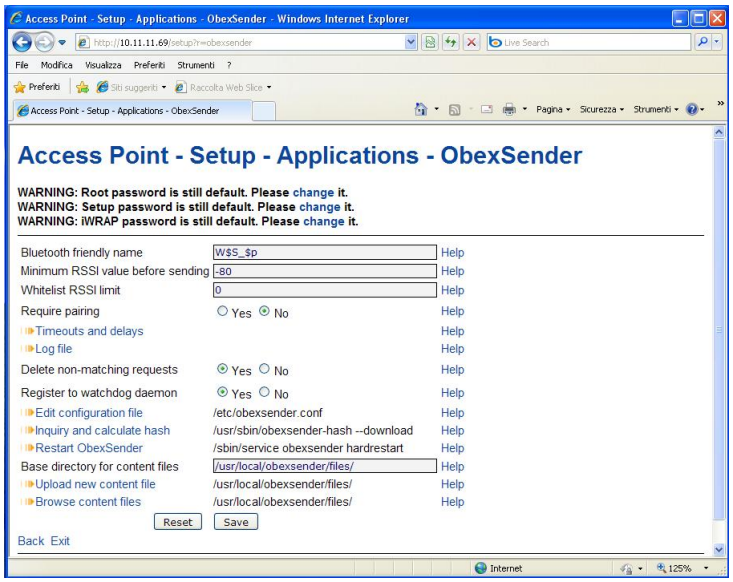

Figura 4.11: Menù di ObexSender

Settaggio di Base

Bluetooth friendly name [string]
É il nome con il quale i dispositivi Bluetooth identificano l'Access Server.
Minimum RSSI value before sending [integer]
Si può configurare o limitare il raggio operative di ObexSender. Quando ObexSender ricerca i dispositivi ne misura l'RSSI (Receiver Signal Strength Indicator). Questo valore è compreso tra -128 e -1.

- Valori di RSSI da -128 a -90 significano che il segnale è estremamente debole e un tentativo di connessione fallirebbe.

- Volori di RSSI da -80 a -65 significano che il segnale è buono. Può essere stabilita una connessione. Con dispositive Bluetooth di classe 2, quasi tutti i cellulari, significa che il dispositivo è a 10-20 metri di distanza. Un dispositivo di classe 1 (come un Access Server) può trovarsi anche a più di 100 metri di distanza.

- Valori di RSSI da -45 a -30 significa che il segnale è molto forte. I dispositivi bluetooth sono molto vicini all'Access Server (meno di un metro di distanza).

Whitelist RSSI [integer]
Questo settaggio determina il limite di RSSI che permette la rimozione del dispositivo da tutte le block list. Può essere usato in modo che se si porta il dispositivo molto vicino all'Access Server è libero di ricevere contenuti di nuovo senza bisogno di aspettare l'Ok o il Fail delay. Per esempio se si riceve con successo un file da ObexSender bisogna aspettare un Ok delay per poter ricevere di nuovo il file. Tutta via se si è configurato il whitelist RSSI limit per esempio a - 45 e si passa vicino all'Access Server, questo dispositivo sarà rimosso dalla block list e potrà ricevere di nuovo il file. Per essere rimosso dalla block list bisogna avvicinare il cellulare all'Access Server almeno entro un periodo di scan (ricerca dispositivi).
Require pairing [selection]
Se impostato su Yes, i dispostivi devono effettuare il pair con l'Access Server per ricevere i file. Gli utenti devono sapere il codice PIN Bluetooth configurato per fare il pairing.
Delete non-matching requests [selection]
Se abilitato, ObexSender cancellerà i file inviati come richiesta che non hanno trovato nessun match con le procedure reply presenti. Se disabilitato i file verranno salvati nell'Access Server.
Register to watchdog daemon [selection]
Se abilitato, ObexSender procederà automaticamente al riavvio dell' Access Server in caso di un fallimento nelle funzionalità di ObexSender.
Upload a new content file [link]
Questo link serve per caricare i file nella memoria dell'Access Server. Attraverso il pulsante sfoglia è possibile scegliere quale file dal proprio PC caricare.
Browse content files [link]
Questo link serve per visionare i file presenti nella memoria dell'Access Server.
Edit configuration file [link]
Questo link serve per accedere alla pagina di modifica del file di configurazione

di ObexSender.

Inquiry and calculate hash [link]

Questo link fa un indagine sui dispositive Bluetooth vicini e visibili e ne calcola il valore hash di riconoscimento. Questo link è usato per inserire nuovi dispositivi nel database di ObexSender.

Restart ObexSender [link]

Riavvia l'applicazione ObexSender.

View log [link]

Mostra su una pagina web il contenuto del file log di ObexSender. Per visionare il file log direttamente via sftp si rimanda ai test.

Delete log [link]

Cancella il file log di ObexSender. Necessita conferma.

Timeout and delays

Delay between inquiries [seconds]

Il ritardo tra successive scan (ricerca di dispositive Bluetooth) in secondi.

If previous was ok, timeout before sending again [seconds]

Se il file è stato mandato con successo a un dispositivo, questo timeout stabilisce quando si può rimandare contenuto al medesimo dispositivo.

If previous was fail, timeout before sending again [seconds]

Se è fallita la trasmissione del file a un dispositivo oppure l'utente non ha accettato il file, questo timeout stabilisce quando ObexSender può rimandare contenuti allo stesso dispositivo.

Delay between retrying call [seconds]

Quando l'utente non accetta o respinge il file, ObexSender riprova automaticamente a mandare il file. Questo messaggio determina quanto tempo aspetta prima di riprovarci. È utile nel caso l'utente non abbia notato la trasmissione Bluetooth in entrata. Il valore di default è di 120 secondi perché molti dispositivi come i cellulari lasciano il link Bluetooth aperto per questo periodo prima di chiuderlo.

Delay after scanning [seconds]

Quando viene ricevuta una richiesta remota da un utente, questo settaggio determina quanto aspettare affinché il file di risposta venga inviato al dispositivo. Il valore di default è 5 secondi perché molti dispositivi come i cellulari possono ricevere file tramite Bluetooth solo dopo 5 secondi dall'invio (si intende inviati usando bluetooth).

Tester delay [seconds]

Questo campo stabilisce quanto spesso un file viene inviato a un dispositivo Blue-

tooth "tester". I dispositivi bluetooth tester sono identificati tramite del codice nel file di configurazione di ObexSender (vedremo in seguito come).

Pair expire timeout [seconds]

Questo settaggio definisce per quanto tempo l'Access Server tiene in memoria le informazioni per il pairing. È utile nel caso gli utenti debbano fare il pairing per ricevere i file.Valore 0 equivale a tenerli per sempre.

Local content management

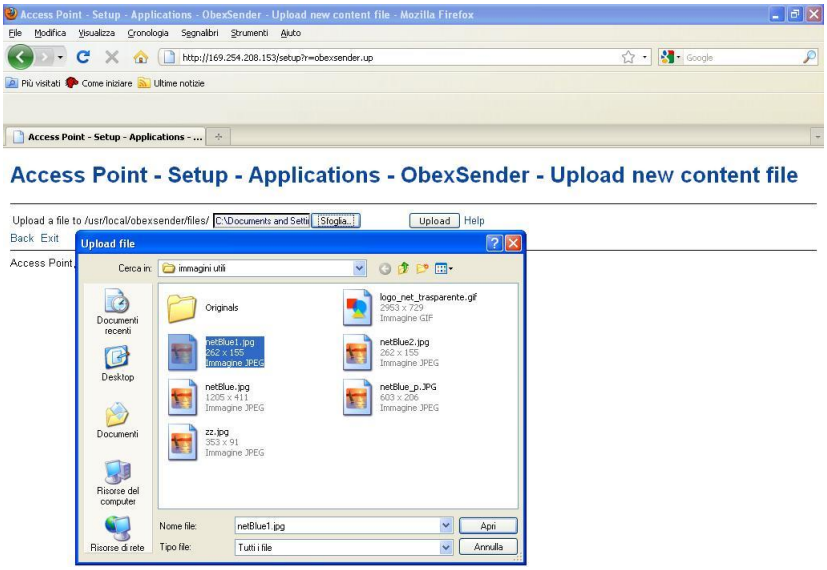

Figura 4.12: Schermata upload file

I contenuti possono essere caricati attraverso la pagina di confgurazione. Premendo il tasto Browse si apre una finestra di dialogo per capirare i file selezionati dal PC nel file sistem del device. É possibile eseguire l'upload di un contenuto anche via SFTP e SSH, oppure utilizzando una chiavetta USB o un disco fissso.

Note: I nomi dei file non possono contenere spazi vuoti, inoltre di ricorda che l'ambiente Linux è case-sentitive, quindi è necessario prestare attenzione alla maiuscole e alle minuscole

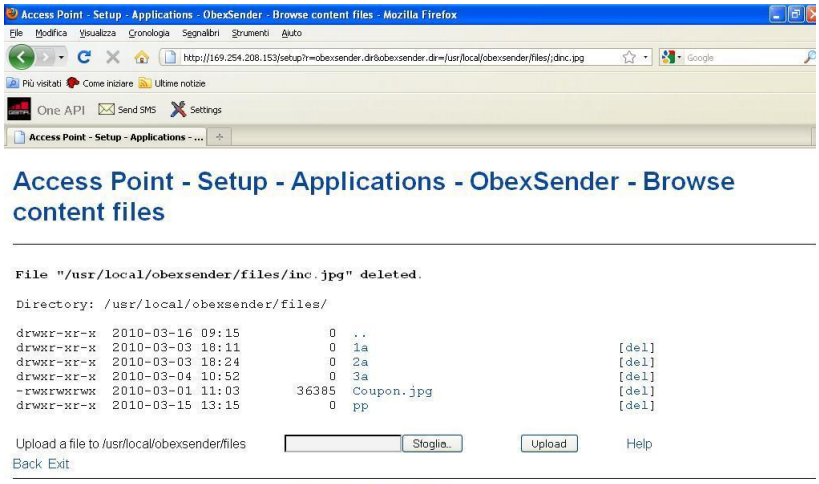

Figura 4.13: Schermata dei contenuti caricati

4.7 File di Configurazione

4.7.1 Funzioni di base

In questa sezinoe vengono presentati i comandi di base per il controllo di ObexSender

Send

Estratto del file Obexsender.conf 4.1 ***Sintassi del comando di Send***

```
send {
          label <name>              (optional, defaults to "start")
           match <regex>             (optional)
           time <timestamp>          (optional)
           file <filename>
           exec <command>
           goto <ok-time> <label>   (optional)
         }
```

Attraverso questa sintassi è possbile creare una sendig rule che permetta l'invio di contenuti a cellulari nelle vicinanze con le seguenti opzioni:

- Match: con questa opzioni è possibile restringere il target dei cellulari ai quali sarà inviato il contenuto.
- Time: questo comando veicola l'invio dei messaggi ad una specifica finestra temporale

- Exec permette lanciare le applicazioni scritte attrvars il Software Development kit tool, per ulteriori dettagli si consulti il capitolo 6.
- Label e goto infine permettono di creare delle concatenazioni di messaggi: se un utente riceve correttamente la prima tranche di file, dopo un ritardo prestabilito l'access point tenta di inviarli la tranche successiva.

Reply

Estratto del file Obexsender.conf 4.2 ***Sintassi del comando di Reply***

```
reply {
           keyword <regex>          (optional)
           match <regex>            (optional)
           time <timestamp>         (optional)
           file <filename>
           exec <command>
           goto <ok-time> <label>   (optional)
         }
```

L'access Point è anche in grado di ricevere messaggi da parte di altri cellulari e rispondere in modo adeguato ad una richiesta. Attraverso il comando reply è possibile comandare questo tipo di risposta. Una volta che il file viene ricevuto viene ricercata la keyowrd, e se presente, si attiva la reply rule. É possibile inserire nello stesso file di configurazione diverse reply rule, soluzione utilizzata per creare applicazioni avanzate, per ulteriori dettagli si consulti il capitolo 7.

4.7.2 Altri parametri gestibili da questa pagina

Il file di configurazion contiene alcuni settings avanzati, che non sono editabili dalla pagina di setup attraverso l'interfaccia web. É quindi necessario editare manualmente questo file a questo indirizzo

Access Point → *Setup* → *Applications* → *ObexSender* → *Edit configuration file*

basedir [directory]

Basedir indica la cartella principale dove si trovano tutti i file di contenuti che ObexSender deve inviare.

La cartella di default è : basedir /usr/local/obexsender/files/

baseband [IP-addr] [port] [password]
Questa opzione permette di scegliere quali radio Bluetooth sono usate dall'ObexSender per inviare i file. Di default sono in uso tutte le radio, ma si può cambiare la configurazione commentando le linee superflue.
baseband 127.0.0.1 10101
baseband 127.0.0.1 10102
baseband 127.0.0.1 10103
baseband 127.0.0.1 10104

Baseband 10104 può essere usata solo per funzioni di test e non interferisce con la funzionalità. Nel caso sia abilitato iWRAP password nei settaggi di sicurezza allora anche i client locali necessitano di usarla e bisogna includere la password anche nel file di configurazione di ObexSender.

ignore [BD_ADDR]
Con questa configurazione si possono definire gli indirizzi Bluetooth da bloccare in modo che ObexSender non invii file a questi indirizzi. Nella configurazione di default è bloccato il range di indirizzi del fornitore, così che gli Access Server non possano inviarsi file tra loro. Default: *ignore 00:07:80:* Si possono inserire diverse righe se si vuole ignorare anche altri range.

tester [BD_ADDR]
ObexSender invierà sempre ai dispositivi definiti con questo settaggio. Tester timeout configura quanto spesso vengono mandati file ai dispositivi test. Ci possono essere diversi tester

scandir [path] Questa configurazione definisce la cartella che ObexSender deve scansionare per delle richieste remote. Questa cartella dovrebbe essere la cartella root di ObexSender: *scandir @* @ è un tag speciale e riferisce alla cartella root di Obexserver.

broadcast [target] Questa configurazione stabilisce a quale interfaccia network ObexSender manda informazioni sui disposistivi Bluetooth serviti. Questa configurazione può essere usata in una rete di Access Server in cui un nodo vuole prevenire che un utente riceva gli stessi file più volte. L'obiettivo può essere un unicast o un broadcast IP-address oppure un nome interfaccia di una rete. La configurazione di default riferisce ad un interfaccia network Ethernet: broadcast nap. Si veda la sezione SCENARI INTERMEDI per ulteriori dettagli.

uuid [uuid[,uuid[,...]]] Questa configurazione stabilisce quali sono i Bluetooth profiles di ObexSender per trasmettere file. La configurazione di default non dovrebbe essere cambiata ed è: uuid OBEXOBJECTPUSH,OBEXFILETRANSFER.

Gli uuid sono identificatori univoci universali a 128 bit.

4.7.3 Memorizzare i file ricevuti

La funzione reply permette di analizzare in maniera elementare il nome di un file inviato e se è il caso, di rispondere con un contenuto adeguato. Può essere interessante mantenere i file in memoria, gli scenari in cui questa features potrebbe esser utile sono molti; si pensi a una discoteca in cui gli utenti scattano delle foto con il cellulare per pubblicarle su un maxi schermo, oppure per esempio, si può utilizzare questa funzionalità per richiedere un ticket numerato. La procedura più semplice è la seguente:

1. Accedere attraverso una connessione WWW alla pagina

$$Access\ Point \rightarrow Setup \rightarrow iWRAP \rightarrow Profiles \rightarrow ObjP$$

2. Modificare la voce Optional parameters for server dalla configurazione di default: *–bdaddr $b –prefix $b-$P-* in questa:
"–bdaddr $b –prefix $b-$P- –fork '/bin/cp $$t /tmp/$$p$$d-$$T'"

4.7.4 Comportamento con molte connessioni

Come detto in precedenza AP riesce a trattare contemporaneamente fino a 7 connessioni BT. L'obiettivo di questo paragrafo è di descrivere il comportamento del device in presenza di dispositivi oltre la sua capacità. La specifica Bluetooth permette di creare una connessione con solo 8 nodi; per questo motivo l'access point può essere connesso solamente con 7 dispositivi contemporaneamente. Al momento dello scan ogni cellulare nel raggio viene registrato nel file di log e ObexSender sceglierà da questa lista di possibili clienti un telefonino da servire. Vengono lanciati quindi tanti thread, fino ad un massimo di 7, quanti sono i possibili cellulari da servire. Come potrebbe sembrare logico, quando un thread finisce di servire un cellulare non cerca di servire un altro telefonino, ma attende la successiva operazione di Scan. Quando la nuova operazione di scan la lista dei telefonini nel raggio sarà aggiornata con i cellulari già serviti, e saranno selezionati nuovamente da questa lista i cellulari da servire. Ciò avviene per non tentare di servire un cellulare che potrebbe essere uscito dal raggio dell'access point. La scelta del dispositivo da scegliere è casuale, le future versioni del SW permetteranno di assegnare delle priorità ai dispositivi da servire.

4.8 Gestione dei log

Nella configurazione di default il log non sono memorizzati in modo permanente, ma sono consultabili in ogni momento, finchè la macchina sia alimentata attraverso l'interfaccia WWW nella pagina:

Access Point → *Setup* → *Applications* → *ObexSender* → *Log* → *View log*

4.8.1 Opzioni dei log

Log file name
Permette di rinominare il file di log, per il pathname non è necessario cambiare il nome in questo ma è sufficiente modificare il file di configurazione come mostrato in seguito.

Log prefix
Prefisso messo all'inizio di ogni riga del log creata da OBEXSender. Utile in caso di log particolarmente complessi.

If sending was failure, log it too
Se attivata nel log compariranno anche le transazioni non completate con successo

Verbosity level
Regola il dettaglio di dettaglio del log da 0 (minimo) a 4 (massimo). Si consiglia di selezionare 0 o 1.

Block list save delay
La block list contiene i device che hanno già ricevuto un contenuto informativo o lo hanno rifiutato o stanno aspettando per una trasmissione. Con il valore 0 si disabilita la feature, se si desidera attivare si consiglia di scegliere un tempo relativamente grande, nell'ordine di 900 (15 minuti).

Block list file name
Pathname della block list appena descritta.

4.8.2 Come leggere i log

A livello verbosity log=0 i messaggi sono molto sintetici, il che li rende ottimi in regime di funzionamento standard ma non efficaci durante lo sviluppo. Dopo un operazione di scan ogni cellulare nel raggio viene classificato in questi modi:
Device found: Dispositivo non servito, con un segnale di risposta abbastanza potente.
Device blocked: Dispositivo già contattato, può aver accettato, rifiutato o

ignorato la risposta, il comportamento dell'Access Point sarà determinato dai valori

- If previous was ok, timeout before sending again
- If previous was fail, timeout before sending again
- Delay between retrying call che vedremo successivamente.

É possibile vedere il conteggio dei secondi mancanti per il rinvio di un qualche contenuto a quel cellulare.

Device too weak: Dispositivo che risponde con un segnale troppo debole per iniziare una comunicazione. Probabilmente troppo distante, non sarà considerato fino alla prossima operazione di scan. La soglia può essere modificata.

Device already active: Dispositivo che sta effettuando una connessione con il device.

Queste voci sono sempre accompagnate dalla indicazione RSSI, e possono essere eliminate dai log attraverso l'opzione <Inquiry log No> nel file di configurazione o alla pagina:

$$Access\ Point \rightarrow Setup \rightarrow Applications \rightarrow ObexSender \rightarrow Log$$

Ex:

Estatto di log 4.1 ***Esempio di log a livello 0***

```
Device found: 00:1a:89:84:ea:bc (RSSI -58) \\
Device blocked: 00:1e:3b:a5:e8:5f (RSSI -53, 35978s)\\
Device too weak: 00:0e:07:c4:ad:bc (RSSI -88)
```

A *livello 0* per segnalare il successo o l'insuccesso di una transazione si usa un codice numerico:

- OK/0
  Il file di configurazione è coerente, il contenuto è stato trovato nel filesystem e inviato con successo. In caso la regola di invio imponga la trasmissione di più file il messaggio sarà MORE/0, e solo all'ultimo file sarà associato il codice OK/0. Quindi MORE/0 significa che fino a quel momento tutti i file sono stati inviati con successo, e che si sta cercando di trasmettere il file successivo.

- FAIL/1
  Password sbagliata, o errori interni a ObexSender, errore critico, se le password sono state settate correttamente conttattare il supporto tecnico

- FAIL/2
  Errore fatale, ObexSender non viene evocato correttamente. Causa il riavvio della macchina.

- RETRY/3
  L'utente non ha notato il prompt per accettare il file in ingresso. Dal lato del cellulare la connessione viene interrota automaticamente dal cellulare stesso se l'utente non accetta ne rifiuta il contenuto. Se questi messaggio sono troppo frequenti è bene riconsiderare la posizione fisica del dispositivo, il valore RSSminimo da usare, le dimensioni dei file che vengono inviati e quanto bene si sta informando il pubblico della campagna USB.

- FAIL/4
  L'utente ha rifiutato volontariamente il contenuto

- FAIL/5
  Il device è riuscito a riconoscere il cellulare e realizzare una connessione, ma non è stato trovato il contenuto da inviare. Spesso ciò significa che il nome del file salvato nel device è diverso da quello usato nel file di configurazione.

- FAIL/6
  Come il caso precedente ma usato nel caso di SEND con multipli contenuti, ciò significa che i file precedenti sono stati inviati con successo.

- FAIL/7
  Nel dispositivo obiettivo non sono stati trovati ne OPP (Object Push Profile) ne FTP (File Transfer Protocol). Appare anche se alcuni altri UUID rispetto a OPP o FTP erano configurati in ObexSender.

A *livello 2* si aggiungono altri messaggi, con un dettaglio maggiore.

- 104 HCI_ERR_PAGE_TIMEOUT: Dopo aver effettuato una connessione con successo il cellulare non risponde più facendo scadere la richista di invio. La causa più probabile è che l'utente sia uscito dall'area del device durante la connessione stessa, o nel periorodo tra la scan e l'operazione di call. Riportato come RETRY/3

- 105 HCI_ERR_AUTHENTICATION_FAILED: Capita in occasione del mismacth del PIN o durante un errore di pairing. Riportato come FAIL/4

- 106 HCI_ERR_KEY_MISSING: Errore nel PIN, usato solamente quanto è certo che si sia verifiacato un errore nell'immissione del PIN. Riporatto come FAIL/4

- 108 HCI_ERR_CONNECTION_TIMEOUT: La connessione è stata stabilita, ma ad un certo punto il cellulare ha smesso di rispondere per causa non identificata. Potrebbe essere uscito dal raggio, aver spento il cellulare o il Bluetooth. Riportato come RETRY/3

- 110 HCI_ERR_HOST_TIMEOUT: Timeout dal lato cellulare, non si è ricevuta nesuna risposta durante il tentativo di connessione, e la connessione è stata terminata. Il cellulare non è uscito dal raggio del device. Riportato come RETRY/3

- 122 HCI_ERR_LMP_RESPONSE_TIMEOUT: Errore di connessione a livello LMP. Riportato come FAIL/4

- 464 RFCOMM_ERR_REJECTED: L'utente ha rifiutato il contenuto, o non aveva abbastanza risorse per ricevere il contenuto. Riportato come FAIL/4

- 465 RFCOMM_ERR_TIMEOUT: Fallimento a livello RFCOMM, probabilmente l'utente non ha notato il prompt per la connessione. Riportato come RETRY/3

A *livello 4* abbiamo il massimo grado di dettaglio del log. Durante la fase di accensione possiamo vedere l'attivazione di alcuni servizi e molte delle opzioni gestibili dall'utente. Nel dettaglio:

Estatto di log 4.2 ***Primo esempio di log di livello 4***

```
 Input 7: - iWRAP> SET BLUETOOTH BDADDR 00:07:80:90:96:04
 Input 7: - iWRAP> SET BLUETOOTH NAME "Demo\_Ne-t"\\
 Input 7: - iWRAP> SET BLUETOOTH LISTEN 3 "/usr/sbin/obexserver
--bdaddr \$b --prefix \$b-\$P-" 256 5\\
```

Inoltre la comunicazione viene descritta con un livello di dettaglio maggiore: dal tentativo di connessione con il telefonino:

Estatto di log 4.3 ***Secondo esempio di log di livello 4***

```
Input 7: Calling: 00:22:66:da:31:6a 9 ObjP
Input 7: - iWRAP< LOCK
iWRAP< CALL 00:22:66:da:31:6a 9 RFCOMM
Input 7: - iWRAP> CALL 1
Input 7: - iWRAP> RINGING 1
Input 7: - iWRAP< UNLOCK
Input 7: - iWRAP< CLOSE 0
Input 7: - iWRAP> NO CARRIER 0 ERROR 000 IN=0,OUT=0,ELAPSED=4
 Input 7: - iWRAP> CONNECT 1 RFCOMM 36104
Connected: 00:22:66:da:31:6a ObjP
```

fino all'invio dei contenuti:

Estatto di log 4.4 ***Terzo esempio di log di livello4***

```
Input 7: - Exec: /usr/sbin/obexsender.script
Input 7: Sending: /usr/local/obexsender/files/netBlue.jpg 0/38511 bytes
Input 7: - OBEX> Continue|Final
Input 7: Sending: /usr/local/obexsender/files/netBlue.jpg 10708/38511 bytes
Input 7: - OBEX> Continue|Final
Input 7: Sending: /usr/local/obexsender/files/netBlue.jpg 21462/38511 bytes
Input 7: - OBEX> Continue|Final
Input 7: Sending: /usr/local/obexsender/files/netBlue.jpg 32216/38511 bytes
Input 7: - OBEX> Success|Final
Input 7: Sent: /usr/local/obexsender/files/netBlue.jpg 38511 bytes
Input 7: - OBEX< Disconnect Input 7: - OBEX> Success|Final
Input 7: - iWRAP< CLOSE 1
Input 7: - iWRAP> NO CARRIER 1 ERROR 000 IN=22,OUT=38591,ELAPSED=3
Input 7: Info: All done
Input 7: - iWRAP< QUIT
Input 7: Exit: OK 0
```

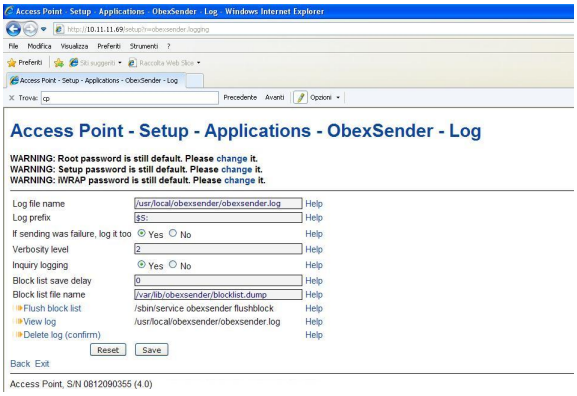

Figura 4.14: Menù per la gestione dei log

4.8.3 Memorizzare in modo permanente i log

Questa operazione permette di salvare all'interno dell'Access Point i log in modo che siano reperibili anche dopo lo spegnimento del dispositivo. Attraverso l'interfaccia WWW editare il file: /etc/obexsender.conf. Nella configurazione di Default l'istruzione per il controllo dei log si presenta come: log -, che indica che il file verrà cancellato a ogni reboot. É necessario agire su questa istruzione:

log /pathname

e come consiglia il supporto clienti si può scrivere:

log /usr/local/obexsender/obexsender.log

In questo modo accedendo attraverso l'interfaccia SFTP il logo sarà all'indirizzo:

/usr/local/obexsender/obexsender.log

4.8.4 Memorizzare in modo permanente i log in un dispositivo di memoria esterno

L'operazione si divide in due fasi già illustrate:

1. Istallazione dei un dispositivo di memoria esterno, una chiavetta USB.

2. Indicazione destinazione log.

Entrambe le procedure sono state precedentemente affrontate si fornisce quindi un esempio tipico Si supponga di avere inserito solo una chiavetta USB dal reboot e che in questa si desideri salvare i file di log. Si accede alla pagina

Access Point → *Setup* → *Advanced* → *System startup script*

e si copi il seguente codice nello **script 4.1**

Dalla pagina precedente fare il reboot del sistema Per verificare il successo dell'operazione è possibile consultare la cartella /mnt/disk

- Attraverso l' interfaccia WWW
- Attraverso l'interfaccia SSH o SFTP navigando nel file system.

Successivamente accedere al file di configurazione

/etc/obexsender.conf

tramite interfaccia WWW o SFTP e cambiare la riga di comando

log-

con

log/mnt/disk/usb.log

4.8.5 Lista dei cellulari serviti

Per funzioni che vedremo nella seconda parte con maggiore dettaglio gli Access Point registrano in una lista particolare i cellulari che hanno servito. La lista è all'indirizzo

/var/lib/obexsender/blocklist.dump

raggiungibile solamente tramite un client ssp o ssh (winscp432, putty). Come descritto precedentemente il device dopo l'operazione di scan controlla la lista di cellulari serviti, sceglie poi tra i cellulari nella zona e non ancora serviti, i cellulari a cui inviare i contenuti. Subito dopo aver servito il cellulare la lista viene aggiornata in una memoria volatile; i dati qui contenuti vengono persi al momento dello spegnimento. Al momento dell'accesione del device, (o in caso di

reboot), il dispositivo cerca la lista, ma per le impostazioni di default, questa non esiste, come possiamo vedere in questo breve log.

Estatto di log 4.5

```
20091018/145453 0812090355: ObexSender startup
20091018/145453 0812090355: Can't load block list from
/var/lib/obexsender/blocklist.dump
20091018/145500 0812090355: Connected to iWRAP 127.0.0.1:10101
20091018/145505 0812090355: Device found: 00:1e:45:c9:f7:4f (RSSI -55)
```

Nella configurazione di default infatti la dump list non viene salvata, ciò avviene per motivi di sicurezza. Infatti il tipo di memoria non volatile di AP non è stata realizzata per sopportare una frequente scrittura. É possibile però attraverso la pagina:

Access Point → *Setup* → *Applications* → *ObexSender* → *Log*

agire sulla voce *Block list save delay*. Questo timer comanda il passaggio da memoria volatile a quella non volatile della lista dei cellulari serviti. Questo sistema non è infallibile, come dimostra l'immagine 4.15. Si immagini il seguente scenario:

- T=0; viene inviato un messaggio con successo al cellulare1, la transizione viene registrata nella memoria voltatile
- T=1; avviene il salvataggio della dump list in memoria non volatile, nel dettaglio viene salvata la transizione del cellulare1
- T=2 viene inviato un messaggio al cellulare 2
- T=3; l'access point si spegne per un qualsiasi motivo
- T=4; l'access point si riaccende e carica dalla memoria non volatile la lista dei cellulari serviti. Entrambi i cellulari sono nella zona di cattura dell'access point, ma al momento dello spegnimento la transizione con il cellulare_1 era già stata salvata nella memoria non volatile, e per questo non verrà riservito. Diversamente la comunicazione con il cellulare_2 era stata era stata registrata solo nella memoria volatile che si è persa quando il device si è spento, per questo motivo il cellulare2 è come se non fosse stato mai servito, e l'access point tenterà di inviarli nuovamente lo stesso messaggio

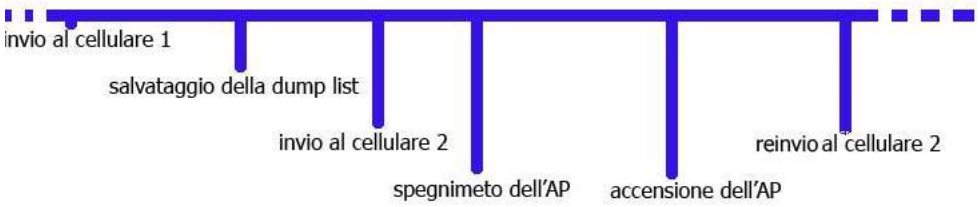

Figura 4.15: Effetto del ritardo del salvataggio della Block list

4.9 Funzioni Intermedie

4.9.1 Utilizzare più di un singolo Access Point

Subito dopo aver registrato la limitata capacità riguardo il numero di connessioni contemporanee si è sentita l'esigenza di superare questa barriera. Non è difficile immaginare scenari in cui questo valore massimo inficerebbe il successo della campagna, anzi gli scenari più interessanti sono quelli in cui in un breve intervallo di tempo si concentrano molte persone, quindi molti potenziali utenti del proximity marketing (intervallo in una partita di calcio, fermata di autobus o tram). Come detto precedentemente i device durante l'operazione di SEND aggiornano una lista con tutti i cellulari, palmari, smart phone, portatili già serviti, in modo da non rinviare gli stessi contenuti agli utenti che li abbiano già ricevuti. É quindi fondamentale costruire una struttura per permettere ai device di comunicare tra di loro per scambiarsi queste liste. Il modo più semplice per permettere ai device è quello di connetterli in rete, attraverso l'uso di normali cavi di rete, ma prima di proseguire è necessario una breve introduzione sugli indirizzi da assegnare.

4.9.2 Indirizzi IP statici e dinamici

Un indirizzo IP è un numero che identifica univocamente i dispositivi collegati in una rete informatica che utilizza lo standard IP (Internet Protocol). Ciascun dispositivo (router, computer, server di rete, stampanti, alcuni tipi di telefoni,...) ha, quindi, il proprio indirizzo IP che può essere visto come l'equivalente di un indirizzo stradale o un numero telefonico. Infatti, così come un indirizzo stradale o un numero telefonico identifica un edificio o un telefono, così un indirizzo IP identifica univocamente uno specifico access point, access server o un qualsiasi altro dispositivo di rete o una rete. Gli indirizzi IP però non sono intrinsechi dell'oggetto, ma dipendono dalla rete, quindi un access point può avere l'indirizzo

A nella rete 1 e l'indirizzo B nella rete 2. Gli indirizzi IP possono essere assegnati in maniera permanente oppure in maniera temporanea, da un insieme di indirizzi disponibili.

- Indirizzi dinamici
  Gli indirizzi dinamici vengono utilizzati per identificare dispositivi non permanenti come ad esempio un personal compuer. L'indirizzamento dinamico richiede la presenza di un server in grado di accettare richieste DCHP ed assegnare indirizzi. L'indirizzamento può avvenire in maniera casuale, oppure in base ad una determinata politica.

- Indirizzi statici
  Gli indirizzi statici vengono utilizzati per identificare dispositivi semi-permanenti con indirizzo IP costante. L'indirizzo statico può essere configurato direttamente sul dispositivo, come vedremo ora.

4.9.3 Indirizzi di un Access Point

Questi apparati permettono sia un indirizzamento dinamico che un indirizzo fisso. Se da una parte l'indirizzo dinamico permette una connessione rapida con il computer, sorvolando su alcuni problemi di gestione, dall'altra il mancato controllo sull'assegnazione degli indirizzi introduce inutili rischi che una attenta gestione a indirizzi statici risolve intrinsecamente. Specialmente se si vuole controllare un device da remoto si consiglia un indirizzamento fisso. Di default i device hanno impostato un IP dinamico, come si può controllare alla pagina

$$Access\ Point \rightarrow Setup \rightarrow Network \rightarrow NAP\ Interface \rightarrow Log\ .$$

alla voce: "Use dynamic network configuration". Passare da un indirizzo dinamico a uno fisso è molto semplice, ma sono necessarie informazioni sulla natura della rete, è inoltre un'operazione molto delicata; un errore potrebbe rendere il device non più raggiungibile da remoto. Per passare da una configurazione a indirizzo fisso è sufficiente premere su Yes e compilare il font con i dati necessari:

- Indirizzo IP
- Maschera di sottorete
- IP default gateway
- IP del DNS

ottenendo una configurazione simile a quella in figura 4.16.

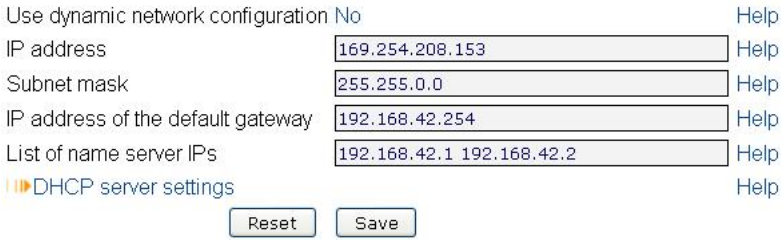

Figura 4.16: Pagina di set up per la configurazione di rete

4.9.4 Comunicazione tra gli Access Point

Una volta deciso in che modo indirizzare gli Access Point e Access Server è necessario avvertirli in che modo cercare gli altri Access Point o Access Server. Se si usa un indirizzo IP statico non è necessaria alcuna operazione aggiuntiva. Se si usa invece un indirizzamento dinamico i dispositivi usano l'interfaccia denominata zeroconfig. In questo caso nel file di configurazione accedibile alla pagina

Access Point → *Setup* → *Applications* → *ObexSender*→ *Edit configuration file*

è necessario modificare l'opzione broadcast nap in **broadcast nap:9**. Nap è una virtualizzazione della presa di rete Ethernet, è quindi caratterizzata da un IP, è necessario usare nap invece di eth0 al quale nap è collegato (linked). Quando si tenta di collegare due device attraverso il cavo di rete, con tale comando si forza i due dispositivi a usare l'interfaccia zeroconfig. Tale interfaccia è un protocollo standard dell'ETF per la configurazione dinamica dei nodi di una rete utilizzante il protocollo IP. Il protocollo non è ancora definitivo, infatti non esiste ancora nessun documento RFC sul protocollo Zeroconf sebbene esistano già diverse implementazioni del protocollo che vengono utilizzate quotidianamente da molti essendo già inclusi in alcuni release di OS open source. L'idea base dello Zeroconf è quella di poter collegare due computer tramite un cavo di rete e senza bisogno di interventi da parte dell'utente: i computers dovrebbero essere in grado di comunicare tra loro. Mentre nelle maggior parte delle applicazioni è necessario settare alcuni parametri manuali, dato che la rete è formata da apparecchi con "in pratica" lo stesso HW e lo stesso SW questo passaggio non è necessario.

4.9.5 Access Point e Access Server Connessi

Una volta realizzata la rete i dispositivi i dispositivi, se posti correttamente su una stessa LAN, si sincronizzano in modo automatico. I messaggi che si mandano sono molto semplici e riguardano la lista dei cellulari serviti. Il periodo di

start-up è particolarmente critico, infatti i device prima iniziano l'operazione di scan e invio, mentre la comunicazione via rete ha una priorità minore. Se infatti si accendono contemporaneamente due device collegati in rete questi tenteranno entrambi di inviare un contenuto. Se invece il cellulare entra nel loro raggio dopo qualche minuto dall'accesione, solo uno tenterà di inviare il contenuto. Interessante la gestione dei timeout. Il device che ha inviato il messaggio associerà a quel numero il proprio valore di timeout (in base al risultalto della transizione). Questa informazione viene passata tramite la rete a tutti i device che man mano che passa il tempo decrementeranno il contatore. Il primo che dopo un'operazione di scan vedrà questo timeout scaduto tenterà una connessione, avvertendo gli altri device. Ecco come variano le operazioni

- SEND: cambia come detto sopra, al device può esser vietato di inviare contenuti ad un particolare cellulare per un tempo pilotato da chi servito quel cellulare.
- REPLY: nessuna modifica, è possibile scegliere a che device inviare un messaggio, è quindi importante non dare nomi uguali ai device che andranno installati nelle vicinanze.
- GOTO: la procedura è locale, la procedura impedisce agli altri access point di inviare i contenuti al cellulare. Solo l'access point che è nel mezzo della struttura label goto tenterà di comunicare con il cellulare. Se per un qualche motivo gli altri access point non ricevessero segnali di sincronizzazione, questo diritto si perderebbe e uno degli altri tenterebbe di comunicare seguendo le regole descritte sopra.

Il firmware attuale non realizza una politica per ottimizzare la scelta dei dispositivi da servire, il primo access point che dopo l'operazione di scan trova un cellulare da poter servire, senza usare alcune informazioni già disponibili come RSSI, o la presenza del cellulare nel raggio durante operazioni di scan precedenti. Si realizza in questo modo un comportamento greedy che cerca di catturare il maggior numero di dispositivi possibile nel modo più leggero possibile. Versioni successive del firmware permetteranno l'implementazione di regole per la gestione.

4.10 Pacchetti WPK

4.10.1 Introduzione

Come detto precedentemente i pacchetti wpk hanno il formato:

name.architecture.wpk

I package sono archivi tar compressi con gzip (*.tar.gz, o *.tgz) e devono contenere un file chiamato wpkg.pif nella root del pacchetto. Tutti gli altri file, se presenti, dovrebbero essere script, esguibili o operazioni di management

4.10.2 Management Packet Information File Format (*.pif)

Il file pif contiene informazioni necessarie per il corretto funzionamento. Si tratta di uno script in Linux che istruirà l'acces point o l'access server su come interpretare in modo corretto il file wpk: per esempio indirizza in modo corretto gli eseguibili, i contenuti, e si occupa di cancellare le cartelle utilizzate per l'installazione del pacchetto.

%wpkg-version: 2

Contiene informazioni sulla versione, 2 è l'unica versione supportata, ed è il valore di fefault.

%wpkg-prepare: [command line[s]]

Uno o più comandi (tutte le righe sono interse essere comandi fino al prossimo tag) da eseguire. I comandi possono contenere parametri, reindirizzamenti e variabili di controllo.

%wpkg-reply: method

Gestisce la funzione di reply, indica dove i pacchetti generati sono inviati. Di defaulto sono inviati a chi ha attivato la funzione stessa. Alcuni possibili valori sono:

- default
- file://path/filename
- scp://remote:file
- objp://bdaddr
- none

%wpkg-format: type

Indica che tipo di risposta viene inviata dalla funzione reply

- ascii (formato default, tutto ciò chiamato con il comando echo sarà inviata)
- tgz (saranno inviati tutti i file contenuti nella cartella).

- vcf (come il formato ascii ma ma in formato V card).
- vmg (come il formato ascii ma ma in formato vMessage).
- vnt (come il formato ascii ma ma in formato vNote).
- vcs (come il formato ascii ma ma in formato vCalendar).
- html (come il formato ascii ma ma in formato HTML).

%wpkg-option: value
Si tratta di parametri addizionali che possono essere utilizzati per evitare di istallare due volte lo stesso pacchetto da una chiavetta USB (una volta inserita e se dopo un reboot se la chiavetta è ancora inserita). Il demon crea un file packet_name.dupe con il un numero seriale e un timestamp. Quando il processo inizierà di nuovo il demone farà abortire l'istallazione duplicata (se si è usato il parametro dupe), o se il time stamp è più vecchio di un'ora (dupe1h), o se ha meno di un giorno (dupe1d). Non si sono esguiti test più approfonditi scegliendo di attendere l'arrivo di SDK per la realizzazione di scenari più complessi.

4.10.3 ObexSender.conf

Si tratta del file di configurazione di ObexSender, tramite un pacchetto wpk è possibile modificare la campagna agendo direttamente sul file di configurazione. Durante l'installazione al file di configurazione viene aggiunta alla fine una sending rule che invia tutti i cellulari tutti i contenuti. Tuttavia modificando in modo adeguato questo file è possibile realizzare una campagna conforme alle nostre esigenze, come se modificassimo realmente il file di configurazione. Si veda il test #24 in appendice per ulteriori dettagli.

I file eseguibili

Per file eseguibili intendo quei file che permettono di realizzare una funzione. Possono essere principalmente di due tipologie:

- Applicazioni eseguibili da shell, cron o rc.local: sono applicazioni che possono migliorare il funzionamento dell'access point o aumentarne il tempo di vita. Per esempio è possibile realizzare una funzione che permetta di cancellare i file di log meno recenti salvando solo alcuni dati per inviarli ad un certo indirizzo e-mail. Infine è possibile realizzare un'interfaccia web personalizzata da fornire al cliente finale. Queste applicazioni sono di default installate in usr/bin

- Applicazioni per ObexSender: si tratta di applicazioni eseguibili tramite ObexSender per l'invio di contenuti ai cellulari. Queste applicazioni verranno trattate nel capitolo dedicato a SDK

4.11 Realizzare un factory reset

Questa procedura permette di riportare il SW del device come se fosse appena uscito dalla fabbrica. Si tratta di una misura drastica, infatti vengono persi tutti i dati salvati compresi:

- I log
- Le modifiche ai file di configurazione
- Le applicazioni installate
- I contenuti caricati

La procedura è di fondamentale importanza perché permette di reimpostare il sistema operativo del device in caso di una sua corruzione. La procedura più semplice è quella attraverso una chiavetta USB.

1. Scompattare il file factory reset scaricato item Salvare i file di reset nella root della chiavetta USB
    - Per Access Point: kernel.lt, root.lt e uImage.lt
    - Per Access Server: kernel.if, root.if e uImage.if
2. Connettere la chiavetta USB alla porta USB del dispositivo
3. Connettere la presa di alimentazione alla corrente per accendere il dispositivo
4. Lasciare la chiavetta USB connessa, i led avranno lo stesso comportamento di un fimware update

4.12 Database dei disposiviti

ObexSender usa un particolare database per il riconoscimento dei dispositivi Bluetooth. Questo database contiene l'informazione BluID (impronta digitale) del

dispositivo Bluetooth così come regole per il riconoscimento del dispositivo. Il riconoscimento del cellulare viene eseguito in base al MAC del dispositivo. Questo paragrafo descrive il formato del database e illustra alcune linee guida su come usarlo. É consultabile attraverso una connessione http all'indirizzo:

Access Point → *Setup* → *Advanced* → *Browse all files*

seguendo questo indirizzo:

/var/lib/obexsender/obexsender.hash

Oppure seguendo andando direttamente a questo indirizzo con una connessione SSH o SFTP.

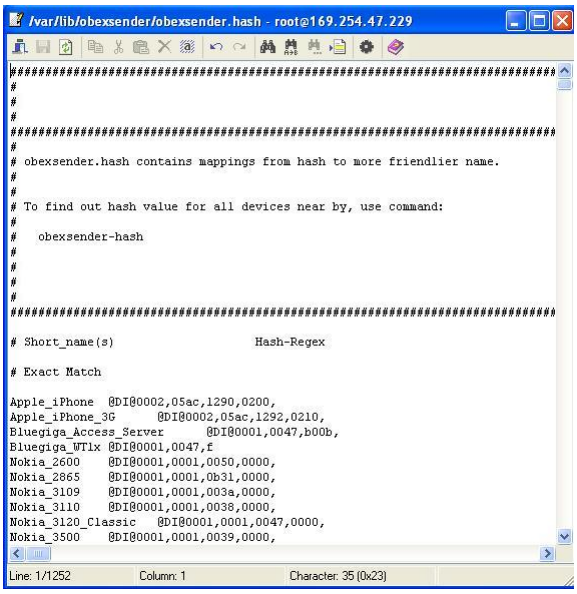

Figura 4.17: Dettaglio del file di hash

Il database è così strutturato: <Device name/group> + <ID>

- <Device name/group>: Questo campo contiene il nome del dispositivo o del gruppo di dispositivi a cui appartiene. Nella figura 4.17 questo campo contiene il produttore e il modello del dispositivo bluetooth. É inoltre possibile creare dei gruppi di celulari che condividano una certa caratteristica; per esempio queste righe inseriscono i cellulari Nokia_N72, Nokia_N73, Nokia_N80, Nokia_N90 nel gruppo Nokia_N_Series e Symbian60.

Nokia_N72, Nokia_N_Series, Symbian60

Nokia_N73, Nokia_N_Series, Symbian60
Nokia_N80, Nokia_N_Series, Symbian60
Nokia_N90, Nokia_N_Series, Symbian60
Nokia_6600, Symbian60

Il nome del gruppo di dispositive è usato nel file di configurazione di Obexsender per decidere quale file mandare a questi dispositivi. Per esempio per mandare un file a tutti i dispositivi sopra tranne il Nokia 6600 si può usare:

Estratto del file Obexsender.conf 4.3 ***Esempio di Sending rule***

```
send {
   match Nokia_N_Series
   file Nokia.jpg
    }
```

- <BluID>: questo campo contiene i dati identificativi del dispositivo. É presente una classificazione dei device in base alla qualità della informazione per il riconoscimento del device

  1. Exact Macth. Informazione sicura, è possibile il riconoscimento del modello di cellulare in ogni momento. Codice: @DI@
  2. Good Guess e Relaxed Guess. Informazioni buone, il riconoscimento è molto probabile. Codici: @GG@ e @BN@
  3. Wild Guess. Non viene garantito il riconoscimento, ma i file dovrebbero venire inviati con successo. Codice @FN@

4.13 Database delle features dei cellulari

Questo database dovrebbe raggruppare i cellulari in base alle loro caratteristiche; dovrebbe permettere di abilitare opzioni avanzate per il match per permettere l'invio ai cellulari non in base alla marca o modello, ma in base alle caratteristiche e funzioni del cellulare stesso.

- camera: solo ai cellulari dotati di fotocamera
- mp3: solo cellulari che leggano il formato mp3

- 3gp: solo cellulari che leggano il formato 3gp
- Html: solo cellulari che supportino la lettura di pagine html
- Symbian 60: solo ai cellulari che supportino il linguaggio Symbian nella versione 60
- Symbian 80: solo ai cellulari che supportino il linguaggio Symbian nella versione 80

Questo file è configurabile dall'utente (si veda il test #44 in appendice), ma per un aggiornamento completo sarebbero necessario molto tempo.

4.14 Software eseguibile sui device

4.14.1 Introduzione

In questo paragrafo viene presentata una divisione del SW presente nell'access point. La divisione cerca di seguire sia lo scopo sia il linguaggio in cui è scritto.

Kernel di Linux e Boot Loader

Il kernel è il cuore di un sistema operativo (nucleo) e fornisce tutte le funzioni essenziali per il sistema, in particolare la gestione della memoria, delle risorse del sistema e delle periferiche, assegnandole di volta in volta ai processi in esecuzione. La controparte del kernel è la shell, ovvero l'interfaccia utente del sistema, la parte più esterna. I programmi chiedono le risorse al kernel attraverso delle chiamate (system call) e non possono accedere direttamente all'hardware. Il kernel si occupa quindi di gestire il tempo processore, le comunicazioni e la memoria distribuendole ai processi in corso a seconda delle priorità (scheduling). Il core è multitasking, ossia permette di eseguire numerosi processi. Si tratta di una versione del kernel molto leggera, e ottimizzata, infatti molte componenti non sono installate (per esempio non è possibile usare il comando "locate", ma solo il grep). Il boot loader è il programma che carica il kernel di un sistema operativo e ne permette l'avvio. Il termine deriva dal fatto che il processo di avvio di un computer viene chiamato bootstrap

Driver per il kernel proprietario

Sono funzioni aggiuntive fornite dal produttore per la gestione dei device. Si tratta di driver non presenti di default nelle release. Sono utilizzati per controllare

le funzioni hardware come i led, la funzione di reset e la gestione del segnale BT. Alcuni esempi sono: led output,bbreset.

Applicazioni Servizi Bluetooth realizzate dal fornitore

Si tratta dei servizi e delle applicazioni realizzate per il funzionamento dell'Access Point e dell'Access Server. Alcuni esempi sono: ObexSender e connector

Applicazioni e Servizi realizzati dal fornitore

Sono quelle applicazioni che permettono di utilizzare le funzionalità dell'Access Server in modo semplice come wpkgd per l'utilizzo dei pacchetti wpk o come setup che gestisce l'interfaccia web presentato precedentemente.

Applicazioni GPLed (BusyBox, OpenSSH, bash)

Essendo una macchina Linux è possibile utilizzare applicazioni di opensource come Busybox , OpenSSH e bash.

Script scritti dall'utente

Come detto, trattandosi di una macchina Linux è possibile realizzare degli script sia per ottimizzare la gestione della macchina, e anche per gestire una campagna di invio con messaggi bluetooth. Per esempio è possibile cancellare i log in caso questo raggiunga dimensioni pericolose per il corretto funzionamento dell'access point stesso.

Applicazioni SDK

Con SDK è possibile realizzare delle applicazioni avanzate utilizzando C come linguaggio di programmazione ad alto livello. A questo potente strumento sarà dedicato il capitolo 6.

Capitolo 5

PROGETTI REALIZZATI

5.1 Introduzione

In questo capitolo viene dato un esempio di ciò che è possibile realizzare senza il software development kit tool. Si tratta di campagne relativamente semplici, ma funzionali e efficaci, realizzate con i comandi presentati fin'ora.

5.2 Polesine Innovazione

La prima campagna realizzata è stata realizzata in data 23 ott. 09 per Polesine Innovazione per La Fiera dell'elettronica e dell'informatica di Rovigo

Caratteristiche del progetto

- Ogni minuto il server realizza una operazione di SCAN per trovare dispositivi cellulari con il Bluetooth accesso e impostati in modalità visibile.
- A ogni cellulare nel raggio di 10-2 0m verrà inviato una richiesta di conferma per la ricezione del messaggio da "Polesine Innovazione"
- In caso di risposta affermativa l'utente riceverà il contenuto promozionale: Coupon.jpg.
- Dopo 5 minuti chiunque abbia ricevuto il primo messaggio riceverà anche il secondo.
- Gli utenti che hanno accettato entrambi i contenuti riceveranno nuovamente la campagna dopo 4 ore.
- Coloro che hanno rifiutato il primo messaggio, non saranno più disturbati

- Coloro che non hanno né accettato né rifiutato, se ancora all'interno del raggio del server riceveranno nuovamente il messaggio ogni mezz'ora finché non decideranno di accettarlo.

Figura 5.1: Coupon

Figura 5.2: Tecnologia Amica

Questa semplice campagna si è ottenuta tramite l'inserimento di queste righe di codice nel file di configurazione /ect/obexsender.conf.

Estratto del file Obexsender.conf 5.1

```
send {
   file Coupon.02.jpg
   goto 300 file2
      }
send {
   label file2
   file Tecnologia.jpg
     }
```

5.3 Quiz

Attraverso il comando retry è possibile creare un quiz da presentare all'utente. L'access point invia una domanda, attraverso una nota o un'immagine. A questo punto l'utente risponde una nota all'Access Point. Se la risposta è giusta l'Access Point invierà una nuova domanda. La possibilità di accedere a diversi tipi di media, testi, immagini, video, audio può essere vincente nell'attirare l'attenzione del pubblico. La successione di domande viene realizzata con una catena di Reply che permette di simulare una serie di domande. Questa realizzazione presenta due pesanti difetti. Innanzi tutto come detto l'analisi del file inviato dalll'utente è molto semplice. Il sistema è care sensitive, quindi la risposta "Zante" è diversa da "zante";e non si nessun tipo di controllo sul flusso delle risposte di ogni singolo utente. In questo modo , non possiamo reagire nel caso l'utente non sappia la risposta, possiamo solo reagire a due eventi:

- L'utente risponde in modo corretto
- L'utente non risponde in modo corretto.

Ma non è possibile esprimere la condizione "l'utente invia per tre volte una riposta errata". Utilizzando SDK è possibile ovviare a questo problema memorizzando tutte le risposte da un singolo cellulare.

5.4 Zaha Hadid

Un altro esempio interessante è la campagna per la mostra di Zaha Hadid, tenutasi a Palazzo della Ragione. Zaha Hadid (Baghdad, 31 ottobre 1950), è un' architetto

e designer irachena naturalizzata britannica. Esponente del decostruttivismo, è stata la prima donna a vincere il Premio Pritzker [22]. La campagna prevedeva l'invio di 5 messaggi due tranches distinte con una pausa di 5 minuti. Nella prima tranche venivano inviati 3 messaggi:

- Un coupon con il luogo e le date mostra.
- Una foto con un opera.
- Un file di testo con i diritti d'autore della foto.

Figura 5.3: Prima tranche della campagna

Nella seconda tranche invece:

- Una foto con un opera diversa dalla precedente.
- Un file di testo con i diritti d'autore della foto

Per creare una campagna più interessante si è cercato di fornire al cliente immagini diverse. Senza Sdk non era possibile inviare file personalizzati al cliente; non era possibile inviare file diversi a chi avesse già ricevuto quegli stessi file. Si è quindi scelto di realizzare sei sottocampagne, tre per il Lunedì, Mercoledì, Venerdì e Domenica, e tre che per il Martedì, Giovedì e Sabato che ruotassero ogni mezzora.

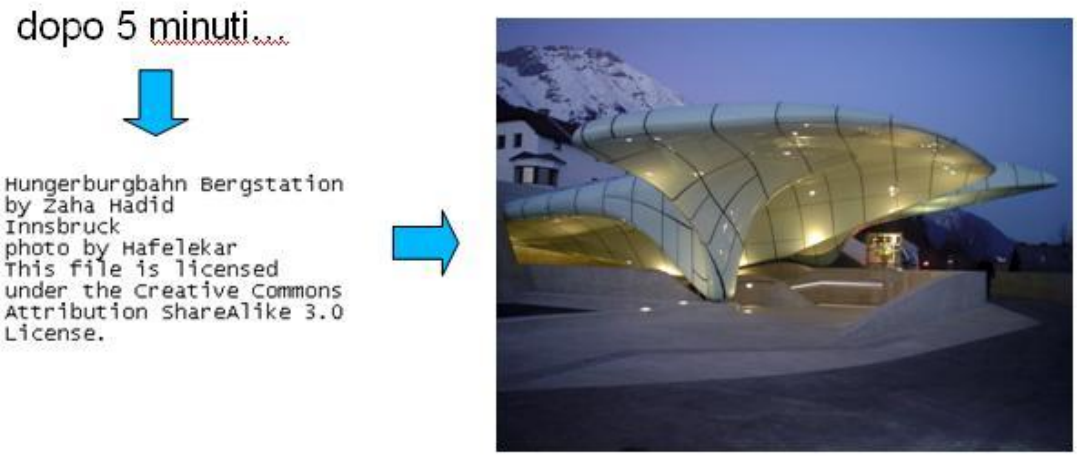

Figura 5.4: Seconda tranche della campagna

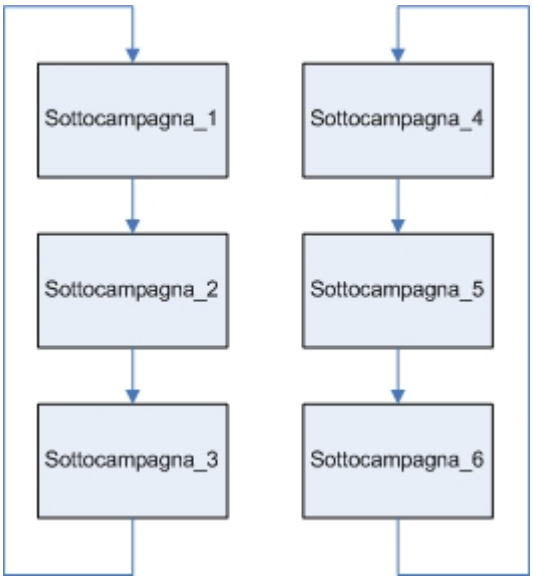

Figura 5.5: Schema di rotazione delle sottocampagne

Estratto del file Obexsender.conf 5.2

```
send {
          time Mon Wed Fri Sun 06:00-06:30 07:30-08:00 9:00-9:30...
file 1_Museo_del_Mediterraneo.txt
file 1_Museo_del_Mediterraneo.jpg
        file Coupon.jpg
goto 300 fileB1
}
send {
          label fileB1
file  2_Museo_del_Mediterraneo.jpg
file 2_Museo_del_Mediterraneo.txt
}
```

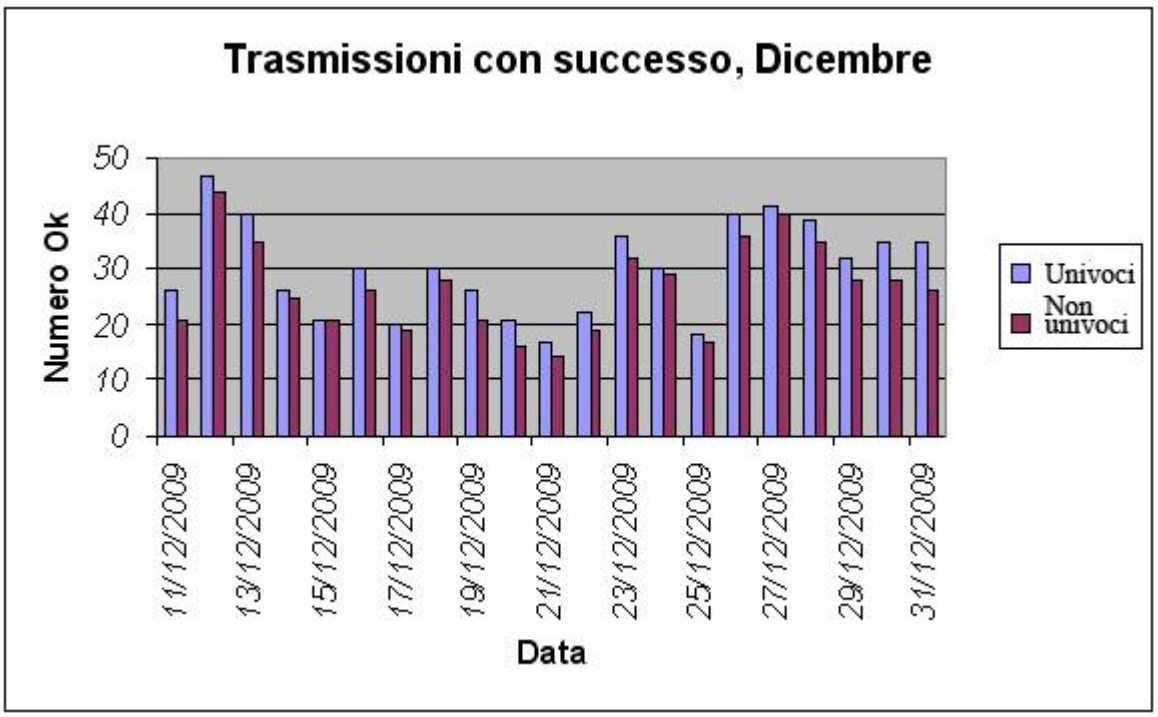

Figura 5.6: Risultati della campagna da 11/12/09 a 31/12/09

Capitolo 6

SDK: SOFTWARE DEVELPMENTO KIT

6.1 Introduzione

Come detto fin'ora senza SDK è possibile creare campagna pubblicitarie, caratterizzate da un grado libertà abbastanza limitato. I limiti che si sono presentati sono stati:

- I contenuti devono essere precaricati, e quindi statici. Si sentiva l'esigenza di avere un controllo maggiore sui contenuti. L'invio di un messaggio personalizzato, per esempio con l'ora o con il nome del dispositivo avrebbe colpito maggiormente il pubblico, aumentando l'aspetto emozionale.

- Successivamente il controllo dei flussi di messaggi messo a disposizione dal comando go to è eccessivamente limitato. Sarebbe preferibile che la campagna sia in grado di modificarsi dinamicamente in base ai risultati della campagna stessa. Per esempio un device si trovasse nella condizione di avere tutti i canali occupati, capendo quindi di aver davanti un gran numero di potenziali clienti, potrebbe scegliere di inviare dei contenuti più leggeri per massimizzare i contatti; oppure servire in maniera diversa un cellulare che avesse già ricevuto il contenuto.

Per questo motivo Ne-t by Telerete Nordest srl ha deciso di munirsi di questo kit di sviluppo.

6.2 Installare SDK

6.2.1 Requisiti Software

- Distribuzione Linux
  SDK è stato testato sulle seguenti piattaforme: Ubuntu 8.10, Ubuntu 8.04, CentOS 5.2, Fedora 10, Fedora 9, Foobar Linux 5, RedHat Enterprise Linux 5.3 and openSUSE 11.1 dalla casa madre. Inoltre si è dimostrato stabile anche su processori a 64 bit.

- Pacchetti aggiuntivi
  Se si sceglie di installare SDK nella cartella di default /usr/local/arm sono necessarie le librerie zib e bzip2 (per Ubuntu il package gettext tramite il comando sudo apt-get install build-essential zlib1g-dev libbz2-dev gettext). Se invece si vuole installarlo in una directory diversa è necessario ricompilare tutto il kit e sono necessari altri pacchetti addizionali (gawk bison flex libtool automake texinfo libncurses-dev). L'operazione è molto pesante e quindi vivamente sconsigliata.

- BGT-sdk-wrap-4.0-090416.iso, immagine del cd del kit di sviluppo

6.2.2 Requisiti Hardware

- Spazio sul disco fisso: 2GB (4GB se è necessario ricompilare SDK).

- Una connessione ethernet LAN è caldamente consigliata (Necessaria per ricompilare SDK)

6.2.3 Procedura

É necessario che chi istalla il kit abbia privilegi root, in dettaglio è necessario poter montare l'immagine del cd e creare directory. Il metodo più veloce è tramite shell dei comandi. Prima di tutto è necessario installare l'immagine del cd attraverso i comandi

Riga di comando 6.1 ***Comandi per fare il mount di una iso***

```
\$ mount /dev/cdrom /mnt/cdrom
\$ (or mount -o loop /path/to/sdk.iso /mnt/cdrom)
\$ cd /mnt/cdrom
```

Se tutto è stato svolto correttamente dovrebbe essere possibile visualizzare il contenuto del cd. Successivamente è possibile procedere con l'installazione vera e propria tramite il comando

Riga di comando 6.2 ***Comando per istallare SDK***

```
\$ sudo sh sdk-install
```

selezionando al cartella in cui installare SDK.

Riga di comando 6.3 ***Installazione di SDK***

```
SDK installation\nline
NOTE! If you change directory entire toolchain must be recompiled.
Recompile process is very complicated and may fail on your system.
Toolchain directory: [/usr/local/arm] ENTER
SDK directory: [/home/user/asdk] ENTER
Install toolchain sources? [y/N] ENTER
```

L'installazione se si mantiene la cartella di default può durare fino a 10 minuti e sposta nella cartella /asdk tutti i file necessari per un corretto funzionamento del kit di sviluppo.

6.3 Cosa è SDK

Si tratta di un toolchain che traduce un file in C o C++ in un pacchetto in formato wpk, quindi leggibile dagli access point e access server. Compilando il codice sorgente attraverso la specifica toolchain vengono creati i pacchetti sia per la versione access server che access point. A differenza di un normale compilatore C SDK permette di sorvolare su alcuni warning dovuti alle differenze di file system tra la macchina in cui scriviamo il codice, e quello in cui andrà eseguito:

in particolar modo gli errori dovuti alla funzione Symlink sono completamente ignorati. Durante la compilazione compaiono a video tutte le informazioni di compilazione, compresi creazione delle cartelle, aggiornamento, errori e warning.

Per quanto riguarda il loro utilizzo non esiste una procedura standard ma dipende dal loro scopo. Possiamo identificare 3 tipi di applicazioni:

- Applicazioni eseguibili come un normale programma, da bash o attraverso cron
- Applicazioni per l'invio di messaggi tramite ObexSender.

Il mio studio si è concentrato sulle seconda tipologia, tuttavia, uno sviluppo futura sarà realizzare una applicazione del primo tipo che sia in grado di modificare alcune regole implementate in ObexSender che non permettono un pieno controllo del processo, ulteriori dettagli saranno forniti nel paragrafo limitazioni di questo capitolo.

Prima di analizzare gli esempi forniti dal supporto clienti vediamo come creare un nuovo progetto.

6.4 Creare un nuovo progetto

Per facilitare il lavoro dello sviluppatore, come detto, il kit realizza in modo autonomo una versione dell'applicazione per l'access point e una per l'access server. Il comando da usare è il seguente:

Riga di comando 6.4 ***Comando per crare un nuovo progetto sdk***

```
\$ sudo make appwiz APP=<pathname del nuovo progetto>.
```

per esempio

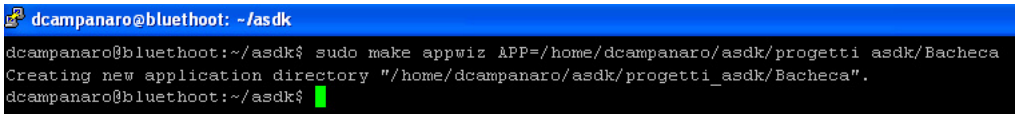

Figura 6.1: Compilazione di un progetto con SDK

Con questo comando vengono creati due file:

- progetto.c: un file sorgente in C editabile, di default viene creata l'applicazione Hello,world!.

- Makefile: ha un' importanza fondamentale, infatti contiene le informazioni su come usare il codice C scritto, dove memorizzarlo nel device e quindi a che servizi deve appoggiarsi per funzionare in modo corretto.

Per compilare un progetto è sufficiente spostarsi nella cartella del progetto e immettere il comando make

Riga di comando 6.5 ***Comando per creare un'applicazione***

```
\$ sudo make
```

6.4.1 Risultato della compilazione

Il risultato della compilazione, come detto, è un file wpk. É possibile aprire questo file cambianone l'estensione in tgz. Infatti dopo alcuni test è risultato che un file wpk non è altro che un file tar.tgz. L'estensione tgz, formata da soli tre caratteri come nello standard DOS, serve ad identificare un insieme di file archiviati con tar in seguito compressi con gzip. Solitamente, nell'ambito UNIX, l'estensione si presenta nella sua forma completa .tar.gz. è molto comune nel mondo UNIX e GNU/Linux per distribuire pacchetti composti da più file, solitamente, come nel nostro caso, programmi in formato sorgente.

- Per quanto riguarda le applicazioni da shell

```
|-applicazione.(HW).tar
        |-control.tar.gz
                |-control.tar
                                |-control
        |-data.tar.bz2
                |-data.tar
                                |-\usr\bin\applicazione
```

- Per la realizzazione di una applicazione ObexSender.

```
|-applicazione.tar
        |-applicazione (HW).deb
                |-control.tar.gz
                                |-control.tar
                                                |-control
```

|-data.tar.bz2
|-data.tar
|-\usr\local\obexsender\bin\applicazione

Il file control contiene le informazioni della versione del software, del hardware, dei package utilizzati e delle dipendenze; mentre il file senza estensione è l'applicazione compilata.

- Un'altra struttura infine è quella per aggiornare o modificare una campagna USB utilizzando un unico file:

|-applicazione.tar
|-contenuto1
|-contenuto2
|-wpkg
|-obexsender.conf

Il contenuto può essere un file in qualsiasi formato, wpk è in formato Package Interchange Format che comanda lo spostamento dei contenuti nella cartella /usr/local/obexsender/files/ e la sovrascrittura del file di configurazione.

6.5 Esempi forniti dal costruttore

6.5.1 Hello,world!

Il software development kit viene fornito con alcuni esempi di applicazioni molto semplici, ma che permettono di prendere confidenza con i comandi messi a disposizione.

Il primo come detto è un'applicazione di tipo Hello,world. Il suo funzionamento è molto semplice, una volta caricato il pacchetto wpk è necessario aprire una shell nel device (il sistema operativo è una versione lite di Linux) e immettere il comando:

Riga di comando 6.6 ***Invocazione di un'applicazione da shell***

```
\$ usr/bin/helloworld.
```

In blocco di codice 1.1 viene presentato il file sorgente per l'applicazione hello,world!

Blocco di codice 6.1 ***File sorgente per l'applicazione helloworld***

```
//helloworld.c
#include <stdio.h>

int main(void)
{
printf("Hello world!\n");
return 0;
}
```

Il codice come si vede è il più semplice possibile, una stampa a video di una stringa, il makefile invece è più interessante:

Blocco di codice 6.2 ***makefile per l'applicazione helloworld***

```
# Makefile
include ../../Rules.mak
DESC = example: Hello world!
_$(HW)/helloworld: helloworld.c
$(do_link)
# wpk
mkdir -p tmp/usr/bin
cp $@ tmp/usr/bin/
$(do_wpk) $@

# eof
```

Anche il makefile è relativamente semplice, come si può osservare dal blocco di codice 1.2 include prima di tutto Rules.mak, un file che gestisce la doppia compilazione per la versione Access Server e Access Point nel quale è possibile gestire i vari flag e le opzioni per la compilazione e i linker. Attraverso il comando \ $ (do_link) viene compilata l'applicazione. Invece con le righe successive si istruisce dove l'applicazione andrà ad installarsi una volta che il pacchetto wpk verrà trasferito nel device.

6.5.2 www

Questo esempio permette di cambiare l'interfaccia grafia dell'accesso WWW, permettendo una personalizzazione del prodotto. In questo metodo sarà possibile personalizzare le interfacce di accesso ai servizi, e quindi permettere una migliore gestione del progetto Bluetooth anche ad personale con bassa preparazione tecnologica. Non viene usato nessun codice in C, ma attraverso il makefile si indica dove i file debbano essere copiati all'interno del device.

Blocco di codice 6.3 ***makefile per la creazione di pagine con SDK***

```
# Makefile
HW = noarch
include ../../Rules.mak
DESC = demonstration of WWW server capabilities.
www: Makefile index.html httpd.conf test.cgi
# wpk
mkdir -p tmp/var/www/html/cgi-bin
cp index.html tmp/var/www/html/example.html
cp httpd.conf tmp/var/www/html/cgi-bin
cp test.cgi tmp/var/www/html/cgi-bin
$(do_wpk) $@
touch $@
clean:
rm -f www *.wpk
# eof
```

I file index.html, httpd.conf , test.cgi vengono copiati nella cartella: var/www/html del device. Dopo l'installazione del pacchetto wpk accedendo all'indirizzo http://wrap-ip-address/example.html sarà possibile controllare il risultato.

6.5.3 Led test

Questa applicazione permette il controllo dei led attraverso la shell dei comandi. Il codice presentato è per la versione ACCESS SERVER 3393, è possibile controllare solamente i led di status che sono nominati: B, C, D, E. La lettera maiuscola sta a indicare il led acceso, mentre a una lettera minuscola corrisponde un led minuscolo.

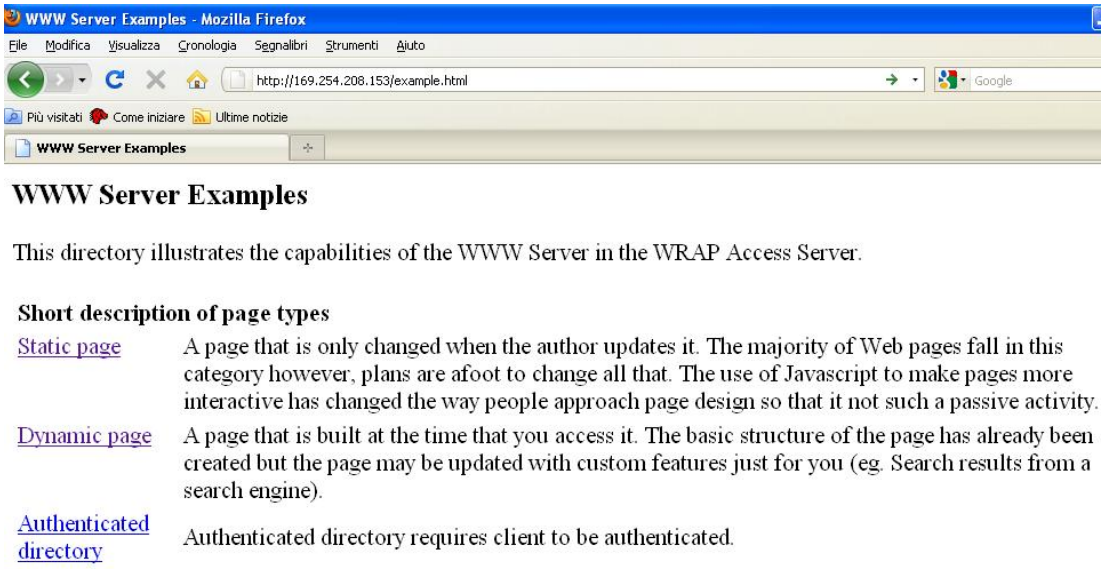

Figura 6.2: Esempio di personalizzazione di una pagina html

Blocco di codice 6.4 ***File sorgente per l'applicazione ledtest***

```
//ledtest.c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int main(void)
{
  char kitt[8][5]={
    "Bcde",
    "bCde",
    "bcDe",
    "bcdE",
    "bcdE",
    "bcDe",
    "bCde",
    "Bcde"
  };
  char buf[2];
```

```
  int i, led;
  led=open("/dev/led",O_RDWR);
  for (;;) {
    for (i=0; i<8; i++) {
      write(led,kitt[i],4);
      usleep(100*1000);
    }
    buf[0]=0;
    buf[1]=0;
    i=read(led,buf,2);
    printf("read=%d, \"%c%c\"\n",i,buf[0],buf[1]);
    if (i<2) lseek(led,0,SEEK_SET);
  }
  close(led);
  return 0;
}
```

Il vettore kitt contiene le istruzioni per l'accensione e lo spegnimento dei led; il flusso led, reinterpretato dal toolchain, comanda il cambio di stato dei led ogni secondo. Visto che questa applicazione viene lanciata dalla shell, il makefile è molto simile:

Blocco di codice 6.5 ***Makefile per l'applicazione ledtest***

```
# Makefile
include ../../Rules.mak
DESC = example: LED control.
_$(HW)/ledtest: ledtest.c
$(do_link)
# wpk
mkdir -p tmp/usr/bin
cp $@ tmp/usr/bin/
$(do_wpk) $@
# eof
```

L'applicazione viene installata nella cartella temp/usr/bin ed è utilizzabile con il comando $ usr/bin/ledtest. I led a questo punto inizieranno a lampeggiare come descritto dal vettore kitt

6.5.4 Lottery

Questa applicazione permette di simulare una lotteria tra coloro che sono presenti nel raggio d'azione di ObexSender, ai quali arriverà un messaggio di vittoria o un messaggio di sconfitta. Per attivare questa applicazione è sufficiente, dopo aver caricato il pacchetto wpk in modo corretto, è necessario creare una sending rule come mostrato nell'estratto 6.1.

Estratto del file Obexsender.conf 6.1 ***Comando per eseguire l'applicazione lottery***

```
    send {
exec /usr/local/obexsender/bin/lottery
}
```

É senza ombra di dubbio l'esempio più utile, infatti con questa applicazione è possibile inviare dei contenuti creati dinamicamente durante l'esecuzione del programma stesso, liberandoci del pesante vincolo di precaricare i contenuti. Questi inoltre possono essere personalizzati come mostreremo ora.

Blocco di codice 6.6 ***File Sorgente per l'applicazione lottey***

```
#lottery.c

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

int main(int argc, char *argv[])
{
FILE *f;
char s[1024], bdaddr[32], name[256], hash[1024];
int verbose, win;

if (argc<3) return 1;
if (!(f=fopen(argv[2],"rt"))) return 2;

/* Read config file */
```

```
while (fgets(s,sizeof(s),f)) {
if (*s && s[strlen(s)-1]=='\n') s[strlen(s)-1]=0;
if (!strncmp(s,"bdaddr ",7)) strcpy(bdaddr,s+7);
if (!strncmp(s,"name ",5)) strcpy(name,s+5);
if (!strncmp(s,"hash ",5)) strcpy(hash,s+5);
if (!strncmp(s,"verbose ",8)) verbose=atoi(s+8);
}
fclose(f);
/* Make lottery! */
srand(time(NULL));
win=rand() & 1; /* 50% chance to win */

/* Write fakename to reply config file */
if (!(f=fopen(argv[2],"wt"))) return 3;
if (win) fprintf(f,"file you_won.txt\n");
else fprintf(f,"file you_lost.txt\n");
fclose(f);

sprintf(s,"%s.reply",argv[2]);
if (!(f=fopen(s,"wt"))) return 4;
if (win) fprintf(f,"You won, %s!\n",name);
else fprintf(f,"Sorry, %s, you lost.\n",name);
fclose(f);

/* Return ok */
return 0;
}
```

La prima parte del codice inizializza le variabili necessarie al corretto funzionamento:

- File*f: è il flusso dove vengono letti gli argomenti di argv[2], e salvati i contenuti da inviare
- char s[] conterrà tutte le istruzioni necessarie per soddisfare la richiesta di invio del contenuto.
- char bdadrr[] è l'indirizzo macchina del cellulare a cui è legata l'esecuzione dell'applicazione stessa

- char name[] è invece il nome friendly del dispositivo Bluetooth
- char hash [] infine contiene le informazioni di hash per il riconoscimento del cellulare.

Successivamente viene calcolata la variabile win; non viene svolta una vera e propria lotteria, ma in base a questa variabile aleatoria si decretano vincitori e perdenti. Infine attraverso la scrittura su un file viene creato il contenuto; nel nostro caso un semplice .txt che utilizza all'interno del testo il nome friendly del dispositivo Bluetooth.

6.6 Come funziona un applicazione SDK

Come è possibile vedere dai blocchi di codice 6.6 e nel dettaglio 6.7 le applicazioni hanno due tipi di input

- Una stinga alfanumerica contenente le seguenti informazioni:
  1. Indirizzo MAC del cellulare
  2. Il nome amichevole del dispositivo
  3. Il risultato della funzione di hash
  4. Il verbosity log level.
- L'indirizzo del file di configurazione che andremo a modificare con la formattazione /tmp/obexsender. *ip+porta (socket)*.reply; per esempio /tmp/obexsender10101.3.2, con con ip uguale a 10101 e porta indicata da 3.2. L'indirizzo IP è relativo all'antenna che cerca di stabilire una connessione, per quanto riguarda la porta il primo numero è legato indica l'ennesimo cellulare che si tenta di contattare durante la transizione e l'ultimo numero indica sta l'access point sta cercando di inviare l'emmesimo. Per esempio 10101.3.2 indica che l'access point con Ip 10101 sta cercando di inviare al quarto cellulare catturato durante l'ultima operazione di scan il secondo file della sending rule.

Dal nostro sorgente in C non controlliamo l'invio del file, e nemmeno siamo interfacciati ai comandi di invio del contenuto, bensì siamo ad un livello logico più elevato dove è solamente possibile elaborare il contenuto da inviare attraverso la scrittura di un file di configurazione. Se volessimo inviare l'immagine_A al cellulare_1, in questo file di configurazione andrebbe scritto qualcosa traducibile

con: Invia al cellualre_1 l'immagine_A usando il servizio 10102.reply. Una volta che questo file è stato scritto il programma si può concludere.

Blocco di codice 6.7 ***Dettaglio per scrittura del file di configurazione***

```
fprintf(f,"File_da_inviare.jpg \n");
sprintf(s,"%s.reply",argv[2]);
symlink ("/usr/local/obexsender/files/ File_da_inviare.jpg",s);
```

Successivamente ObexSender andrà a interpretare questo file di configurazione per inviare il contenuto al cellulare. Proprio per questo motivo, come confermato dal supporto clienti non esiste alcun modo automatico per verificare se il contenuto sia stato inviato in modo corretto o meno. Per ovviare a questo problema è sufficiente interfacciarsi ai log della macchina stessa, soluzione che può rilevarsi particolarmente onerosa in caso i log siano di grandi dimensioni, o salvati in un disco fisso o un server esterno. Per lo stesso motivo risulta chiaro che con un singolo file sorgente non è possibile pilotare l'invio di due contenuti. A ciò si può ovviare creando una sending rule che lanci l'esecuzioni di due applicazioni. L'applicazione che noi creiamo viene chiamata solamente al momento del tentativo di connessione; il flusso logico è mostrato in fugra 6.3

A meno che l'applicazione non sia particolarmente pesante il file di configurazione viene già creato ancor prima che il cellulari accetti o rifiuti il contenuto. Le informazioni disponibili all'interno del codice sono molto poche:

- Nome amichevole BT
- Codice di hash, da cui è possibile risalire al modello del cellulare
- Indirizzo macchina del cellulare.

Tutte le altre informazioni vanno reperite attraverso l'analisi dei log interni alla macchina, in modo tutt'altro che comodo, introducendo overhead. Da ciò risulta chiaro che questo kit di sviluppo, per quanto abbia un funzionamento corretto, non è ancora ottimizzato, e non è in grado di rispondere, almeno per ora ad alcune funzioni avanzate. Il fornitore attualmente sta lavorando a pacchetti di codice da affiancare a SDK per semplificarne l'utilizzo

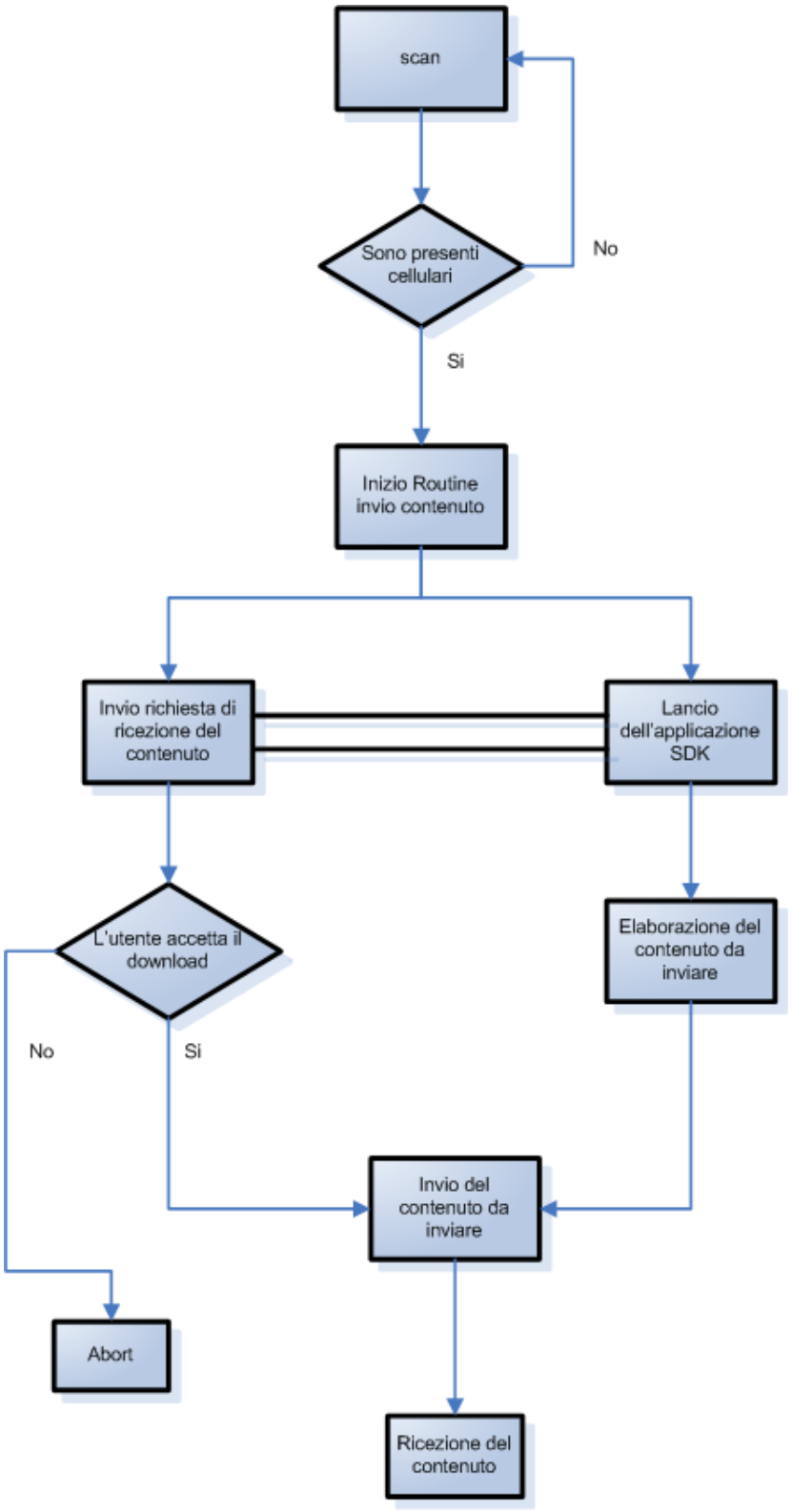

Figura 6.3: Funzionamento generale di una applicazione SDK per l'invio di contenuti

6.6.1 Limitazioni

Con SDK è possibile realizzare applicazioni avanzate, ma ci sono alcuni limiti. Abbiamo un controllo ad alto livello sulle direttive da utilizzare; infatti non possiamo comunicare direttamente con le API per l'invio di dati tramite il canale Bluetooth, ma solamente interfacciarsi tramite un file di configurazione ad ObexSender, il servizio per l'invio di messaggi. La struttura di questo file di configurazione è molto rigida e permette solamente di inviare un singolo file. Per inviare messaggi in modo sequenziale, durante una singola transizione, è necessario realizzare diverse applicazioni SDK, e lanciarne l'esecuzione durante una singola sending rule

Estratto del file Obexsender.conf 6.2 ***Sending rule, applicazioni multipli***

```
    send {
exec /usr/local/obexsender/bin/Applicazione_1
exec /usr/local/obexsender/bin/Applicazione_2
exec /usr/local/obexsender/bin/Applicazione_3
}
```

Questa soluzione ci permette di inviare un qualsiasi numero di contenuti in modo sequenziale, tuttavia, per ogni contenuto è necessario creare una diverso programma, come vedremo nella prima applicazione della bacheca digitale.

Un'altro pesante limite è l'impossibilità di conoscere se il messaggio è stato inviato correttamente durante l'esecuzione dell'applicazione stessa, infatti, finchè applicazione non termina, ObexSender non interpreta il file di configurazione. Inoltre, la politica di logging non permette di reperire in modo chiaro e veloce quale utente abbia ricevuto quale file; infatti viene registrato quale porta abbia servito quale cellulare.

Estatto di log 6.1 ***Applicazioni SDK***

```
    Sent /tmp/obexsender.10101.1.0.reply to 00:0f:de:37:69:ba: FAIL/4
Sent /tmp/obexsender.10101.0.0.reply to 00:22:66:da:31:6a: MORE/0
Sent /tmp/obexsender.10101.0.1.reply to 00:22:66:da:31:6a: MORE/0
Sent /tmp/obexsender.10101.0.2.reply to 00:22:66:da:31:6a: OK/0
```

Per ovviare a questo problema è stato realizzato un sistema che aggiungesse righe al file di log riportando la tupla *mac, porta, file*, in questo modo, ogni

successiva applicazione SDK è in grado di ricostruire quali immagini il cellulare abbia già ricevuto.

Estatto di log 6.2 ***Righe aggiunte***

»»Sending file net-Blue.jpg to 00:22:fd:d4:d6:f1 port /tmp/obexsender.10101.0.1
»»Sending file Telerete.jpg to 00:22:fd:d4:d6:f1 port /tmp/obexsender.10101.0.2
»»Sending file Coupon.jpg to 00:22:66:da:31:6a port /tmp/obexsender.10101.0.2

Questo ostacolo è stato aggirato, inserendo nel file di log di ObexSender alcune righe che permettono di costruire la tupla: cellulare-file-porta-risultato dell'invio. Dal punto di vista logico ciò che viene realizzato è un join, usando la porta utilizzata per l'invio e il mac del cellulare per unire il nome del file con il risultato dell'invio del file stesso.

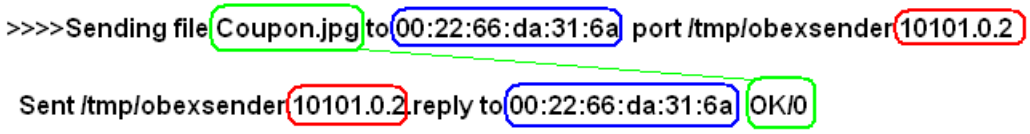

Figura 6.4: Iterazione tra i due log

6.7 Una prima applicazione

Una delle richieste più impellenti era le realizzazione di una campagna che potesse inviare dei contenuti personalizzati, un contenuto riconoscibile e univoco, preferibilmente legato al momento temporale dell'invio e al destinatario del messaggio stesso. Perciò si è deciso di realizzare una applicazione, Info_cellulare.c che inviasse un singolo file di teso con i seguenti campi:

- Friendly name del cellulare
- Indirizzo macchina del cellulare
- Modello del cellulare
- Ora di invio del messaggio

Le prime due informazioni sono già disponibili in chiaro, per reperire il modello del cellulare è sufficiente analizzare il risultato della funzione hash. Il risultato di questa fuzione ha una struttura standard.

Modello_del_Cellulare,Funzioni_Associate,
@BB@Probabilità_di_riconoscimento_del_device
@BD@indirizzo_macchina@FN@BT_friendly_name@.

Identificando li carattere , come fine del campo modello del cellulare è possibile identificare in modo univoco questa informazione. Per recuperare l'ora di invio è sufficiente la funzione: newtime=localtime(&t). Esempio del testo inviato dall'applicazione Info_cellulare:

Info.txt
Friendly name: Sbrillo
Indirizzo Macchina: 00:22:66:da:31:66
Modello: Nokia: Nokia_N73
Ora invio: Wed Jan 13 18:15:52 2010

Naturalmente non andremo a inviare al cliente finale questo messaggio, ma è utile per capire quali informazione siano reperibili all'interno di una applicazione SDK. Mentre il nome amichevole del cellulare serve per aumentare l'aspetto emotivo della campagna pubblicitaria le altre informazioni ci permettono costruire applicazioni più avanzate, è lecito supporre che il MAC di un cellulare, sia un codice identificativo univoco. Identificare univocamente un dispositivo ci permette di capire molto dei comportamenti del suo cliente, tra le altre cose possiamo risalire ai file che ha già ricevuto precedentemente, e al suo comportamento nei confronti della ricezione dei messaggi.

6.8 Utilizzo di uno script

Dalla analisi delle applicazioni di SDK e dalla loro interazione con ObexSender sono nati alcuni test che hanno portato alla stesura di una procedura per utilizzare gli script per la realizzazione delle campagne.
Come detto in precedenza il programma viene invocato con due parametri di input: passati attraverso argv[ ]:

- L'indirizzo mac del cellulare a cui inviare il messaggio

- Il file dove salvare i dati relativi all'invio che indica il servizio a cui andiamo ad appoggiarsi

Esattamente come un applicazione ObexSender è possibile utilizzare queste informazioni per la realizzazione di una campagna di proximity marketing.

Ovviamente le possibilità si programmazione ottenibili con uno script sono inferiori a quelle ottenibili utilizzando SDK, ma per valutare tutte le funzioni di questi device sono stati condotti alcuni test. Tali programmi sono costituiti da semplici file di testo, che a loro volta contengono una sequenza di istruzioni (uno script appunto), con la possibilità di utilizzare condizioni (if e then), cicli (for) e altri costrutti tipici di un linguaggio di programmazione. Per certi versi uno script di shell è l'equivalente nel mondo Unix di un file batch del DOS (più correttamente è vero il viceversa, visto che la programmazione di batch è molto più limitata rispetto a quella di script). Queste istruzioni vengono in seguito interpretate ed eseguite dalla shell di sistema, per questo il formato di questi script dipende fortemente dal tipo di shell utilizzata. Il linguaggio di script è quindi un linguaggio interpretato, in contrapposizione con i linguaggi come il C che sono compilati.

6.8.1 Gli script

Gli script, come detto, sono file di testo che contengono delle istruzioni, per convenzione la prima riga deve essere sempre #!/bin/sh (senza ulteriori commenti). Il carattere usato per i commenti è proprio il simbolo # (tutto ciò che segue il cancelletto è ignorato dalla shell). L'utilizzo più semplice degli script di shell è come file macro per lanciare più comandi in successione (in questo caso lo script non è altro che un elenco di comandi). L'utilizzo più professionale è la realizzazione di veri e propri programmi che utilizzano parametri passati da linea di comando e variabili interne

Variabili e parametri

Le variabili sono delle stringhe alfanumeriche e il loro valore viene identificato da $nome_variabile. É importante notare che per la shell la variabile VAR1 è diversa da var1 (ricordo che Unix e la maggior parte dei programmi distinguono tra maiuscolo e minuscolo). Per inizializzare una variabile si usa l'espressione var=stringa (senza alcun spazio prima e dopo il segno d'uguale), dove stringa deve essere racchiusa tra virgolette (o apostrofi) se contiene degli spazi (i caratteri speciali vanno invece preceduti da \). Oltre alle variabili interne, in uno

script di shell è possibile accedere alle variabili d'ambiente (le stesse che vengono visualizzate dal comando env). Di solito, le variabili d'ambiente sono indicate con caratteri maiuscoli, come PATH (contiene il percorso di ricerca dei file eseguibili) e PS1 (contiene la stringa del prompt di primo livello). Alcune variabili vengono direttamente gestite dalla shell, come ad esempio UID (ID dell'utente), OSTYPE (tipo del sistema operativo), HOSTYPE (tipo di macchina) e molte altre. In una macchina linux normale è sufficiente utilizzare il comando man bash per visualizzare tutti i comandi possibili. Tuttavia la shell linux all'interno dei device in analisi è una versione più leggera con meno comandi, ottimizzata dal punto di vista di memoria occupata nel disco. I parametri dello script possono essere letti mediante i simboli $n (con n intero positivo), in particolare $0 contiene il nome dello script, $1 contiene il primo parametro, $# contiene il numero di parametri e $* contiene tutta la riga dei parametri. Proprio in questo modo avviene il passaggio dei parametri da ObexSender allo script che viene invocato nel seguente modo:
<pathname><Mac del cellulare da servire><servizio da utilizzare per inviare il messaggio>

Quoting

Con il termine di quoting si indica l'utilizzo di apici e virgolette per racchiudere stringhe di caratteri. Ad esempio per visualizzare una frase si possono usare indistintamente i due comandi echo frase oppure echo 'frase' (di solito si utilizzano le virgolette se la frase contiene degli apostrofi o si utilizza l'apostrofo se la frase contiene le doppie virgolette). L'utilizzo dell'accento grave (carattere ‘) è molto importante e determina un risultato totalmente differente: tutto ciò che è racchiuso dalla coppia di caratteri viene interpretato come un comando. In questo modo è possibile assegnare ad una variabile il risultato di un comando. Per inserire caratteri speciali in stringhe o espressioni bisogna utilizzare il carattere di escape \, usato anche per andare a capo su una nuova riga senza interrompere il comando.

Reindirizzamento

Lavorando in ambiente di shell è importante aver ben chiaro il concetto fondamentale di standard input (stdin), standard output (stdout) e standard error (stderr). Ad ogni comando che viene eseguito vengono aperti questi tre flussi e la shell consente un facile reindirizzamento degli stessi verso altri comandi e verso

file. Nella bash i tre flussi vengono numerati rispettivamente con 0, 1 e 2. Per reindirizzare il risultato del comando com verso il file testo, è sufficiente utilizzare l'espressione com > e testo (equivalente a com 1 > testo). Per reindirizzare lo standard error è invece necessario l'espressione com 2 > errori. Per utilizzare sia il flusso 1 che il flusso 2 si utilizza & >. É importante notare che il reindirizzamento su un file già esistente ha come risultato la sovrascrittura del file stesso (a meno che la variabile noclobber sia settata); per aggiungere in coda ad un file si utilizza > >.

Comandi esterni

In ambiente di shell è possibile richiamare ed utilizzare comandi di sistema (o altri script) o direttamente o mediante il quoting (con il carattere ‘) visto in precedenza. Quando un comando chiamato direttamente termina la sua esecuzione restituisce il controllo allo script; per sapere se il comando è andato a buon fine è disponibile una variabile particolare (indicata con ?) che contiene il codice di uscita dell'ultimo comando. Il test [$? = 0] verifica che non vi siano stati errori (per convenzione i comandi restituiscono un valore diverso da 0 quando si è riscontrato un qualche problema). All'interno di uno script, per fornire condizioni d'uscita diverse da 0 (ad esempio per segnalare un errore), è possibile utilizzare il comando exit num, che sospende l'esecuzione dello script e restituisce il valore num (nella variabile ?).
I comandi richiamati mediante quoting, vengono impiegati per ottenere informazioni runtime. Ad esempio, per visualizzare la versione del kernel è possibile utilizzare il seguente comando: echo bin/uname -a. Per consentire la comunicazione tra script diversi (o tra script e comandi) è possibile utilizzare i comandi trap (per accettare un segnale) e kill (per inviare un segnale). Specificando in uno script l'istruzione trap <comando, segnale>, appena che arriva il segnale specificato, viene eseguito il comando corrispondente.

Strutture condizionali

Per realizzare condizioni all'interno di script di shell si utilizza il costrutto if then else fi. Tali strutture si possono nidificare una nell'altra ed esiste anche una sintassi speciale (elif), per evitare di generare troppe coppie if fi. Per uscire direttamente dal costrutto si utilizza il break. L'argomento di if è un test ossia un'espressione che restituisca un valore vero o falso; nella bash la funzione per realizzare il test è già inclusa e si utilizzano le parentesi quadre per racchiudere

l'espressione del test (altre shell utilizzano il comando esterno test). Esistono numerosi tipi di test, negli esempi vedremo alcuni esempi di condizioni su file e su variabili. è importante inserire uno spazio prima e uno dopo alla parentesi quadra, per consentire alla shell di riconoscere i singoli comandi (si ricorda che in linea di comando il separatore di campo è lo spazio).

Test su file, stringhe e numeri

I test sui file disponibili per la programmazione di script sono:

- -x (vero solo se il file esiste),
- -s (vero solo se il file esiste e non è vuoto),
- -d (vero solo se il file è una directory),
- -f (vero solo se il file è realmente un file),
- -r (vero solo se si hanno i diritti in lettura al file),
- -w (vero solo se si hanno i diritti in scrittura al file).

Un discorso a parte vale per la gestione delle variabili, poichè la shell non gestisce direttamente le variabili numeriche (ogni variabile è semplicemente una stringa). Per contro sono disponibili test specifici per variabili numeriche. è possibile valutare l'uguaglianza di una variabile numerica con un certo valore, come [$numero -eq 0] che restituisce vero se la variabile numero è diversa da 0. In alternativa a -eq è possibile usare il segno di uguale (ma sempre preceduto e seguito da uno spazio). D'altronde questo tipo di test restituirebbe vero anche se venisse condotto a livello di stringa (ossia mediante [$numero -eq 0]). I test specifici per le variabili numeriche sono:

- -gt (vero se maggiore),
- -lt (vero se minore),
- -ge (vero se maggiore o uguale),
- -le (vero se minore o uguale) ,
- -eq (vero se uguale).

Il problema delle variabili numeriche viene a galla tentando di modificare il valore della variabile stessa. Supponiamo per esempio che count valga 1 e si desideri incrementare il valore di 2. Purtroppo dopo l'assegnazione count=\$count+2 il nuovo valore di count sarà 1+2 (perchè la shell considera tutto ciò che è alla destra di una assegnazione come un'unica stringa). per effettuare operazioni matematiche sulle variabili è necessario utilizzare il comando esterno expr, la sintassi corretta diventa quindi:
count=‘expr \$count + 2‘ # è importante rispettare gli spazi prima e dopo il segno +. Anche in questo caso gli spazi hanno importanza: vanno messi per separare gli argomenti di expr, mentre non devono essere usati per l'assegnazione di count. Volendo utilizzare expr per operazioni di moltiplicazione bisogna utilizzare * (questo perchè l'asterisco ha un particolare significato per la shell). I test specifici per le variabili stringa sono invece:

- l'uguaglianza (=),
- la disuguaglianza (!=) e
- la verifica sulla lunghezza della stringa (-z per verificare la lunghezza nulla e -n per verificare che la stringa sia non vuota).
- per considerare anche gli spazi all'interno delle stringhe usare [str1 = \$str2].

6.8.2 Esempio di script

Gli script sono stati utilizzati per soddisfare obbiettivo imposto del settore commerciale. Per aumentare l'aspetto emotivo della campagna è stato richiesto di poter inviare un immagine personalizzata con il nome amichevole del cellulare; l'immagine di base sarebbe dovuta essere il flyer della discoteca da sponsorizzare. Come detto in precedenza, però, l'input di uno script è leggermente diverso dall'input di un programma realizzato con SDK, viene passato solamente il mac del cellulare e la porta con cui tale cellulare viene servito. É stato quindi necessario risalire al nome amichevole del cellulare in modo indiretto, questo problema è stato risolto analizzando le ultime righe dei log.

Script 6.1

```
A=‘tail -n 100 /usr/local/obexsender/obexsender.log | grep Hash
| grep "${BD}"‘
FN=‘expr match "${A}"  '.*@"\(.*\)"@.*'‘
```

Con queste righe analizziamo le ultime 100 righe del log cercando una stringa contenente Hash e il nome amichevole del cellulare, una volta trovata viene effettuata una operazione di parsing per catturare il nome amichevole del cellulare utilizzando una espressione regolare.

Estatto di log 6.3

```
Hash: Nokia_N73,S60,@DI@,,,,""@BB@c0018651cf90cd76@BN@57e532cd33c2
@BD@00:22:66:da:31:6a@BP@11668025@FN@"Sbrillo"@
```

Nel dettaglio la riga scelta per reperire tale informazione ha una struttura standard e il nome amichevole del cellulare è sempre posto alla fine, tra doppi apici e il simbolo @. L'espressione regolare *'.*@.*@.*'* cattura alla fine della riga del file di log la stringa posta tra tali simboli, mentre *${A}* permette di salvarla in FN.

Le espressioni regolari vengono definite in informatica teorica come delle espressioni che servono a descrivere insiemi di stringhe. L'operazione principale che utilizza una espressione regolare è il matching, cioè la verifica che una stringa appartiene all'insieme descritto dall'espressione regolare. In realtà il Linux (come pure molti altri programmi) utilizzano questo concetto in maniera più ampia rispetto alla definizione teorica. Innanzitutto il matching non viene applicato soltanto all'intera stringa per la verifica della corrispondenza; normalmente invece viene effettuata una ricerca di sottostringhe che la soddisfino. Naturalmente è possibile specificare che si vuole un matching sull'intera stringa e non su una sottostringa (ed è un caso frequente).

Una volta salvato il nome amichevole del cellulare è necessario creare il file personalizzato e creare il file di configurazione per l'invio del file di configurazione. Il nome del file che verrà inviato è proprio il nome amichevole del cellulare, quindi, ora viene costruita la stringa da scrivere nel file di configurazione che successivamente ObexSender dovrà interpretare; come nel caso di una applicazione SDK nel file passato in arg[2] si scrive la stringa : *file <nome del file>*.

Script 6.2

```
STR1="file "
STR3=".jpg"
STR=$STR1$FN$STR3
```

A questo punto è necessario creare il file che andremo a inviare, invocando il programma ale: advanced label editor

Per la composizione dell'immagine è stato utilizzato un editor di testo inviatoci dal supporto clienti. L'installazione di tale applicazione avviene come una normale installazione di una qualsiasi applicazione attraverso un pacchetto wpk.

Script 6.3

```
Usage:  printf("\n");
printf("Parameters are:\n");
printf("  --help           This help\n");
printf("  --input file     Background image (JPEG, PNG, GIF)\n");
printf("  --output file    Output image (filetype will be same
as input)\n");
printf("  --text text      Text to place on image\n");
printf("  --color ffffff   Text color as hexdecimal\n");
printf("  --size 1..5      Text size (1 smallest, 5 biggest)\n");
printf("  --x          X coordinate to text upper left corner\n");
printf("  --y          Y coordinate to text upper left corner\n");
printf("  --sizex          Width of output image (optional)\n");
printf("  --sizey          Height of output image (optional)\n");
printf("  --resample       Resample instead of resize (slower but
better quality)\n");
printf("  --font           Filename of used font (optional,
font size has no limits after this, use negative value to
disable antialiasing)\n");
```

I comandi di questa applicazione sono molto semplici e permettono di aggiungere una stringa di testo in una qualsiasi immagine in formato JPEG, PNG, GIF. Come si evince dalla sintassi possiamo decidere che font utilizzare, la posizione del testo, il suo colore e la sua dimensione.

Nello script tale programma viene invocato in questo modo.

Script 6.4

```
/usr/bin/ale --input /usr/local/bluetooth.jpg --output ${2}.reply
--text ${FN} --color 000000 --size 5 --x 5 --y 102 1>
/dev/null 2>/dev/null
```

Attraverso *–output ${2}.reply* colleghiamo il file appena creato con il file di configurazione che ObexSender successivamente interpreterà per inviare il file.

Infine è necessario collegare il nome del file con il file di configurazione.

Script 6.5

```
echo $STR > ${2}
```

Questo è lo script completo per inviare un'immagine personalizzata.

Script 6.6

```
!/bin/shBD="$1"

A=‘tail -n 1000 /usr/local/obexsender/obexsender.log | grep Hash
| grep "${BD}"‘

FN=‘expr match "${A}"  ’.*@"\(.*\)"@.*’‘

STR1="file "
STR3=".jpg"
STR=$STR1$FN$STR3
echo $STR


/usr/bin/ale --input /usr/local/bluetooth.jpg --output ${2}.reply
 --text ${FN} --color 000000 --size 5 --x 5 --y 102 1>
/dev/null 2>/dev/null

echo $STR > ${2}

exit 0
```

Capitolo 7

L'APPLICAZIONE BACHECA DIGITALE

7.1 Introduzione

Una volta presa confidenza con il Software Development Kit il progetto successivo ha previsto la realizzazione di una bacheca digitale. Di solito una bacheca contiene avvisi, circolari, informazioni utili. L'applicazione propone contenuti ad un pubblico ristretto offrendo contenuti diversi: file di testo, audio, video.

Si è deciso di realizzare due versioni, una prima che funzionasse stand alone, e una seconda ottimizzata per gestire in modo più efficace la comunicazione con il cellulare grazie ad un client da installare nel cellulare.

7.2 Bacheca digitale base

Vengono create delle categorie di utenti e l'applicazione invia al cellulare tutti i file che ha diritto di ricevere. Se il cellulare ritorna successivamente riceve i nuovi file oppure viene avvisato che non ci sono aggiornamenti.

Dall'analisi di questo semplice use case diagram e per la tutela della informazioni sono stai estratti i seguenti obiettivi.

Obiettivi Bacheca digitale base

- Per motivi di privacy solo i cellulari abilitati devono poter ricevere i contenuti
- Creazione di gruppi di utenti in base alla categoria

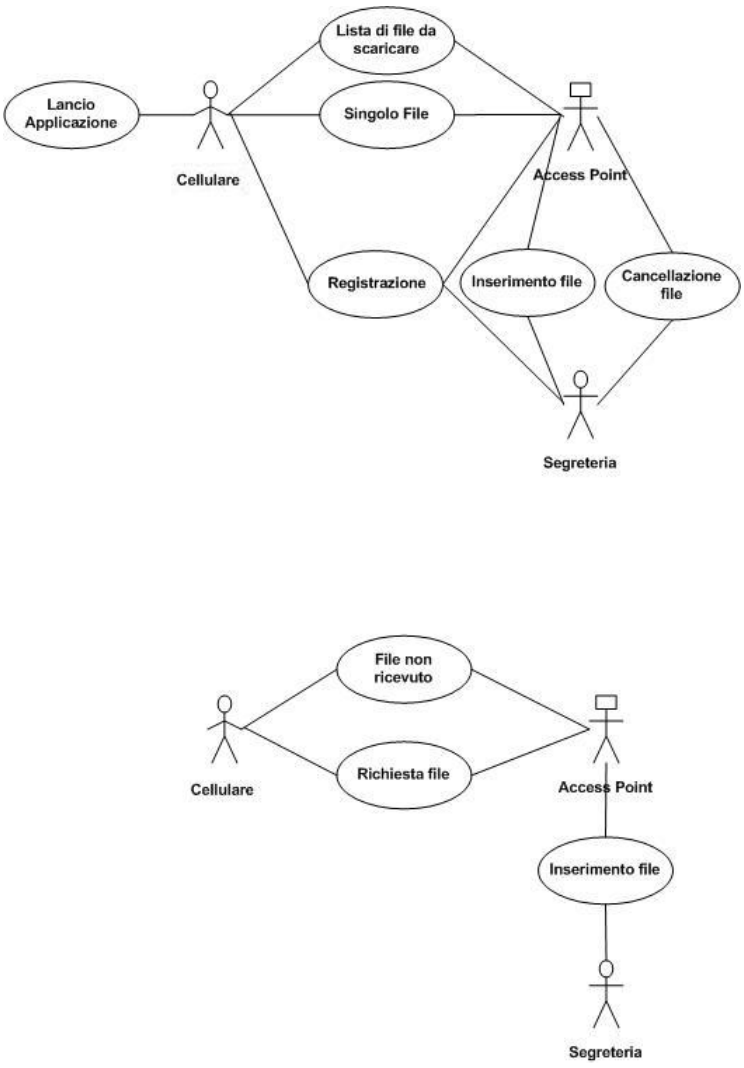

Figura 7.1: Use case bacheca digitale

Nome	File non ricevuto
Attori	Il cellulare e l'access point
Precondizioni	Il cellulare deve avere il Bluetooth accesso, visibile a tutti e accettare la richiesta per scaricare i contenuti
Flusso principale	Se il cellulare non ha ricevuto tutti i file riceve solo quelli che non ha già scaricato
Flusso secondario	Se il cellulare ha già ricevuto tutti i file riceve una un file di testo con i nomi dei file che dovrebbe aver scaricato
Postcondizione	Il cellulare ha ricevuto tutti i contenuti, nel caso avess già ricevuto tutti i contenuti durante una precedente comunicazione riceve una lista con il nome dei file che dovrebbe aver ricevuto

Tabella 7.1: Use case per la ricezione di un file non precedentemente ricevuto

Nome	Inserimento File
Attori	La segreteria e l'access point
Precondizioni	Autentificazione dell'operatore
Flusso principale	La segreteria carica un file nell'access point associandolo alla categoria di destinazione
Postcondizione	File inserito nella corretta directory con la corretta associazione alla categoria

Tabella 7.2: Use case per l'inserimento di un file

Nome	Richiesta File
Attori	Il cellulare e l'access point
Precondizioni	Il cellulare ha già ricevuto tutti i file e ricevuto l'elenco dei file disponibili
Flusso principale	L'utente invia un messaggio con testo il nome del file che vuole ricevere all'acces Point. L'access point risponde inviando tale file
Flusso secondario	Se l'utente invia un nome non corretto l'access point lo informa con un messaggio di errore
Postcondizione	Il cellulare ha ricevuto il file che aveva richiesto

Tabella 7.3: Richiesta di un file

- Un cellulare non deve ricevere due volte lo stesso contenuto a meno che non sia il cliente a richiederlo in modo esplicito.
- Nel caso vengano aggiunti alcuni file il cliente deve riceve solamente i file che non ha già ricevuto a meno che non sia il cliente a richiederlo in modo esplicito.
- In caso il cliente abbia già ricevuto tutti i documenti deve riceve un avviso che lo avverti con una lista di tutti i file presenti nella campagna
- Il sistema deve supportare almeno 5 file
- Il cliente deve poter richiedere uno specifico documento
- Deve essere possibile contare i contatti univoci.
- Il sistema deve registrare l'ora e il giorno del contatto con il cellulare.

L'applicazione

Quando l'applicazione viene invocata ha come parametri di input, come detto, il codice macchina del dispositivo che andrà a servire. Attraverso questa informazione e un file di log creato dalla applicazione stessa è possibile risalire in ogni momento a quale file siano stati già inviati precedentemente a quel particolare dispositivo.
In questo modo quando un cellulare ritorna nel raggio del dispositivo riceve solamente i file inseriti dopo il suo precedente passaggio. Nel caso che tutti i file nella cartella contenuti siano stati precedentemente ricevuti verrà inviato un file di testo con un lista di tutti i contenuti associati ad un codice. Attraverso questo codice usando la funzione reply dei device potrà richiedere l'invio di un singolo file. Inoltre va considerato che non tutti gli utenti sono destinatari di tutti di documenti; per questo motivo è possibile creare categorie di utenti. Nel file di configurazione di ObexSender viene creata una sending rule per ogni categoria, per ognuna delle quali verrà creata una cartella nella quale caricare i contenuti. In questo modo oltre a rispettare le regole sulla privacy vengono inviati all'utente solamente i documenti realmente interessanti, aumentando il grado di soddisfazione dell'utente finale.

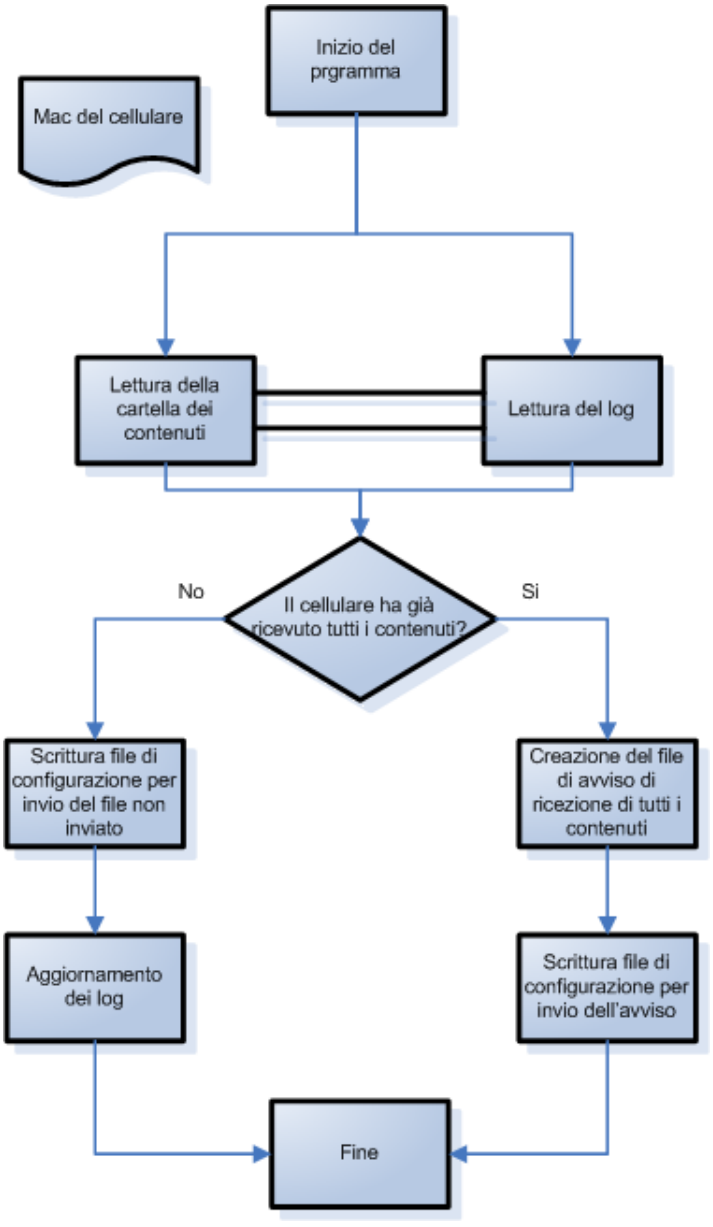

Figura 7.2: Applicazione bacheca digitale

Lo sviluppo dell'applicazione ha reso necessario superare due limiti di ObexSender già sottolineati precedentemente:

- L'impossibilità di conoscere quali messaggi siano stati inviati
- L'impossibilità di inviare due contenuti distinti con una sola applicazione

L'applicazione, ricevendo in input il mac dell cellulare da servire, prima di decidere cosa inviare consulta il file di log per ricostruire i file effettivamente i file ricevuti dall'utente, per i dettagli si faccia riferimento al paragrafo 6.1.1 Limitazioni.

Per inviare più file in un'unica transizione è stato necessario creare numerose applicazioni che permettessero di inviare i file in modo coerente in base alla loro posizione. Il primo sottoprogramma invierà un file che l'utente non ha, il secondo sottoprogramma invierà un'altro file che l'utente non ha ancora ricevuto e cosivia, fino a finire i file da inviare. Se l'ennesima applicazione ha inviato l'ultimo file, l'applicazione n+1 invia una lista con i file che l'utente dovrebbe aver ricevuto mentre le successive terminano senza inviare nulla.

Estratto del file Obexsender.conf 7.1 ***Esempio Sending Rule per la bacheca digitale base***

```
send {
#sending rule per la categoria ABC
match 00:23:62:fr:31:6a #1)Gialli Luca
match 00:23:62:fr:31:6a #2)Rossi Marco
match 00:25:3t:ty:88:g4 #3)Verdi Laura
#...
exec /usr/local/obexsender/bin/ABC_1
exec /usr/local/obexsender/bin/ABC_2
exec /usr/local/obexsender/bin/ABC_3
exec /usr/local/obexsender/bin/ABC_4
}
```

Il sistema permette all'utente di ignorare il messaggio o di rifiutare i contenuti senza inficiare il corretto funzionamento dell'applicazione. Ad ogni categoria è associata una cartella nella qualche vengono salvati i file a questa destinati, in questo modo è possibile monitorare facilmente quali siano i file caricati per ogni categoria. Per questa prima versione di prova le operazioni di: inserimento file, registrazione di un utente e stesura della reply rule per la richiesta di un

particolare contenuto sono realizzabili solamente da un utente esperto agendo direttamente sui file di configurazione dell'access point. Come si vedrà a breve per la versione finale queste operazioni verranno gestite attraverso una interfaccia web.

Figura 7.3: Esempio file system per la bacheca digitale

7.3 Interazione con i cellulari

Una volta realizzata questa applicazione lo studio si è spostato sulla possibilità di interazione tra cellulare e access point. La funzione di reply come detto precedentemente permette all'utente di esprimere un esigenza cognitiva; inviando un messaggio di testo con una particolare keyword è possibile riceve un determinato contenuto. É necessario valutare attentamente due aspetti:

- Per comunicare con l'access point è necessario utilizzare un nota in formato .txt, la creazione di questo file può risultare difficile e noisa, inoltre prima di poter inviare questa nota al dispositivo è necessario che il cellulare esegua un'operazione di scan e che l'utente selezioni il corretto dispositivo. Tutte queste operazioni scoraggerebbero chiunque da tentare una qualsiasi interazione con il device.

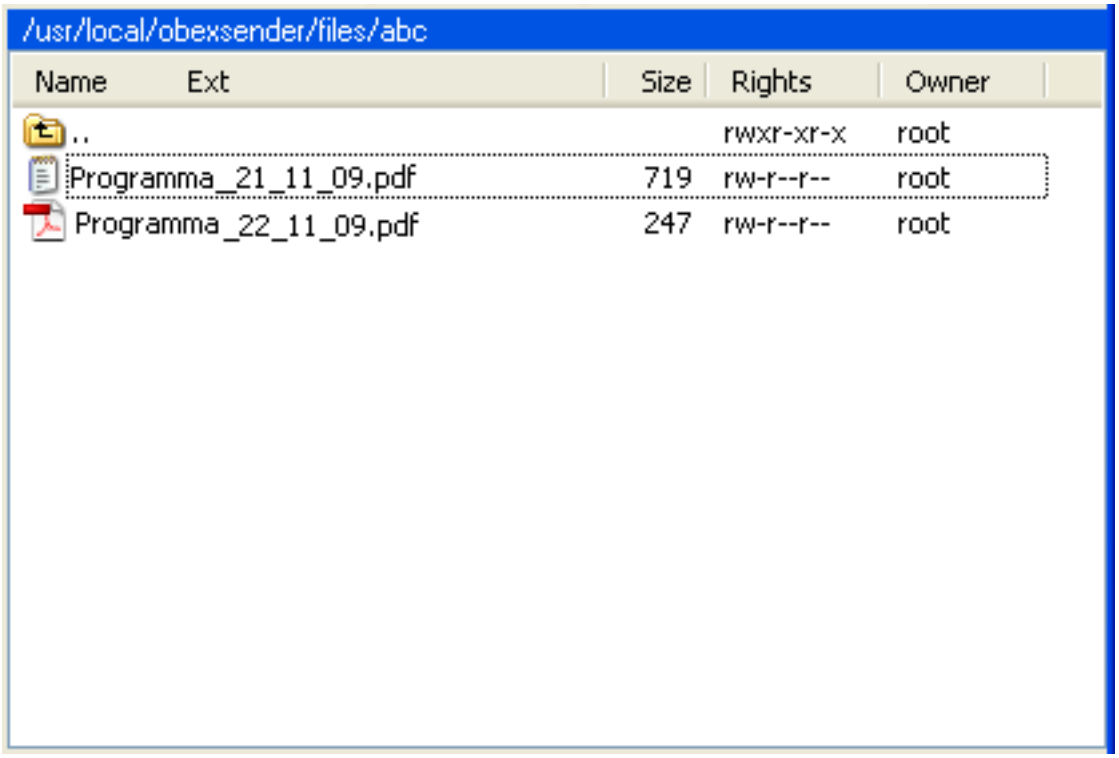

Figura 7.4: Esempio, file da inviare agli utenti abc

- Per ogni trasmissione il cliente deve accettare il prompt di connessione, operazione necessaria per un corretto funzionamento della specifica Bluetooth, indispensabile per salvaguardare la privacy, ma che aumenta la difficoltà nell'utilzzare questa interessante features.

Si è quindi deciso di realizzare un applicazione per cellulari che permettesse di velocizzare queste procedure. L'applicazione ha come obiettivo di rendere trasparente all'utente tutte le operazioni di sincronizzazione dei device e di fornire una interfaccia user friendly per permettere all'utente di usufruire di tutti i servizi senza la necessità di scivere file di testo. Grazie a questa applicazione mobile viene completamente ribaltato il paradigma di comunicazione. Mentre prima era l'access point a iniziare la comunicazione inviando dati, in base alle informazioni di cui disponeva, ora può essere l'utente a decidere di iniziare una connessione, scegliendo da una lista di possibili file quale scaricare.

É stato scelto di iniziare il progetto utilizzando J2ME, micro edition, una versione di Java ottimizzata per cellulari che permettere di creare applicazioni perfettamente compatibili con i sistemi Symbian. Interfacciandosi direttamente alle API del cellulare è stato possibile creare applicazioni che scavalcassero i

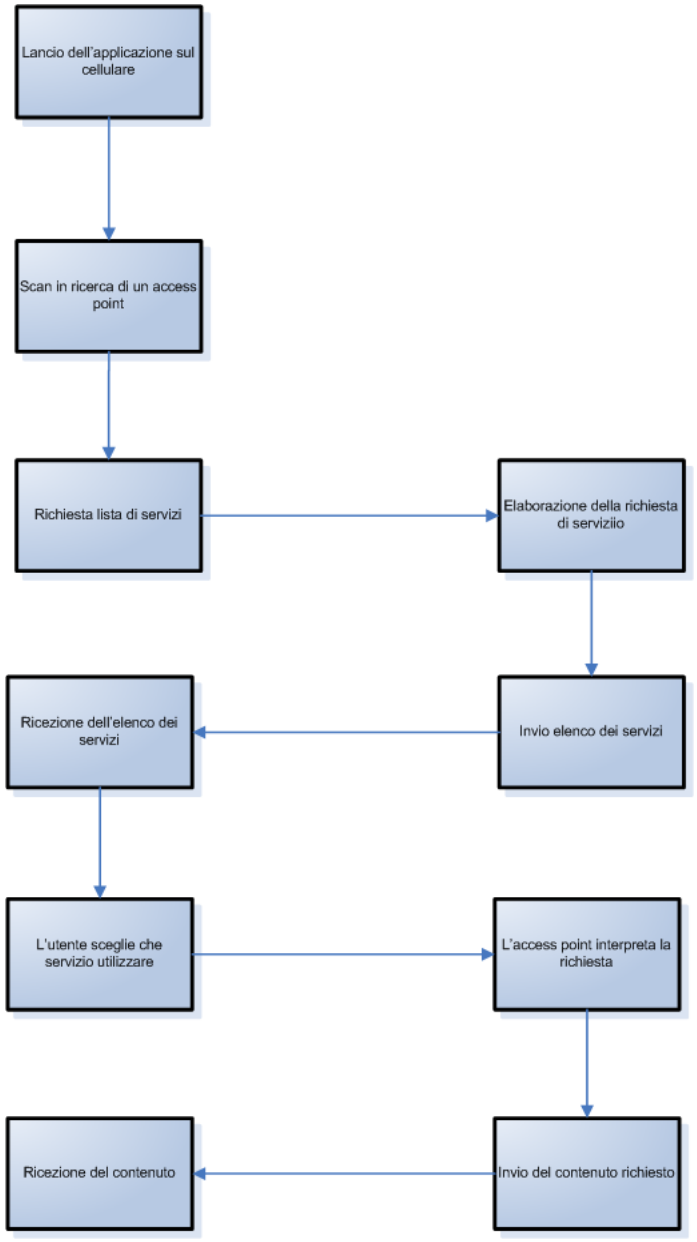

Figura 7.5: Nuovo paradigma di comunicazione

programmi forniti dal cellulare stesso per realizzare programmi che gestissero la comunicazione in toto.

7.4 Bachecha Digitale Client/Server

Il primo progetto pilota per questa applicazione è stata la bacheca digitale. Come detto in precedenza la bacheca invia in modo automatico ai cellulari target tutti e soli i file che non hanno già ricevuto. Risulta evidente però che l'utente poterebbe ricevere un file, a lui destinato, ma che non gli interessi o che l'utente riceva files in un momento in cui non intende leggerli. Benchè attraverso la funzione di reply sia possibile richiedere un contenuto, come detto l'interazione con l'access point non è intuitiva. Per questo motivo si è scelto di utilizzare l'applicazione mobile per migliorare la bacheca digitale.

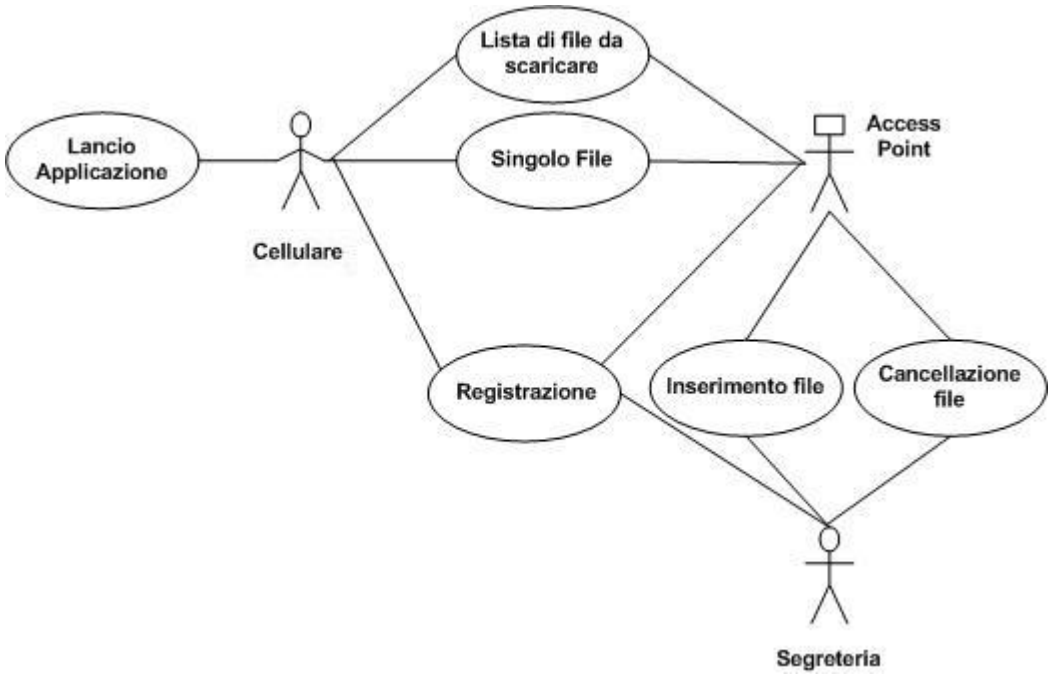

Figura 7.6: Use case bacheca avanzata

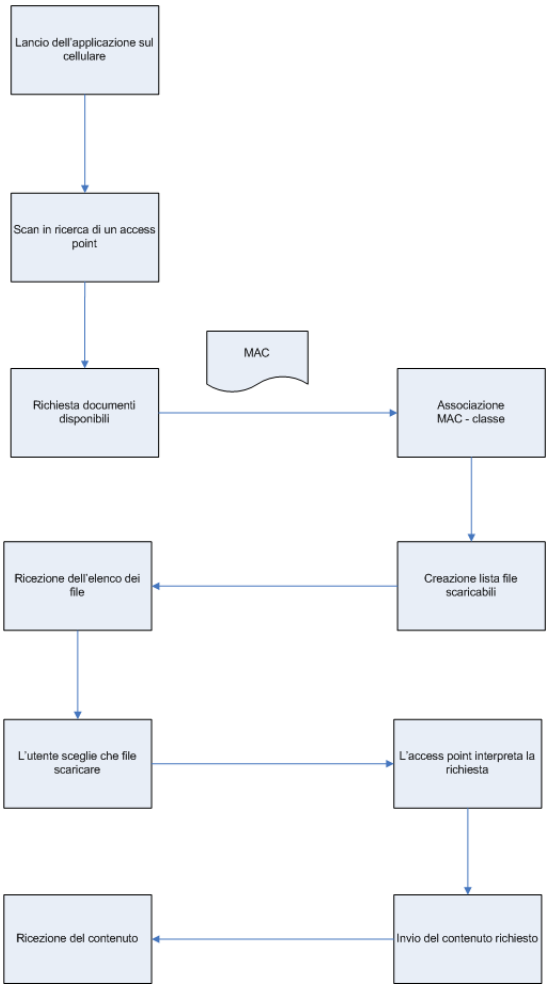

Figura 7.7: Interazione per Bacheca Digitale Client/Server

Nome	Registrazione
Attori	Cellulare, Access Point, Segreteria
Precondizioni	Il cellulare deve avere il Bluetooth accesso, visibile a tutti e accettare la richiesta per scaricare i contenuti
Flusso principale	La segreteria comanda l'access point l'invio al cellulare dell'utente dell'applicazione lato clienti. Successivamente abilita tale cellulare alla ricezione dei file di una categoria.
Postcondizione	L'utente ha l'applicazione da installare nel proprio cellulare ed il cellulare è stato registrato e associato alla categoria d'appartenenza del figlio.

Tabella 7.4: Use case per la registrazione di un utente

Nome	Inserimento file
Attori	Access Point, Segreteria
Precondizioni	Segreteria autentificata tramite password, file non inserito precedentemente
Flusso principale	La segreteria carica un nuovo file associandolo alla categoria di destinazione attraverso l'apposita interfaccia
Postcondizione	Il file è stato caricato all'interno del sistema e associato alla categoria

Tabella 7.5: Use case per l'inserimento di un file

Nome	Cancellazione file
Attori	Access Point, Segreteria
Precondizioni	Segreteria autentificata tramite password, file inserito
Flusso principale	La segreteria cancella un file attraverso l'interfaccia
Postcondizione	Il file è stato cancellato

Tabella 7.6: Use case per la cancellazione di un file

7.4.1 Obiettivi Bacheca digitale client/server

Dopo il diagramma dei casi d'uso vengono presentati gli obiettivi di questa soluzione. Gli obiettivi vanno divisi in tre gruppi:

1. Obiettivi legati all'access point

   (a) Per motivi di privacy solo i cellulari abilitati devono poter ricevere i contenuti.

Nome	Lancio Applicazione
Attori	Cellulare, Utente
Precondizioni	Applicazione Scaricata
Flusso principale	L'utente in prossimità dell'access point accede all'applicazione
Postcondizione	Applicazione avviata con successo e pronta a comunicare con l'Access Point

Tabella 7.7: Use case per il lancio dell'applicazione

Nome	Lista file da scaricare
Attori	Cellulare, Access Point
Precondizioni	Cellulare Registrato, Applicazione attivata
Flusso principale	L'applicazione si connette automaticamente con l'access point, inviado il mac univoco del dispositivo. L'Access Point interpretandolo restituisce al cellulare la lista dei file disponibili
Postcondizione	Il cellulare ha ricevuto la lista dei file disponibili ed ora l'utente può scegliere che file scaricare

Tabella 7.8: Use case per la ricezione dei file da scaricare

Nome	Singolo File
Attori	Access Point, Cellulare e Utente
Precondizioni	Cellulare registrato, applicazione lanciata, lista file ricevuta
Flusso principale	L'utente attraverso l'applicazione richiede l'invio di un file che l'access point gli fornisce direttamente sul cellulare
Postcondizione	L'utente può consultare sul suo cellulare il file appena ricevuto.

Tabella 7.9: Use case per la ricezione di un singolo file

(b) Creazione di gruppi di utenti.

(c) Il sistema deve supportare almeno 10 file.

(d) Il file di log deve poter registrare ogni transizione

(e) Il sistema deve registrare l'ora e il giorno del contatto con il cellulare

(f) L'inserimento, la cancellazione di un file, la registrazione di un cellulare devono essere gestiti da una interfaccia grafica semplice ed intuitiva.

(g) Il sistema deve gestire file con nomi uguali destinati però a diverse classi

2. Obiettivi legati alla applicazione per il cellulare

   (a) Il cellulare deve creare una interfaccia grafica amichevole con la quale selezionare i file da scaricare

   (b) Una volta selezionato il file da ricevere il cliente deve poter ricevere il file senza alcun'altra operazione

   (c) L'applicazione lato cellulare deve essere compatibile con il cellulare di test Nokia N73

3. Obiettivi in comune all'applicazione per il cellulare e all'applicazione per l'access point

   (a) L'applicazione deve riconoscere l'access point destinato a questo servizio

   (b) L'applicazione deve poter leggere correttamente la lista dei file che ha diritto di scaricare

   (c) L'applicazione deve riuscire a inviare correttamente una richiesta per scaricare un certo file

   (d) Il cliente deve poter richiedere uno specifico documento

   (e) L'applicazione lato cellulare deve essere scaricabile via Bluetooth

   (f) Ogni cellulare deve poter scaricare solo i file di cui detiene di diritti.

7.4.2 Raggiungimento degli obiettivi

Per raggiungere gli obbiettivi sono state create 3 applicazioni ex novo ed è stato necessario ObexSender.

1. La *applicazione lato access point* si occupa di creare in modo dinamico la lista dei file che ogni cellulare può scaricare.

2. L'*interfaccia utente* lato access point, raggiungibile tramite in web browser via ethernet, permette di gestire semplicemente la campagna.

3. Infine l'*applicazione lato celluare* ottimizza la comunicazione con l'access point.

Obiettivi legati all'access point

Per ogni categoria viene creata una cartella nella quale verrano caricati tutti destinata ad questa, e ogni MAC registrato viene memorizzato nel file /usr/bin/ObexSender/classi.txt associato alla categoria appropriata con la formattazione: categoria/MAC.
Esempio di file classi.txt:
1a/00:22:66:da:31:6a
1a/00:22:66:bg:f4:a7
2a/00:15:a0:51:d7:ae

L'applicazione viene invocata con il MAC del cellulare da servire come paramentro di input, in questo modo viene costruita in modo dinamico la lista dei file che il cellulare può scaricate. In questo modo vengono raggiunti gli obiettivi 1a e 1b. L'obiettivo 1c viene risolto grazie al ribaltamento del paradigma, visto che è l'utente a richiedere il file, non è più necessario creare sending rule complesse per inviare più file in un unica transizione. La reply rule sarà molto semplice.

Estratto del file Obexsender.conf 7.2 ***Esempio reply rule***

```
    reply {
keyword file.pdf
file file.pdf
}
```

Tale soluzione permette di soddisfare anche 1d e 1e, infatti in questo modo utilizziamo ObexSender per l'invio dei file, lasciando all'access point la gestione dei log, features a lungo testata e che si è dimostrata uno dei punti forti dei prodotti. Per l'obiettivo 1f viene creata una applicazione in bash di linux per creare delle pagine html per gestire l'applicazione. Tale applicazione ha tre funzioni: inserire un file, cancellare un file e registrare un cellulare.

- Inserire un file
  L'inserimento di un file è un operazione complessa; tramite il comando browse è possibile selezionare il file desiderato dal filesystem, successivamente è necessario selezionare la categoria al quale il file è destinato. Una volta caricato il file l'applicazione, oltre a inserirlo nella corretta cartella lo rinomina con la seguente sintassi:
  categoria.data(GG.MM.AAAA.HH.MM.SS).nomefile.estensione e modifica il file di configurazione inserendo l'appropriata reply rule. In una questa

prima versione la parola chiave è il nome del file stesso, in una versione futura si utilizzerà una funzione di hash per proteggere il sistema da eventuali tentativi di download fraudolenti. La lunghezza della parola chiave non è un problema visto che il messaggio per attivare la reply rule sarà attivata dalla applicazione lato cellulare in modo automatico. Si noti che rinominando in questo modo il file il sistema è perfettamente in grado di gestire file con nome univoci soddisfacendo il punto 1g.

- Cancellazione di un file
  Selezionando sul link di una categoria è possibile controllare quali file siano associati ad una categoria e tramite un apposito bottone è possibile cancellare il file, la cancellazione comporta anche l'eliminazione della reply rule associata.

- Registrazione di un utente
  La registrazione di un utente avviene in due fasi:

  1. Al cellulare viene inviata l'applicazione lato cellulare e attraverso la prima applicazione presentata nel capitolo 6, un file di testo con l'indirizzo MAC del cellulare.

     Estratto del file Obexsender.conf 7.3 ***Sending rule necessaria per la registrazione***

```
whitelistrssi -45
send {
exex exec /usr/local/obexsender/bin/info
file bacheca_ mobile.jar
}
```

     La sending rule, viene attivata solo se ci si avvicina a meno di 50 cm, in questo modo chi ha già ricevuto l'applicazione non la riceverà di nuovo, e chi l'avesse cancellata per sbaglio può riscaricarla nuovamente.

  2. Successivamente l'utente comunica il proprio numero di MAC alla segreteria che, attraverso l'interfaccia web, lo registra associandolo alla categoria.

Obiettivi legati all'applicazione per cellulare

L'applicazione, una volta lanciata, richiede la lista dei file che il cellulare ha diritto di scaricare, l'access point, come già descritto sopra risponde con una lista

di file in formato txt. L'applicazione, ripulisce i nomi dei file togliendo i campi categoria, data e espansione, e presenta all'utente una serie di pulsanti con i quali selezionare i file disponibili (2a). Una volta che l'utente attiva il pulsante l'applicazione genera la richiesta da inviare all'access point, un file di testo che ha come corpo, in questa prima versione, il nome del file richiesto. Quando l'access point invia il file richiesto, l'applicazione risponde in modo positivo al prompt di conferma di ricezione del messaggio rendendolo disponibile all'utente. L'applicazione semplifica la comunicazione in modo radicale, da una parte crea il messaggio da inviare all'access point, dall'altra parte rende trasparente all'utente la nuova comunicazione, in questo modo viene soddisfatto 2b.

Obiettivi in comune all'applicazione per il cellulare e all'applicazione per l'access point

La parte più interessante della soluzione risiede nello scambio dei messaggi tra i due attori: l'applicazione lato cellulare e l'applicazione lato access point. Per soddisfare 3a all'interno dell'applicazione è presente l'indirizzo MAC dell'access point destinato all'invio dei documenti, la lista dei file di scaricare viene inviata con un codifica ben precisa, in formato .txt, quindi semplice a reinterpretare per l'applicazione stessa (3b).

L'applicazione lato cellulare inizia con una operazione di scan cercando l'access point, una volta trovato invia la stringa _bacheca_digitale_ che attiva la seguente reply rule.

Estratto del file Obexsender.conf 7.4 ***Sending rule per attivare la bacheca digitale avanzata***

```
reply {
keyword _bacheca_digitale_
exex exec /usr/local/obexsender/bin/bacheca_avanzata }
```

Nonostante sia possibile realizzare un'applicazione SDK che permetta di leggere leggere i file inviati dalla applicazione lato cellulare l'invio è gestito attraverso la funzione reply di ObexSender per 3 motivi:

- Semplicità, è una scelta ovvia, vengono utilizzati i comandi ObexSender direttamente senza utilizzare un altra applicazione SDK che rallenterebbe il tempo di risposta.

- Log, utilizzando direttamente la funzione di reply, di default il log creati mostrano quale cellulare abbia scaricato quale file direttamente, senza la necessità di aggiungere ulteriori righe di codice.

- Infine alcuni cellulari inviano i file di testo in modo differente, per ulteriori dettagli si faccia riferimento al capitolo dedicato alla funzione reply di ObexSender.

Attraverso l'applicazione l'utente può richiedere in modo semplice solamente tutti e solamente i file che ha diritto di scaricare (3d e 3f). Infine 3e viene raggiunto salvando l'applicazione lato cellulare in formato .jar, formato che permette un facile utilizzo su tutti i cellulari symbian. La procedura cambia da cellulare a cellulare: mentre il modello N73 necessità di istallare il programma, nel modello 5200 tale passaggio non è necessario.

Interfaccia grafica lato access point

Come detto precedentemente l'interfaccia grafica deve gestire i file e le registrazioni dei cellulari.

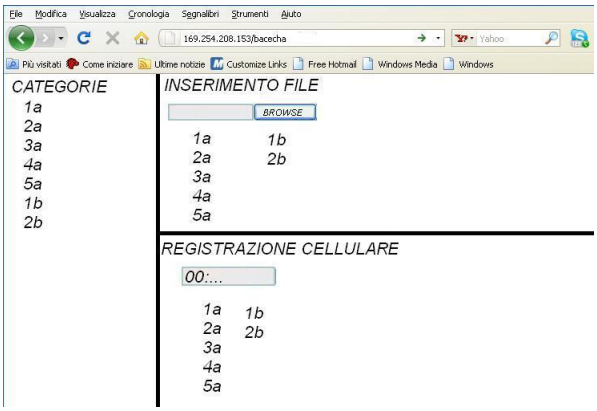

Figura 7.8: Interfaccia 1 per Bacheca digitale Client/Server

Attraverso il pulsante Browse è possibile navigare il file system e ricercare il file desiderato, una volta selezionato, premendo il link della categoria di destinazione viene avviato lo script che termina l'inserimento. Prima di completare l'inserimento lo script controlla che il file non sia già presente, e in caso positivo lo rinomina con la sintassi già descritta precedentemente e crea la reply rule. La procedura per l'inserimento di un cellulare è uguale. E' sufficiente inserire il mac del dispositivo e premere la categoria al quale deve essere associato. In caso il file

che si cerca di inserire, o il cellulare ,sia già presente viene generato un messaggio di errore e l'operazione viene interrotta.

Selezionando invece sul link delle classi nella colonna di sinistra è possibile vedere tutti i file destinati a quella particolare categoria. Premendo su un link del file otterremo la sua cancellazione. Tale operazione cancella anche la reply rule nel file di configurazione di ObexSender.

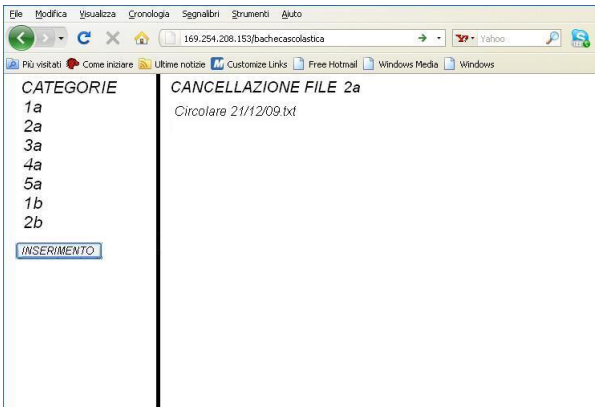

Figura 7.9: Interfaccia 2 per Bacheca digitale Client/Server

Interfaccia grafica lato mobile

L'interfaccia grafica per il cellulare è stata creata utilizzando i template messi a disposizione da Java 2 ME per essere portabile sul maggior numero di cellulari possibili.

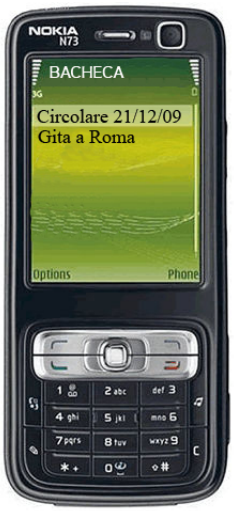

Figura 7.10: Interfaccia client per cellulare

L'applicazione non appena viene lanciata ricerca l'access point associato e dopo pochi istanti restituisce la lista dei file che l'utente può scaricare. Con il cursore è possibile scorrere i file. Premendo su Option o premendo il joystick inizierà la procedura di comunicazione tra il cellulare e access point; come detto la procedura è invisibile all'utente e dopo pochi istanti il file desiderato sarà disponibile sul display.

Project Management

Vista la complessità della soluzione in fase si sviluppo, per garantire il controllo e permettere ai miei collaboratori di continuare lo sviluppo anche successivamente il mio tirocinio, oltre agli Use Case mi sono servito del WBS e della tabella dei rischi. Il Work Breakdown Structure, una struttura gerarchica in cui vengono raccolte tutte le attività relative al progetto, in figura 7.11, sono evidenziati lo stato di avanzamento di ogni attività e la persona responsabile del suo sviluppo. Viene qui presentata la versione aggiornata al 20/04/2010.

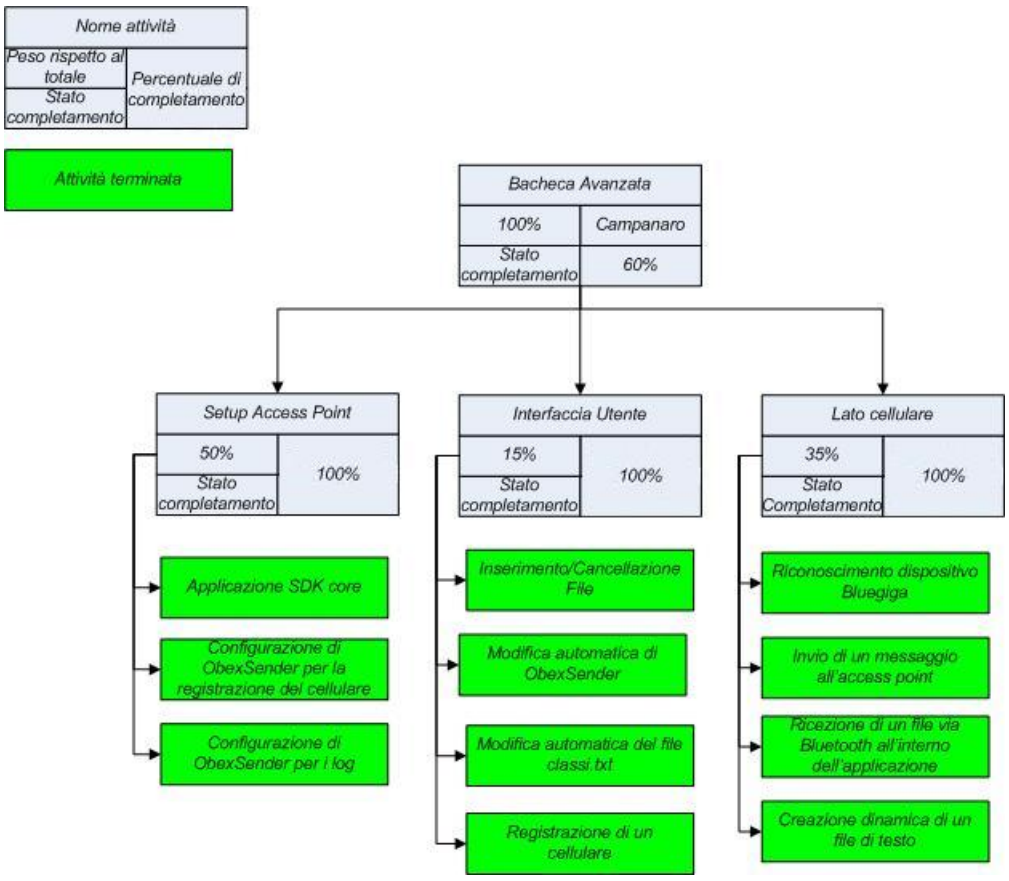

Figura 7.11: WBS, 12/04/2010

Inoltre tenuto conto della diversità dei moduli, e le diverse responsabilità è stata stilata una tabella dei rischi, che in figura 7.12 viene riportata aggiornata al 12/04/2010.

	Tabellla dei rischi albo scolastica avanzato			
Id	Nome	Stato	Criticità	Soluzione
	Lato Access Point			
1	Access Point non interpreta messaggi dal cellulare		Estrema	Utilizzo di ObexSender
2	Access Point non riesce a riconoscere il cellulare che invia il messaggio tramite l'applicazione		Estrema	Utilizzo di ObexSender
3	Modifica dei file di configurazione via script		Alta	Funzionalità già testata
4	File con nomi uguali, o con spazi		Bassa	Rinominare i file
	Lato Cellulare			
5	L'applicazione non riesce a riconoscere l'Access Point		Estrema	Riconoscimento tramite MAC
6	L'applicazione non riesce a inviare file riconoscibili dall'Access Point		Estrema	Invio di file di testo testato.
7	L'applicazione non permette di ricevere messaggi		Estrema	L'applicazione non blocca il canale.
8	L'applicazione non riesce a catturare la lista dei file da scaricare		Media	Lettura copia del file salvata in memoria del telefono
9	L'applicazione non risponde in modo autonomo alla richiesta di ricezione del cellulare		Media	Ad ogni file ricevuto l'utente riceverà una conferma

Figura 7.12: Tabella dei rischi, 12/04/2010

Come è possibile vedere dalla tabella 7.12 tutti gli elementi sono stai già risolti con successo.

Capitolo 8

IL PROGETTO BLUE-BEAUTY

8.1 Introduzione

Nonostante il design accattivante e le dimensioni contenute l'Access Point necessita di essere alimentato. Uno degli aspetti fondamentali di una buona campagna di proximity marketing è l'efficacia nel raggiungere gli utenti finali, per la natura del nostro media, è indispensabile che la posizione fisica del dispositivo sia indipendente da più variabili possibili; inoltre, nonostante il segnale del'access point sia di classe 1, non va dimenticato che quello dei cellulari ha un raggio di solo 20 metri; che in situazioni di disturbo possono ridursi fino a 8. Gli scenari di utilizzo, purtroppo, fiere, stadi, centri commerciali si caratterizzano per la presenza di numerosi utenti, quindi si spera, di numerosi dispositivi Bluetooth che diventano una fonte di disturbo l'uno per l'altro. Oltretutto non è detto che il luogo scelto per l'installazione abbia a disposizione una presa di corrente. Inoltre un prodotto che non deve essere alimentato, che funzioni indipendentemente da ogni variabile e che non necessiti di nessuna iterazione da parte del cliente intermedio risulta più interessante e commercialmente appetibile. Tal esigenza è stata messa in evidenza anche dal settore commerciale di Telerete: soprattutto in una prima fase di presentazione del servizio, fornire un prodotto che sia in grado di funzionare as is è sicuramente di maggior impatto emotivo. Inoltre ciò permette di presentare al cliente intermedio un servizio ancor di più chiavi in mano. Per questi motivi ho cercato di rendere portatile l'Access Point: si è scelto di inscatolare il device con una batteria.

8.2 Gli obiettivi

Prima di realizzare il prototipo era fondamentale stilare una lista di prerequisiti che delineassero il risultato finale, grazie ad alcune interviste con il settore commerciale e al supporto dell'area tecnica sono stati stilati una serie di requisiti minimi e una serie di requisiti opzionali

8.2.1 Gli obiettivi minimi

- Durata ininterrotta di 12 ore.
- Durata non ininterrotta di 36 ore .
- Dimensioni massime 30 cm x 30 cm x 10 cm
- Peso massimo 5 Kg
- Possibilità di spegnere il device con un interruttore.
- La carica della batteria deve potersi realizzare con il dispositivo spento.
- Schermatura del segnale inferiore al 25%
- Non infrangere la garanzia del fornitore

8.2.2 Gli obiettivi Opzionali

- Costo Contenuto
- Design accattivante
- Schermatura del segnale inferiore al 10%
- Carica della batteria inferiore alle 10 h.
- Segnalatore di fine carica
- Scatola apribile
- Indicatore ON/OFF.
- Possibilità di programmare il dispositivo via ETH o USB

Analisi degli obiettivi

- **Durata ininterrotta di 12 ore**
  Durata giornaliera di una fiera, era già stato effettuato un test in cui l'access point è stato mantenuto acceso per 14 ore funzionando correttamente.

- **Durata di trentasei ore non ininterrotta** Periodo di tempo stimato di funzionamento durante una tipica fiera di 3 giorni.

- **Dimensioni massime 300 mm x 300 mm x 100 mm** Il prototipo deve essere portatile, quindi deve avere delle dimensioni che ne permettano uno spostamento comodo, il vincolo non è particolarmente stretto.

- **Peso massimo 5 Kg** Per il motivo già citato anche il peso deve essere contenuto, il componente più pesante è stata la batteria, l'access point, l'interruttore e i morsetti hanno un peso trascurabile.

- **Possibilità di spegnere il device con un interruttore**. Per prolungare il tempo di funzionamento del device e non costringere il cliente intermedio ad aprire il prototipo deve essere possibile spegnere il device con un interruttore esterno. L'assenza di una soluzione del genere avrebbe reso necessario usare delle batterie per il funzionamento ininterrotto di almeno 80 ore per essere impiegato in una fiera (almeno 3 giorni).

- **La carica della batteria deve potersi realizzare con il dispositivo spento**
  Questo vincolo è stato imposto per motivi di sicurezza, infatti durante la carica della batteria, specialmente al momento di inserimento ed estrazione della spina si possono verificare degli abbassamenti di tensioni pericolosi per il device stesso.

- **Schermatura del segnale inferiore al 25%**
  Oltre al limite alla dimensione e al peso è necessario porre un vincolo alla potenza del segnale emesso; infatti se il segnale emesso dal prototipo non fosse necessariamente potente non saremmo in grado di raggiungere i cellulari. Escludendo quindi a priori tutti i materiali di tipo metallico, lo studio se è concentrato sui materiali plastici

- **Non infrangere la garanzia del fornitore**
  Ciò si traduce nella impossibilità di modificare e aprire l'access point o l'alimentatore.

- **Costo contenuto** Il progetto deve avere il costo più basso possibile.
- **Design accattivante** Questo obiettivo è stato tralasciato in un'ottica di minimizzazione dei costi e delle dimensioni. Una versione successiva, con vincoli di dimensioni e di costo meno rigido potrà soddisfare anche questo requisito.
- **Schermatura del segnale inferiore al 10%** Se possibile sarebbe preferibile una schermatura minima per non inficiare il raggio di trasmissione del segnale.
- **Carica della batteria inferiore alle 10 h** La carica della batteria deve potersi completare nell'arco di una notte in modo da essere riutilizzabile la mattina. Se utilizzato in una fiera, il dispositivo deve poter essere utilizzabile giorno dopo giorno.
- **Segnalatore di fine carica** Il prototipo deve potere essere utilizzabile da chiunque, per questo motivo deve essere chiaro lo stato della batteria.
- **Scatola apribile** La scatola deve essere facilmente apribile in modo da poter recuperare l'access point. La scatola scelta ha un coperchio bloccato da delle viti che ne permettono una facile apertura.
- **Indicatore ON/OFF** In ogni momento deve essere reso comprensibile lo stato del dispositivo. Quest'obiettivo è stato raggiunto solamente in parte; l'inserimento di un led per verificarne l'accensione avrebbe comportato un discreto aumento del costo.
- **Possibilità di programmare il dispositivo via ETH o USB** Per questioni di comodità sarebbe preferibile poter programmare l'access point senza doverlo estrarre dalla scatola, è quindi necessario raggiungerlo dall'esterno con una presa eth o usb.

8.3 Hardware utilizzato

Il primo problema affrontato è quello della alimentazione: era necessario soddisfare i requisiti di durata, scegliendo valori di voltaggio che permettessero di lavorare in sicurezza senza danneggiare la macchina. Questi sono i dettagli inviatoci dal servizio clienti:

Consumo di AP			
Voltage (V):	9	12	15 V
Average (mA):	170	130	109
Maximum (mA):	218	152	137
AVG power (W)	1,53	1,56	1,635
MAX power (W)	1,962	1,824	2,055

Tabella 8.1: Consumo dell'access point

Questo vincolo ha imposto un calcolo sulla batteria con cui alimentare il dispositivo. Si è scelto di utilizzare una batteria al piombo già presente in azienda; con una corrente con un voltaggio di 12. e 7 Ah che fornisce potenza sufficiente per mantenere acceso il device per le 36 ore richieste. Nonostante le batterie al piombo abbiano rapporto energia / peso specifico piuttosto basso il loro costo è molto inferiore ad altre tecnologie più performanti. Per la realizzazione del prototipo, come detto, è stata data priorità alla minimizzazione dei costi [24].

Figura 8.1: Batteria Winner da 12 V e 7 Ah

Dettagli della batteria:
Peso: 2.400 gr.
Altezza max: 100 mm
Altezza: 94 mm
Lunghezza: 150 mm
Profondità: 65 mm
Prezzo ingrosso: 7,57 Euro

Poi l'attenzione si è spostata sulla scelta della scatola esterna. Vista la natura del segnale Bluetooth sono state escluse tutte le scatole di materiale metallico, preferendo le scatole in plastica. I test realizzati hanno dimostrato che una lastra di ferro di 1 mm riduce il raggio d'azione del nostro dispositivo a un metro circa, con riduzione della potenza del segnale di circa l'80%, al contrario i test realizzati con una scatola Gewiss di plastica hanno registrato una riduzione della potenza

A	B	C	D	E	F	N	O	R
240	190	90	254	200	98	194	140	45

Tabella 8.2: Dettaglio dimensioni della scatola

del segnale inferiore al 5% senza variazione alcuna del raggio dello Scan. La scelta è caduta su una cassetta di derivazione da parete Gewiss 44 208, componente in possesso di Telerete [25].

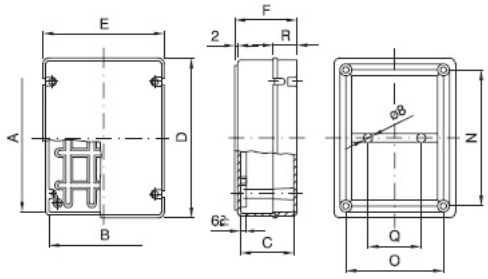

Figura 8.2: Dimensione scatola Gewiss

Dettagli della scatola

Peso: 2.100 gr.
Altezza: 98 mm
Lunghezza: 254 mm
Profondità: 200 mm
Prezzo ingrosso: 8,4 euro

Figura 8.3: Alimentatore Kert Kvik50

A differenza degli altri componenti il trasformatore non era disponibile all'interno dell'azienda, quindi dopo una breve analisi è stato acquistato il caricabatterie kvik50 della ditta italiana k.e.r.t. Oltre a essere uno dei prodotti più

economici applicano un algoritmo di carica fluttuante che permette:avere due step di carica differenti: carica iniziale profonda e carica finale di livellamento a tampone. Con questo tipo di carica si è in grado di sfruttare al massimo la capacità della batteria riducendo in modo notevole i danni causati da una carica errata. Inoltre è dotato di un led che indica lo stato di carica [26]. Per un corretto uso del caricabatterie si consiglia di effettuare il ciclo di carica con una corrente pari a circa 1/10 della corrente nominale della batteria. Una batteria da 12Vdc 24Ah, ad esempio, dovrebbe essere caricata con una corrente pari a circa 2.4A.

Dettagli del caricabatterie
Tensione d'ingresso: 230V ac
Tensione d'uscita: 13.8 V
Corrente 0.5 A
Frequenza: 50 / 60 Hz
Prezzo ingrosso: 8,5euro

8.4 Realizzazione del prototipo

Una volta reperiti i componenti si è proceduti a collocare tutti i componenti nella scatola. Per prima cosa è stata bucata la scatola per far spazio alla presa presa per il caricabatterie, al bottone d'accessione e alla presa di rete (figura 8.5). Una volta collocati questi tre componenti nella scatola sono stati inseriti l'access point e la batteria ed effettuati i collegamenti come è possibile vedere in figura 8.6.

Figura 8.4: il prototipo BlueBeauty

Figura 8.5: Dettaglio sulla presa eth, il bottone d'accessione e la presa di corrente

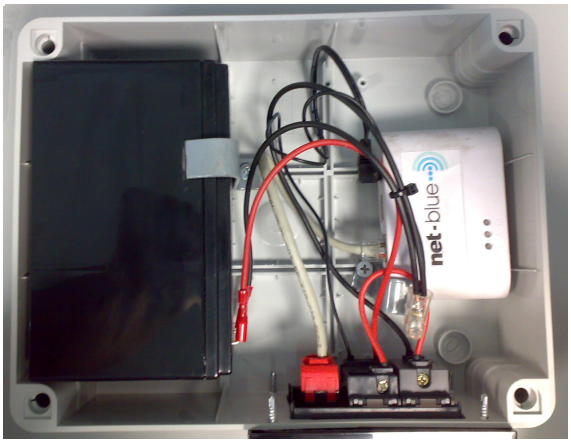

Figura 8.6: Disposizione dei componenti

8.5 Test sul campo

In occasione dello stage-it, incontro tra studenti e mondo del lavoro, Telerete ha deciso di pubblicizzarsi utilizzando proprio net-Blue. Scopo dell'iniziativa, che si ripete annualmente, è quello di agevolare l'incontro tra le imprese e gli studenti che entreranno a breve – in stage - nel mondo del lavoro con specifico riferimento al settore ICT, favorendo un'occasione di conoscenza reciproca mediante colloqui individuali. I risultati sono stati ottimi, oltre la metà degli studenti che ha effettuato un colloquio aveva ricevuto il messaggio accendendo il bluetooth incuriositi dalla cartellonistica, inoltre 15 studenti e 2 aziende si sono avvicinati allo stand per chiedere informazioni riguardo questa nuova metodologia di marketing.

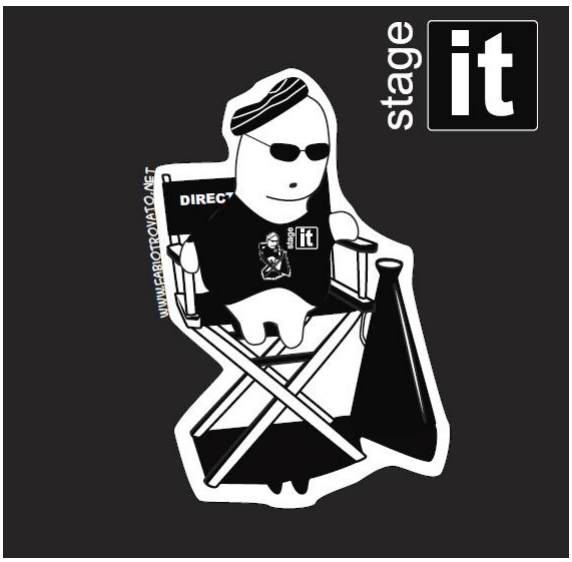

Figura 8.7: Logo della dimostrazione

Capitolo 9

SVILUPPI FUTURI E CONCLUSIONI

9.1 Sviluppi futuri

La bacheca digitale oltre a essere una conclusione del lavoro è solo una dimostrazione di ciò che è possibile realizzare. Gli scenari del proximity marketing, come detto sono motli,ma la possibilità di usare il cellulare come interfaccia a servizi avanzati ne apre di nuovi.

- Micropagamenti come autobus e parcheggi
- Ordinazioni al ristorante
- Pensiline autobus che comunichino ritardi o variazioni di tratte

Dal punto di vista tecnologico vanno approfonditi invece alcuni aspetti legati alla robustezza del sistema di comunicazione:

- Integrazione del servizio con il supporto gprs (sms, mms)
- Implementazione della sicurezza delle transazioni
- Progettazione e sviluppo di un sistema per evitare le interferenze con il wi-fi

9.2 Conclusioni

Lo scopo del mio tirocinio era quello di studiare i nuovi apparati Bluetooth di cui l'azienda si era da poco fornita. Una volta effettuata una analisi della concorrenza e capire cosa significasse parlare di *proximity marketing* sono stati realizzati dei moduli standard per la creazione di campagne di prossimità in linea con le migliori esperienze Europee. Successivamente, si è cercato di proporre qualcosa di più interessante: grazie al contatto con l'ufficio commerciale sono stati identificati i nuovi obiettivi da raggiungere: in primis la personalizzazione del messaggio.
Grazie al software development kit tool sono state create applicazioni avanzate in grado di riconoscere gli utenti in base al loro cellulare per offrire una esperienza più interattiva. Una volta raggiunto anche questo obiettivo ci siamo spinti verso quello che ho definito il *ribaltamento del paradigma di comunicazione*: ossia mettere il cliente al centro della comunicazione, rendendolo il protagonista della transizione. Installando un client direttamente nel cellulare dell'utente è possibile instaurare una connessione con l'access point per usufruire di servizi interattivi come la bacheca digitale. Fornire un servizio del genere permette a Telerete di metter piede in una fetta di mercato scarsamente servita dalle altre aziende, e di sfruttare un vantaggio competitivo non indifferente.
Proprio questo progetto mi ha dato l'opportunità di sperimentare con mano l'utilità di strumenti di controllo e di sviluppo, che hanno permesso, lavorando in team, in poco tempo di raggiungere ottimi risultati. La scelta di lavorare ad un livello intermedio è stata sicuramente vincente, se da una parte si sono presentate alcune difficoltà d'interfacciamento, dall'altra ha permesso di riutilizzare numerore features già ottimizzate.
La base di conoscienza realizzata durante il mio tirocinio è stata trasmessa grazie ai manuali tecnici, ai test effettuati, al codice scritto, alle lezioni frontali ed al wiki (conoscienza che è già attualmente in utilizzo per la realizzazione di un widget sms-bluetooth).
Questi sei mesi sono stati sicuramente intellettualmente stimolatanti, le aspettative riguardo questa mia esperienza ad Ottobre erano molto grandi, e posso dire che sono state pienamente soddisfatte.

Capitolo 10

DOCUMENTI REALIZZATI

I documenti realizzati sono i manuali e le indagini effettuate durante il mio tirocinio, non vengono presentati in questa tesi per motivi legati al segreto industriale.

- **Manuale d'introduzione**
  Destinato ad un pubblico con una preparazione tecnologica medio bassa, contiene le procedure per creare una semplice campagna di Proximity Marketing

- **Manuale Tecnico**
  Destinato all'area tecnica contiene le procedure anche per realizzare campagne avanzate e per creare applicazioni con il Software Development Kit Tool. Contiene tutti i servizi disponibili per i device, i 45 test effettuati e i codici sorgente realizzati.

- **Analisi Concorrenti**
  La ricerca viene divisa in due fasi:

  1. É stata effettuata una prima analisi sui servizi offerti dalle altre aziende nel settore per tracciare un benchmark da raggiungere e possibilmente superare
  2. Successivamente si è proceduto a contattare con un questionario i diversi fornitori per valutare la possibilità di utilizzare un Hw più adeguato alle applicazioni avanzate che si sono sviluppate.

BIBLIOGRAFIA E SITOGRAFIA

[1] http://www.alfredomartinelli.info/

[2] http://en.wikipedia.org

[3] /www.proximitymarketingreview.com

[4] BluetoothSecureSimplePairingUserInterfaceFlowWhitepaper by SING Final Versione, Published September 13, 2007

[5] http://www.ne-t.it/

[6] sito del nuovo fornitore

[7] sito del vecchio fornitore

[8] http://www.python.it/

[9] Programmazione di un access point bluetooth per i serivizi proximiy Marketing By Rinaldi Dario, Padova 17 Dicembre 2009.

[10] http://www.camera.it/parlam/leggi/decreti/

[11] http://www.garanteprivacy.it

[12] http://www.repubblica.it

[13] http://www.nikitadesign.it/

[14] http://www.vallelunga.it/index.php?id=189

[15] http://www.4ruote.it

[16] www.bluetooth.com

[17] Wifi and Bluetooth figth for bandwidth by Richard A Quinnel, EDN, August 2005

[18] http://www.edn.com

[19] ObexSender 2 User Guide, Version 3.3, September 26, 2008

[20] Access Point and Access Server User's and Developer's Guide, April 2009

[21] http://www.debian.org

[22] http://www.zaha-hadid.com/

[23] http://www.ubuntu-it.org

[24] http://www.winnerbattery.net/

[25] http://www.gewiss.com/

[26] http://www.kert.it/

[27] http://www.vitobarone.it

[28] http://www.sicetelecom.it/

[29] http://www.waymedia.it/

[30] http://www.bluesixty.nl

[31] http://proxi-ma.com/

[32] http://www.bluelook.it/

[33] http://www.bluemagnet.com/

[34] http://www.zonablu.net/

[35] htpp://www.blucellnetwork.com/

RINGRAZIAMENTI

In ogni ringraziamento che si rispetti manca sempre qualcuno, quindi probabilmente questo non farà eccezione. Questi ringraziamenti sono inseriti all'interno di questa tesi, perciò nessuno si offenda se inizio con gli addetti ai lavori.
Il grazie più grande va al professor Filira Federico. Una piacevolissima conferma, ha costruito un rapporto non tra studente e professore ma tra due professionisti, consigliandomi e criticandomi sempre in modo costruttivo, credendo nel progetto e nell'investimendo del kit di sviluppo, con un obiettivo comune; collaborare con un lui in un ambito lavorativo è stata un'esperienza formativa sotto ogni punto di vista.
Potermi cimentare nelle dinamiche di un azienda come Telerete mi ha permesso di testare le capacità acquisite in questi anni di studio, aumentando in me la consapevolezza dei miei mezzi.
Il secondo ringraziamento va a Giuseppe, più che un tutor aziendale un amico con cui condividere aspirazioni e la passione per il Giappone. Una grazie anche agli altri tecnici di Telerete per avermi fatto sentire parte di un gruppo: Giuliano per le consulenze nell'ambito della pricacy, Francesco Mandelli, Luigi per la sua consulenza sul mondo Linux (prima non lo conoscevo e ora non posso più farne a meno...), Alberto, Alessio per gli innumerevoli passaggi al Papogiò, Francesco Scaramuzza, Marco, Daniele per i racconti dei viaggi in America, Barbara, Elena e Riccardo per gli inviti a pranzo. Un grazie alle ragazze del call center che hanno saputo accogliermi e spesso consigliarmi sulle mie allergie. Infine un grazie a Karoly, Giordano e Alessandro per aver creduto in me e nel mio progetto.

Un grazie alla mia famiglia, mia madre e mio padre, che più che con le parole, con l'esempio mi hanno trasmesso dei sani valori morali, insegnandomi cosa significhi guadagnarsi qualcosa, senza barare e senza prendere scorciatoie, per avermi sostenuto in questi anni di studio e per avermi motivato trovando sempre le parole giuste, se non fosse per il loro entusiasmo non sarei arrivato a questa meta. A mio fratello, un valido esempio di come la dedizione, l'impegno e il coraggio vengano premiati. Un grazie ai miei amici, primus inter pares, a Giuseppe per avermi saputo incoraggiare, per essere stato il primo a non mollare quando tutto sembrava perso e per aver avuto il coraggio di cambiare strada quando era giusto farlo. A Luca, per aver condiviso con me i primi anni di studio e per offrirmi sempre nuove sfide, a Claudia, Quagg e Berto per le serate ludiche. Un grazie a

Edo e Zevi per una nuova avventura iniziata insieme che spero si possa protrarre il più a lungo possibile. Un grazie a Nicola, Emanuele e Serena. Un grazie ai miei colleghi universitari, a Simone, per tutto quello che abbiamo condiviso e per tutto quello che spero potremo condividere in futuro, collaborare con lui mi ha reso un ingegnere migliore. Un grazie a Carlo Alberto, Vincenzo, Max ed Elisabetta. Delle scuse invece vanno alle persone che ho trascurato per dedicarmi agli studi.

Un grazie, infine, di sentimento va ad Alessia, con cui ho vissuto cinque anni universitari. Ha saputo spronarmi perchè dessi il meglio ogni giorno, credendo in me più di chiunque altro. Non starò qui a scrivere molto, perché so quanto lei odi che si parli di lei, ma soprattutto perché quello che ho dirle spero di continuare a dimostrarlo ogni giorno con i fatti.