ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

CAMPUS DI CESENA
SCUOLA DI INGEGNERIA E ARCHITETTURA

Corso di Laurea in Ingegneria Elettronica, Informatica e delle Telecomunicazioni

WEARABLE COMPUTING E BODY AREA NETWORK: ANDROID E WRISTOX2 COME CASO DI STUDIO

Elaborata nel corso di: Sistemi Operativi

Tesi di Laurea di:
EDOARDO ALEXANDER DEWHURST

Relatore:
Prof. ALESSANDRO RICCI

ANNO ACCADEMICO 2013–2014
SESSIONE II

PAROLE CHIAVE

Wearable Computing

Body Area Network

Bluetooth

Android

WristOx2

Alla mia famiglia, alla mia ragazza e ai miei amici

Indice

Introduzione

L'informatica è in costante evoluzione. Quest'ultima va di pari passo con il cambiamento che viene imposto dalla società. In un mondo sempre più sotto il controllo di un meccanismo che rende gli uomini schiavi dell'inesorabile scorrere del tempo, il tempo diventa prezioso, quasi inestimabile, una delle pochissime variabili che la tecnologia non riesce ancora a fermare o anche semplicemente a controllare. Ci si accontenta quindi di trovare un compromesso che limiti i danni di questo inarrestabile 'nemico'. Nascono così le tecnologie wearable, che si pongono lo scopo, ancora una volta, di fornire ulteriore velocità, di rendere tutto più rapido, di minimizzare la distanza che separa uomo e computer e di favorire un'integrazione pervasiva fra la persone e i dispositivi. Lo scopo che si vuole raggiungere è quello di avere sempre a disposizione la tecnologia, di averla a portata di mano, anche in una società dinamica ed in movimento come al giorno d'oggi.

In questa tesi verranno affrontati due argomenti principali. Il primo sono le tecnologie wearable, comprendendo anche la notazione più generica di tecnologie bearable, che si stanno sempre più diffondendo negli ultimi anni; il secondo sono le BAN (Body Area Network), reti di sensori e dispositivi posti sul corpo umano, utilizzate per rendere possibile la comunicazione e l'interazione fra i device wearable. Si partirà da una trattazione di tipo generico degli argomenti, descrivendo l'architettura fisica delle tecnologie, con focalizzazione sull'aspetto informatico prevalentemente che su quello elettronico e telecomunicazionistico. Si parlerà degli attuali impieghi dei dispositivi e delle reti, e delle loro probabili evoluzioni future. Si introdurranno poi i protocolli di comunicazione principali e se ne analizzeranno le differenze, decretando se sia o meno conveniente puntare su uno o sull'altro rispetto alle esigenze di progetto. Verrà introdotto il sistema operativo Android, descrivendo la sua architettura e fornendo le informazioni basilari per comprendere al meglio il rapporto esistente con la tecnologia Bluetooth.

Verrà infine affrontato un caso reale di implementazione di una piccola body area network, sfruttando la tecnologia di Android combinata con il dispositivo WristOx2. Android viene scelto poiché versatile ed utile in questo tipo di applicazioni, combinandolo, grazie alla facilità di interazione con i device esterni del WristOx2, con quest'ultimo.

Resterà poi da capire quanto queste tecnologie evolvendosi potranno influire (in modo positivo o in modo negativo) sulla vita quotidiana degli utenti, per poi concludere riflettendo sul flebile confine esistente fra evoluzione stratificata dell'essere umano e lenta trasformazione in cyborg.

Capitolo 1

Wearable Computing

In questo capitolo verrà affrontato il discorso relativo ai dispositivi wearable, senza soffermarsi sugli aspetti tecnici di alcun dispositivo in particolare. Si comincerà con una sintetica trattazione sulla storia di questa tecnologia (che sembra essere molto più antica di quanto si pensi), passando poi alla definizione di device 'bearable', illustrando le differenze fra questi ultimi e i dispositivi wearable. Si analizzeranno poi i motivi che hanno portato alla diffusione di queste tecnologie. La sezione successiva riguarderà gli impieghi del wearable nei vari ambiti (medicina, ambito bellico, gaming, e tanti altri), per proseguire poi con una riflessione su quali dovranno essere i requisiti da rispettare perché il wearable continui ad evolversi e ad allargarsi sul mercato.

1.1 La tecnologia wearable

Per wearable computing si intende lo studio o la pratica dell'inventare, progettare, costruire o semplicemente utilizzare device computazionali o sensori in miniatura attaccati al corpo umano

I dispositivi wearable possono essere portati sotto, sopra, o anche all'interno degli abiti. C'è anche la possibilità che un abito sia esso stesso un device. La necessità di avere dei dispositivi indossabili nasce dal bisogno attuale di essere sempre e costantemente immersi nell'ambiente in cui viviamo. La tecnologia sta ormai raggiungendo la maggior parte della popolazione mondiale e l'evoluzione degli ultimi anni ha portato alla creazione

di dispositivi sempre più piccoli e facilmente trasportabili. Una volta che i cellulari sono diventati smartphone, si è provato ad effettuare un ulteriore passo per creare qualcosa che fosse non solo portatile, ma che raggiungesse quasi il livello di una vera e propria estensione del proprio corpo. Ecco allora che sono stati dapprima inventati i social network assieme ad un numero enorme di altri sistemi distribuiti pervasivi, che appaiono ormai come una parte fondamentale della vita di tutti i giorni. L'esempio più lampante di queste reti è sicuramente Facebook, che conta ormai oltre 1 miliardo e mezzo di iscritti. Ecco quindi che nasce il wearable computing, una fusione fra l'uomo e l'elettronica (o forse sarebbe meglio dire fra gli oggetti indossabili dall'uomo e l'elettronica).

1.1.1 Breve storia del wearable

Se è corretto affermare che la tecnologia wearable recente sia qualcosa di nuovo, non si può di certo dire che la ragione che ha portato a tutto questo lo sia. Volendo approfondire, l'uomo ha sempre cercato di portare con sé gli oggetti più avanzati nel relativo campo tecnologico. Come spesso ricorda Steve Mann (considerato da molti il pioniere del wearable, e chiamato spesso ironicamente 'il primo cyborg'),

Già nell'antichità, i cinesi giravano con un anello dotato di un abaco funzionante all'interno

Si può quindi affermare che, con le dovute limitazioni, la tecnologia wearable esista già dal XVII secolo. Più tardi, nel 1810, la Regina di Napoli ricevette in dono un bracciale con al suo interno un piccolo orologio da taschino. E, come quasi qualsiasi altra tecnologia esistente, anche gli orologi da polso trovarono presto utilizzo e sviluppo in ambito militare, per permettere nello specifico ad un ufficiale della marina tedesca di coordinare gli attacchi senza avere le mani occupate a controllare l'ora con l'ormai divenuto ingombrante e scomodo, orologio da taschino. Fra il 1960 e il 1970 furono prodotte numerose versioni di quello che si considera essere il dispositivo wearable tecnologico più antico della storia: una sorta di timer che permetteva di stabilire quale sarebbe stato il comportamento di una roulette. Si può parlare così del primo dispositivo progettato ad hoc, non quindi

general-purpose come quelli dell'era moderna. Negli stessi anni la Hewlett Packard produsse anche la prima calcolatrice da polso.

Negli anni '80 invece, grazie soprattutto al contributo dello stesso Mann, i sistemi divennero appunto più general-purpose. Potevano quindi essere riprogrammati dall'utente, e non essere utilizzati quindi per un unico scopo. Successivamente alcuni appassionati nel settore cominciarono ad interessarsi all'argomento wearable, e vennero prodotti altri tipi di dispositivo, come telecamere indossabili o i primi computer portatili. La Columbia University sviluppò in seguito il primo sistema basato sulla realtà aumentata, il KARMA (Knowledge-based Augmented Reality for Maintenance Assistance), un sistema che era in grado di riconoscere attraverso dei sensori posizionati su diversi dispositivi, come le stampanti, gli interventi di manutenzione necessari. Si tratta forse del primo esempio di 'Internet Of Things'. Nel 1994 venne creato il primo vero e proprio computer da polso, ancora un po' scomodo per l'utilizzo perché alquanto ingombrante, e sicuramente con nulla a che vedere con i dispositivi presenti al giorno d'oggi sul mercato. Negli anni 2000, più precisamente nel 2002, si può trovare la prima interessante applicazione della tecnologia wearable (o meglio bearable, concetto che verrà affrontato in seguito) in ambito medico. Kevin Warwick riuscì a progettare un ciondolo che cambiava colore in base agli impulsi elettrici generati dal suo sistema nervoso. Il ciondolo non veniva indossato da lui, ma da sua moglie, sfruttando quindi anche le tecnologie wireless.

Per quanto riguarda i recenti sviluppi non si può non parlare dei computer, passati da dimensioni impensabili per il trasporto a oggetto comodamente trasportabile con sé, e il telefono, che da fisso è diventato mobile e di dimensioni via via sempre più piccole. Tutto ciò nasce dall'aumento della dinamicità nella vita moderna, in cui il tempo sembra essere sempre meno e la velocità sempre in crescita.

1.1.2 I bearable device

Poco sopra l'inizio di questa sezione, si è affermato implicitamente che i computer portatili e i telefoni cellulari siano dispositivi wearable. In realtà questo succede perché si sta sviluppando sempre di più il concetto di 'intelligenza umanistica' che verrà affrontato poco più avanti. Prima però c'è da discutere un altro concetto, quello dei dispositivi bearable. Sarebbe infatti sbagliato fermarsi ai soli wearable device, perché in realtà quando si

tratta questo argomento si devono considerare anche altri dispositivi, che in questo caso prendono il nome di bearable. Questi device non sono sempre per forza indossabili, ma includono anche quelle categorie di dispositivi che possono essere posizionati sul corpo (sensori di qualsiasi tipo) o addirittura all'interno del corpo stesso. Già dal significato si può infatti intuire che ci sia una differenza piuttosto grande fra i 'wearable' ('indossabili') device e i 'bearable' ('portabili') device. La grande differenza sta nel fatto che i secondi includono i primi, ma possono essere portati con sé dallo user anche all'interno, appunto, del proprio corpo. Andando avanti nella descrizione di tali dispositivi, in ogni caso, ci si riferirà ad essi indifferentemente con il nome di wearable o bearable, rimarcando la differenza solo nel caso in cui questa sia davvero importante ai fini della comprensione.

1.1.3 Distinzione fra wearable device e dispositivi portatili in generale

La distinzione che si può fare fra i bearable o wearable device e, ad esempio, i computer portatili, è che i primi sono stati ideati per facilitare l'interazione fra l'uomo e il dispositivo, mentre i secondi restano ancora in una dimensione propria. Si vuole fare in modo di inserire l'essere umano nel loop di feedback del dispositivo. La differenza quindi fra uno smartphone ed un PC rimane (oltre alla ben più evidente differenza di grandezza) quella che fra il telefono e l'uomo vi sia un'interazione che li porta quasi ad essere un tutt'uno, mentre il PC rimane distaccato da questo scopo. Un esempio di questa distinzione risulta ovvio quando si parla di wearable device come gli smartglass. Immaginiamo di avere un software per il riconoscimento facciale installato sia sugli occhiali che su un computer portatile. Tralasciando le dimensioni dei device (per cui sicuramente l'utilizzo con il PC è più scomodo), il software funziona sia su uno che sull'altro dispositivo allo stesso modo, ma per quanto riguarda gli occhiali, questi percepiranno la scena davanti a loro esattamente come la percepisce chi li indossa, mentre ciò non accade sicuramente con il PC. Ecco perché spesso questi dispositivi vengono definiti come un tentativo di rendere embedded l'intelligenza umanistica (Humanistic Intelligence). Fatta questa definizione ora è più chiaro lo scopo di device impiantati anche all'interno del corpo umano (e a questo punto non si può solo parlare di wearable device, ma il concetto deve per forza essere esteso al termine di bearable device).

1.1.4 Perché i wearable

Un'importante funzione offerta da questo tipo di dispositivi è il multitasking già integrato. Lo user non deve infatti avviare software o programmi, visto che per quanto riguarda il wearable computing, questi sono costantemente in esecuzione in background. Spesso nell'interazione che avviene fra l'uomo ed il computer (Human-Computer Interaction, HCI), si concepiscono le due entità come entità singole che interagiscono fra di loro. L'obiettivo della teoria dell'intelligenza umanistica (Humanistic Intelligence) cerca di rendere uomo e macchina un tutt'uno al fine di avere una sola entità, con dei suoi input e dei suoi output. É come se il computer fosse in questo caso un'estensione dell'uomo, un suo secondo cervello. La necessità di avere questi dispositivi è sicuramente (come già accennato in precedenza) dovuta al fatto che si vuole poter fare sempre di più in sempre meno tempo. Per fare ciò è necessario certamente aumentare le prestazioni dei vari device ma anche avere la possibilità di effettuare più azioni contemporaneamente. La tecnologia wearable permette tutto ciò, grazie al fatto di sfruttare funzioni come il controllo vocale per l'interazione. Il tutto si riduce quindi ad una mera questione di tipo meccanico: l'utente ha due mani libere, tornando a poter effettuare una nuova azione mentre può ad esempio visualizzare un input visivo attraverso degli occhiali.

1.1.5 Situatedness

I sistemi computazionali sono stati inventati per svolgere computazioni complicate per le quali serviva l'aiuto di una macchina. Evolvendosi nel tempo, questi sistemi hanno poi avuto la necessità di comunicare fra di loro, di interagire, inizialmente per svolgere compiti più onerosi e successivamente per poter svolgere funzioni diverse ma dipendenti l'una dall'altra, facendo nascere così il bisogno di definire standard e protocolli di comunicazione. L'ultima evoluzione riguarda il rapporto sempre più necessario che un sistema deve avere con l'ambiente che lo circonda. Se un sistema è in qualche modo consapevole di essere inserito in un ambiente, e se è in grado di interagire con esso, si parla di sistema 'situated'. è alquanto palese che tutti i dispositivi wearable o quasi, inclusi i software al loro interno, debbano godere della proprietà di situatedness.

1.2 Applicazioni

Le applicazioni dei wearable device sono le più varie. Volendo rimanere su un aspetto alquanto generico, si potrebbe dire che qualsiasi cosa portata con sé dallo user sia un dispositivo, senza per forza che esso sia inserito all'interno del suo corpo o fisicamente attaccato ad un suo indumento. Entrano quindi di diritto nella categoria tutti i telefoni e gli smartphone. Nasce perciò (ultimamente) l'idea che qualsiasi 'estensione' elettronica del corpo umano possa essere considerata un dispositivo wearable. Andiamo ora ad analizzare le applicazioni più comuni per questo tipo di tecnologia.

1.2.1 Realtà aumentata e diminuita

Aumentare la realtà significa imporre un livello in più alla realtà del mondo, spesso con immagini o tecnologie di pattern recognition. Può essere utilizzata in diversi contesti, e creata attraverso innumerevoli dispositivi. Il più utilizzato è proprio lo smartphone. L'utilità di questa applicazione è spesso quella di facilitare la visione che si ha del mondo, semplificando ciò che si vede o focalizzando l'attenzione su un aspetto particolare dell'ambiente circostante. Altri utilizzi di questo tipo di tecnologia si riscontrano nei videogiochi, e, ovviamente, in questo caso, lo scopo dell'utilizzo della realtà aumentata risulta semplicemente quello di trarne un divertimento.

Non sempre però quello che si vuole è focalizzare l'attenzione su un dettaglio in particolare, o aggiungere nuovi livelli visivi a ciò che si vede. Ecco allora che nasce la realtà diminuita, utilizzata per rendere l'interazione con l'ambiente più semplice. Persone con problemi alla vista possono quindi osservare il mondo in modo semplificato trascurando i dettagli non fondamentali. Per quanto riguarda questo tipo di tecnologia, come si è accennato in precedenza, il dispositivo più utilizzato rimane ancora lo smartphone, anche se come si può intuire, il trend predominante per questo tipo di 'vista' troverà sicuramente molto spazio negli smartglass (i quali verranno trattati più avanti).

1.2.2 Smartphone

Attualmente il dispositivo wearable (se così lo si può considerare) più diffuso è lo smartphone. Al giorno d'oggi gli smartphone sono estremamen-

Figura 1.1: Esempio di realtà aumentata

te diffusi in tutto il mondo e aiutano i loro possessori, per quanto riguarda la proprietà di essere situated ad esempio, ad orientarsi tramite il GPS o con altre funzioni. In futuro gli studi porteranno a smartphone completamente immersi nell'ambiente, grazie ai quali lo user potrà essere avvertito, riconoscendo autonomamente il luogo in cui si trova, di potenziali pericoli, e fornendo anche una funzionalità simile alla 'scatola nera' presente sugli aeroplani. In questo modo sarà possibile ricostruire, se mai ce ne fosse bisogno, la vita e gli spostamenti d qualsiasi possessore di uno di questi dispositivi. Volendo considerare lo smartphone come wearable e il PC invece no, questo rimane attualmente il dispositivo con la più grande potenza computazionale. A differenza infatti di dispositivi come sensori o device di altro tipo, uno smartphone può contare su delle prestazioni molto elevate, che lo fanno diventare fondamentale per la creazione di applicazioni molto onerose. Come si vedrà in seguito infatti, uno smartphone è un ottimo dispositivo da posizionare al centro di una rete di device e sensori. Sarà anche il punto di riferimento con cui dovrà essere possibile l'interazione, e rimane in ogni caso il dispositivo wearable più diffuso attualmente.

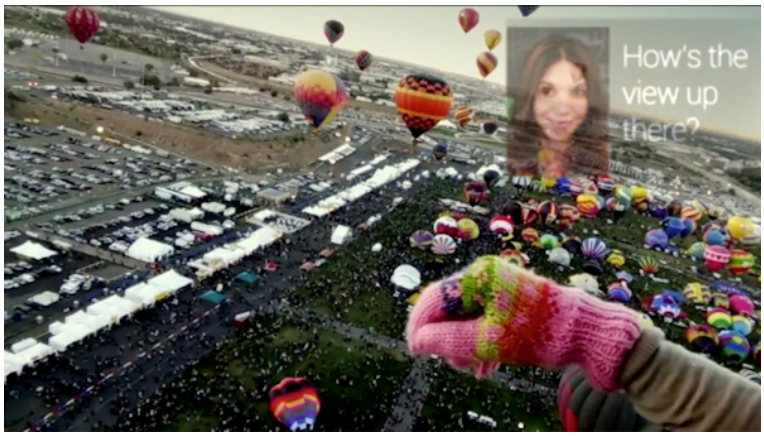


Figura 1.2: Vista con i Google Glass

1.2.3 Smartglass

Il mercato ha da poco visto arrivare un altro tipo di dispositivi: i 'glass'. Si tratta di occhiali 'intelligenti' e sempre più simili ad un normale paio di occhiali da vista che permettono all'utente di ottenere delle informazioni in modo 'hands-free' (ovvero senza che le mani siano occupate). Per raggiungere questo scopo l'interazione con il dispositivo è per forza di cose fondamentalmente di tipo vocale. Gli occhiali più famosi a livello mondiale sono sicuramente quelli prodotti da Google, i 'Google Glass', che sono stati ideati per cercare di distribuire sul mercato una sorta di dispositivo di computing ubiquo. Questi ultimi, che secondo la stessa Google rivoluzioneranno la vita degli utenti attraverso il loro concetto di 'hands-free' già citato in precedenza, prevedono una visione di una porzione di spazio (in alto a destra in ciò che gli occhi osservano) che permetta attraverso comandi vocali di effettuare, contemporaneamente alla visualizzazione di un contenuto, una qualsiasi azione con l'aiuto di entrambe le mani. Un esempio di visione attraverso i Google Glass è disponibile in figura 1.2 (l'opacità della porzione dello schermo può essere modificata). La loro diffusione non è ancora alta, e ciò è dovuto al fatto che questi dispositivi siano ancora molto costosi e non accessibili quindi ad un ampia fetta di mercato (sono attualmente in vendita al costo di 1500 dollari). L'idea che si ha di smartglass è però spesso legata solo al concetto di 'occhiali-computer' più general purpose.

Esistono invece numerosi tipi di occhiali che svolgono funzioni diverse, anche se tutte hanno a che fare con la vista. Esistono occhiali che propongono funzioni di realtà aumentata, per questioni legate alla sicurezza o semplicemente al divertimento, ed occhiali che permettono di collegarsi al mondo circostante, entrando a far parte dell'Internet of Things, un mondo in cui qualsiasi cosa (o quasi) fornisce un output e riceve un input, facendo sì che tutto sia un unica rete nella quale tutto interagisce con tutto.

1.2.4 Smartwatch

Un discorso a parte meritano gli smartwatch. I cosiddetti 'orologi intelligenti' sono visti dalla maggior parte dei consumatori come una vera e propria innovazione per quanto riguarda i dispositivi wearable. Si potrebbe anche pensare che ciò sia vero, prendendo le dovute distanze. Già nel 1985 infatti, era sul mercato il Seiko Epson RC-20, una specie di agenda/-calcolatrice da polso. Ovviamente questo tipo di orologio non aveva nulla a che fare con gli ultimi iWatch o Samsung Gear, ma già all'epoca veniva reputato un oggetto alquanto inutile. La stessa cosa avviene attualmente quando si parla di smartwatch. Mentre ad esempio gli smartglass offrono delle funzionalità ridotte rispetto a quelle offerte da un cellulare (fotocamera con una definizione più bassa, realtà aumentata allo stesso modo col quale questo potrebbe essere sfruttato su uno smartphone) ma con il vantaggio di liberare le mani dello user, ciò non si può dire per un orologio.

Uno smartwatch ha la sola possibilità di avere una sorta di dispositivo mobile con uno schermo più ridotto rispetto a quello di un cellulare, e attaccato al polso. Quindi, riflettendo, uno smartwatch non è altro (dal punto di vista meccanico) che un piccolo cellulare consultabile allo stesso modo nel quale si controlla l'ora su un normale orologio da polso. Avendo già a disposizione uno smartphone quindi, un cliente raramente opta per l'acquisto anche di uno smartwatch, mantenendo la diffusione di questi dispositivi (almeno nel loro stato attuale) piuttosto bassa. Con ciò non si vuole escludere che in futuro, con opportune evoluzioni, questo prodotto non possa ingrandire la sua fetta di mercato.

Figura 1.3: Esempio di smartwatch

1.2.5 Ambito Medico

Nel settore della medicina i dispositivi wearable sono spesso associati ai soli sensori posizionati sul corpo del paziente. In realtà esistono altri tipi di dispositivi che invece di fornire dei dati, attuano delle vere e proprie azioni. É questo il caso degli attuatori di ultima generazione, in grado di somministrare al paziente determinati quantitativi di una qualsiasi sostanza con controllo remoto. Un altro esempio di applicazione in questo campo sono gli avanzatissimi sistemi di EEG (ElectroEncephaloGram) basati sui cosiddetti 'thinking cap'. Quest'ultima tecnologia permette a chi la utilizza di collegare i vari dispositivi a determinate aree del cervello, e non è lontano un futuro in cui una persona non vedente possa utilizzare questa tecnologia come un vero e proprio occhio. Le applicazioni più diffuse rimangono comunque i sensori classici per il controllo dei parametri vitali. Il discorso riguardante l'ambito medico verrà ampliato nel capitolo successivo quando si parlerà di BAN, e nell'ultimo capitolo, quello relativo al caso di studio, si affronterà la creazione proprio di una piccola rete di controllo dei parametri vitali, sfruttando il WristOx2, un dispositivo wearable, appunto, che concerne il settore medico.

Figura 1.4: Combattente di terra e combattente aereo

1.2.6 Ambito Militare

Anche in ambito militare il wearable computing è in netta crescita. Sono sempre di più infatti gli abiti 'intelligenti' che permettono di svolgere diverse funzioni, dalla coordinazione degli addestramenti alla pianificazione di vere e proprie strategie sul campo. Per quanto riguarda i progetti in questo settore, il più interessante e quello che promette gli sviluppi più rapidi è il progetto dell'esercito degli Stati Uniti d'America FCS (Future Combat Systems). In particolare è interessante affrontare il caso della dimostrazione tecnologica del Future Force Warrior, un prototipo di guerriero del futuro. In realtà, se il nome può far pensare ad un cyborg, si tratta di un progetto che sfrutta le tecnologie wearable per offrire numerose possibilità di cooperazione e di interoperabilità ai soldati. Si tratta di un progetto che sfrutta tutte le ultime scoperte riguardanti il campo delle tecnologie wearable e le trasferisce sul corpo di combattenti terrestri o aerei. Lo scopo del progetto è quello di aumentare la forza delle truppe attraverso l'utilizzo della nanotecnologia e della meccanica degli esoscheletri. Rispetto a quello che si è visto in precedenza sono presenti dispositivi situated per la vista, impianti di riscaldamento o raffreddamento wearable, e un generatore indossabile alimentato da una micro-turbina alimentata da un carburante liquido.

1.3 Futuro della tecnologia

Come ogni tecnologia recente, anche quella dei wearable non ha ancora raggiunto l'apice dal punto di vista commerciale. Gli utenti si mostrano ancora scettici verso questo tipo di dispositivi, chi per motivi di privacy, chi per motivi di costo e relativo rapporto utilità/prezzo.

1.3.1 Requisiti per la diffusione futura

I principali requisiti che i dispositivi dovranno rispettare per fare in modo che la fetta di mercato occupata dal wearable si allarghi sono i seguenti:

- L'effettivo valore: Questo tipo di device si diffonderà se porterà un sostanziale miglioramento nella vita degli utenti e se sarà in grado di offrire servizi innovativi. Come accennato prima nel discorso relativo agli smartwatch, spesso si tende ad esagerare nel creare dispositivi che non offrono in realtà alcuna funzione veramente nuova. L'unica vera novità sta nella forma nella quale viene progettato il device. Questo significa che per fare in modo che in futuro il mercato wearable si allarghi, si dovranno progettare dispositivi che oltre a garantire un'evoluzione dal punto di vista estetico, ne garantiscano anche uno, più che dal punto di vista dell'utilità, rispetto al quale funzionalità già esistenti possano essere sfruttate in un modo più congeniale e conveniente per l'utente.

- La sicurezza: Questo secondo requisito riguarda soprattutto quei dispositivi che conterranno informazioni di assoluta riservatezza, come accadrà sicuramente in ambito medico e militare. Se la sicurezza dei dati contenuti nei dispositivi (e delle reti, discorso che verrà affrontato nel capitolo successivo) non sarà garantita, l'utente non sarà portato ad acquistare queste nuove tecnologie, onde evitare di mettere in pericolo le proprie informazioni private.

- La privacy: Questo punto è simile al precedente, ma tratta soprattutto l'aspetto legato all'autenticazione dell'utente in grado di utilizzare il dispositivo. Non si tratta quindi di vera e propria sicurezza delle reti, ma si focalizza più sull'impedire a chi non è il vero proprietario di accedere al device. I dispositivi, infatti, spesso saranno personali, e

l'ultima cosa che si vuole è che l'accesso sia semplice ed ottenibile da chiunque.

- La compatibilità/interoperabilità dei dispositivi: Uno dei requisiti fondamentali per la diffusione è sicuramente questo. In un mondo ormai popolato da smartphone e computer portatili, il wearable deve essere (e solitamente è) visto come un'estensione della tecnologia esistente, e non come una vera e propria tecnologia nuova. Per rendere possibile tutto ciò, i dispositivi devono essere assolutamente in grado di cooperare fra di loro, o di avere quantomeno la possibilità di interagire. Per fare un esempio, è molto difficile che un sensore per il battito cardiaco (ad esempio inserito in una maglia) abbia anche un display sul quale visualizzare i dati. Avere a disposizione il semplice dato senza poterlo visualizzare sarebbe del tutto inutile. Nasce quindi la necessità di fare interagire il sensore con un altro dispositivo, che potrebbe essere uno smartphone, o per rimanere in senso stretto legati al concetto di wearable, un paio di smartglass.

- La facilità di utilizzo: Questo requisito è un requisito sicuramente più dal punto di vista commerciale che dal quello dell'evoluzione nel vero senso della parola. Per ottenere infatti una diffusione per quanto riguarda il numero di dispositivi, c'è bisogno che l'utente (in questo caso visto come consumatore) compri il maggior numero di device possibile. Per ottenere questo risultato, i dispositivi non devono essere utilizzabili solo da un'élite di persone, ma essere di facile consultazione ed utilizzo anche per il consumatore comune, che spesso non ha alcuna base informatica o elettronica.

1.4 Conclusione

Concludendo, la tecnologia wearable ha sicuramente le porte spalancate per evolversi e diventare sempre di più di uso comune, a patto che non cada in errori che potrebbero portarla ad essere inutile ed obsoleta ancor prima di essere inventata e diffusa. Se sfruttata a dovere e portata avanti seguendo i requisiti fondamentali, può rappresentare il vero futuro della tecnologia. Rimane ora da capire se possa o meno sostituire le ricerche ed i risultati ottenuti dalla robotica, che nasce già su un corpo elettronico, ed ottiene

così il vantaggio di avere già in partenza un'interoperabilità fra il 'corpo' e i dispositivi aggiuntivi. Va anche detto che non si potrà mai sostituire il corpo umano quantomeno in ambito medico, dove sicuramente la tecnologia wearable non ha concorrenza. Nel capitolo successivo verranno analizzate le reti che collegano tali dispositivi wearable nel corpo, e rendono possibile poi la raccolta delle informazioni ottenute, o l'attuazione di alcune azioni.

Capitolo 2

Body Area Network

Dopo aver terminato il discorso relativo ai device wearable, passiamo ora ad affrontare il problema successivo. Volendo allargare il concetto di wearable/bearable device a tutto ciò collegato in qualche modo al corpo umano, capace di riceve un input e/o generare un output, resta ora da capire come questi dispositivi possano essere collegati fra di loro. Il problema nasce dal fatto che i dati 'grezzi' sono pressoché inutili, se ad esempio non esiste un modo per visualizzarli. Ecco che allora vengono sviluppate delle reti che mettono in comunicazione tutti i sensori, sebbene in letteratura spesso per body area network, si intenda l'insieme dei sensori, dei device e della rete che li unisce. In questo capitolo verranno affrontate le differenze fra una rete di sensori classica ed una BAN, soffermandosi sui vantaggi apportati da queste ultime e analizzando i motivi della loro diffusione, passando poi ad analizzare l'architettura di tali reti, dopo aver distinto quelle 'wired' (con i fili) da quelle wireless. Sebbene esista una differenza fra BSN e BAN (visto che con la prima si intendono reti esclusivamente composte da sensori, e con la seconda reti composta da qualsiasi tipo di dispositivo), come nel capitolo precedente i due termini possono essere considerati quasi sinonimi, anche se si utilizzerà preferibilmente il termine BAN (in modo da includere anche i device wearable). Se esisterà un valido motivo per specificare un tipo di rete piuttosto che l'altro, questo verrà argomentato.

2.1 Le BAN

Le BAN (Body Area Network) o Body Sensor Network sono delle reti di sensori e device indossabili, con la particolarità di essere installate sugli esseri umani. Le caratteristiche principali di queste reti sono il poter elaborare i dati praticamente in real-time, il fatto che tutti i sensori abbiano un'elevata durata della batteria, quindi un funzionamento nel tempo prolungato, un basso consumo energetico (ed è questo il motivo per cui la loro vita aumenta di durata), la facilità con la quale i vari sensori possono essere sostituiti, poiché sono immediatamente raggiungibili, cosa che in alcuni casi non è possibile per quello che riguarda le reti di sensori generiche (le sensor network).

2.1.1 Differenze fra reti di sensori e BAN

Volendo analizzare nel dettaglio le differenze esistenti fra una normale rete di sensori ed una BAN si notano queste differenze fondamentali:

- La distribuzione e la densità dei sensori: Spesso i sensori in una BAN sono collocati in punti precisi del corpo umano per poterne misurare, ad esempio, i parametri vitali (quali il battito cardiaco, o la saturazione dell'emoglobina ossigenata nel sangue), mentre nelle reti di sensori generiche i sensori sono piazzati senza tener troppo conto di una posizione precisa. Inoltre, per quanto riguarda la quantità di sensori, si può affermare che le BAN ne abbiano un numero sensibilmente ridotto. Questi due fattori fanno sì che, essendo i sensori in numero minore e facilmente raggiungibili, i componenti del sistema possano essere riparati o sostituiti quando ce n'è bisogno. Questo deriva dal soggetto che bisogna analizzare. Una rete di sensori è infatti il metodo più rapido per monitorare ambienti in cui si presentano condizioni critiche, come luoghi in cui si è verificata una calamità naturale, o magari un disastro aereo. Spesso i sensori per monitorare la situazione in questo tipo di avvenimenti, sono posizionati senza tener troppo conto della necessità (in alcuni casi molto rara) del possibile riposizionamento dei sensori una volta distribuiti.

- Data-rate: Il data-rate di una rete di sensori qualunque spesso si basa sugli eventi, ovvero vengono inviati dei dati solo se questi sono impor-

tanti per quanto riguarda un determinato avvenimento (ad esempio, per una rete di sensori che rileva il movimento di una certa persona in un dato spazio, l'evento potrebbe essere generato nel momento stesso in cui il soggetto da monitorare entra effettivamente in questo spazio). Per quanto riguarda invece le BAN, i dati vengono forniti quasi sempre ad intervalli regolari (trattandosi di un monitoring molto spesso delle funzioni vitali). Si incontra quindi un data-rate che nella maggior parte dei casi è molto maggiore rispetto a quello di una normale SN. C'è anche da specificare che, se è vero che il data-rate aumenta per un discorso di pacchetti inviati nell'unità di tempo, è altrettanto vero che spesso le BAN sono di piccole dimensioni, e i dati che è necessario trattare sono di dimensioni molto ridotte rispetto a quelli che sono trattati da sensor network di dimensioni molto più ampie.

- Durata della vita del sensore: La durata media della vita di un componente di una BAN non dura più o meno di quanto non possa durare un componente qualunque di una rete di sensori. La vera differenza sta (come già accennato in precedenza) nel fatto che le batterie possano essere facilmente sostituite poiché i sensori sono raggiungibili anche dopo il deployment della BAN. Questo fa sì che la vita di un componente si allunghi, ma ciò non è una caratteristica intrinseca del sensore, ma deriva dalle caratteristiche topologiche della specifica rete.

- Mobilità: Spesso (quasi sempre) le BAN sono installate su qualcuno che può muoversi. Questo fa sì che per questo tipo di reti il movimento sia un concetto fondamentale. Il sistema, o quantomeno chi raccoglie i dati inviati dai sensori, deve essere consapevole di questo movimento per agire di conseguenza. I dispositivi facenti parte delle BAN devono quindi condividere lo stesso pattern di mobilità, al fine di poter essere coordinati e tutti consapevoli del movimento del sistema che devono andare a monitorare e sul quale sono posizionati.

2.1.2 Wireless Body Area Network

Un'evoluzione delle BAN (intese come wired BAN), sono le WBAN (Wireless Body Area Network). A differenza della BAN wired, come il nome suggerisce, l'interazione fra i sensori e il componente che deve raccogliere

i dati è di tipo wireless. Ci sono numerosi vantaggi nello sfruttamento di questa tecnologia, tra cui i principali sono:

- Minore invasività dei sensori: Ovviamente sensori che non utilizzano fili ma sono wireless sono meno invasivi e meno ingombranti per un essere umano che li indossa. Questo trova una grande utilità soprattutto in BAN che vengono sviluppate nel settore del fitness o in qualsiasi altro campo nel quale il movimento sia una prerogativa fondamentale. La non dipendenza dai cavi permette anche di bypassare le varie configurazioni fisiche di collegamento, rendendo il sistema più aperto ed estendibile.

- Possibilità di comunicare con altri dispositivi: Ritornando sul punto appena discusso, sarebbe scomodo utilizzare uno smartphone come raccoglitore di dati potendo collegare un solo dispositivo cablato ad esso. Ecco perchè utilizzando diversi protocolli di comunicazione i dati raccolti dai sensori possono essere inviati a qualsiasi tipo di dispositivo in grado di elaborare poi i dati ricevuti. Rimane comunque vero che la trasmissione attraverso una rete cablata è più rapida di una attraverso comunicazioni wireless, e questo fa sì che ancora in qualche caso la rete cablata venga preferita a quella senza fili.

- Abbassamento dei costi: Potendo interfacciare vari sensori con più dispositivi, si abbassa il costo di produzione, diminuendo una programmazione ad hoc per focalizzarsi su una strada più general-purpose, che possa soddisfare più richiesta ad un costo meno elevato, data la produzione in massa di dispositivi e sensori.

2.2 Architettura

Le BAN hanno una loro particolare architettura, e come sempre prima del progetto è necessario porsi l'interrogativo su quale sia la soluzione migliore da adottare rispetto al problema affrontato. In questa sezione verrà affrontato il discorso relativo prima all'architettura fisica generica di una rete, e successivamente verranno illustrati i vari tipi di comunicazioni interni ed esterni alle BAN. Il tutto senza soffermarsi particolarmente su aspetti legati alle telecomunicazioni, che in questo elaborato non sono trattati.

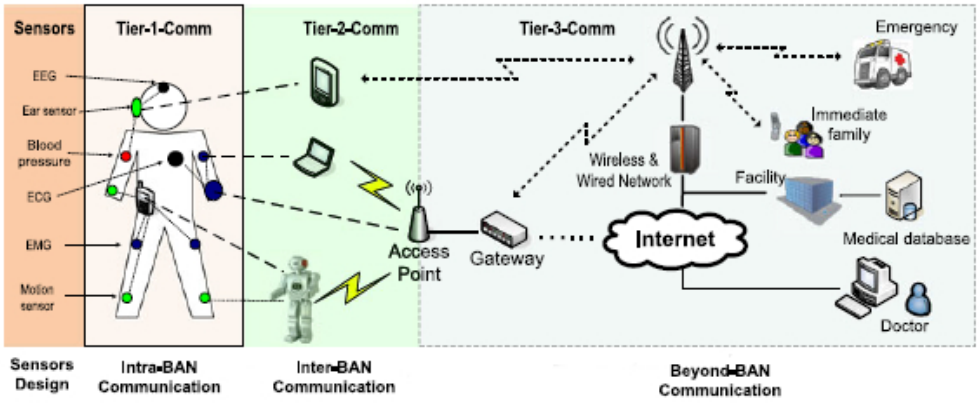


Figura 2.1: Architettura di una health-care BAN

2.2.1 Architettura fisica

Come si può notare in figura 2.1, l'architettura di una body area network si divide fondamentalmente in tre livelli, tutti basati su quali sono le comunicazioni che si svolgono al loro interno. Per quanto riguarda la BAN posizionata fisicamente sul corpo, sono presenti vari tipi di sensori e dispositivi wearable che comunicano fra di loro attraverso comunicazioni cosiddette intra-BAN, ovvero interne alle body area network. Al livello successivo troviamo le comunicazioni inter-BAN, ovvero quelle che avvengono fra varie body area network, distribuite in diversi ambienti di monitoraggio. Il terzo ed ultimo livello, al quale si farà soltanto un accenno, è costituito dalle comunicazioni beyond-BAN, che coprono tutta la trattazione dei dati, inviati poi per esempio ad un server, e pronti per essere elaborati per lo scopo specifico per il quale sono stati raccolti. Per quanto riguarda i dispositivi fisici non ci sono particolari limitazioni a quelle che possono essere le possibilità di utilizzo. Nella maggior parte dei casi, comunque, esiste un PD (Personal Device) o PS (Personal Server) applicato sul corpo dell'utente, che raccoglie i dati inviatigli dai sensori. Anche per quanto riguarda il tipo dei dispositivi fisici che possono fare parte di una BAN non esistono particolari restrizioni, se non che debbano essere in grado di comunicare in qualche modo con la rete, e che possano utilizzare il protocollo di comunicazione specifico della BAN. Quando si parla di Body Area Network è bene

specificare che, rispetto ad una normale Body Sensor Network, questa abbia intrinsecamente nel nome la possibilità di interfacciare oltre che tutti i tipi di sensore, anche innumerevoli dispositivi wearable nel vero senso della parola, come smartphone o smartglass. Spesso gli smartphone sono utilizzati però, più che come veri e propri 'rilevatori' di dati, come 'raccoglitori', utili poi all'elaborazione e all'invio degli stessi.

2.2.2 I sensori

Come già accennato in precedenza i componenti fondamentali delle BSN (viene specificato qui il tipo di rete BSN, poichè si andranno a trattare in particolar modo i sensori, più che i device in generale) sono i sensori e gli attuatori. Per quanto riguarda la loro progettazione bisogna ovviamente tenere conto dell'uso futuro che se ne farà: dovranno essere posizionati sul (o addirittura all'interno del) corpo umano. Per questo motivo, la ricerca dei materiali più adatti allo scopo è costantemente attiva. Se prima dell'avvento delle BSN la misurazione dei dati di un paziente necessitava della presenza fisica di un medico nelle vicinanze, ora non è più così. La raccolta dei dati avviene ora in un modo non più invasivo come in precedenza. I dispositivi che fanno parte delle reti diventano sempre più piccoli e più semplicemente indossabili dal paziente, e questo rende le BAN potenzialmente distribuibili in un ambiente sempre più dinamico e in modo sempre più pervasivo. I sensori sono i più vari, dai classici misuratori del battito cardiaco fino addirittura a siringhe automatiche, capaci di iniettare un determinato medicinale con controllo remoto. Sarebbe in qualche modo errato parlare di semplici sensori quando si parla di Body Area Network. Come già accennato in precedenza infatti, questo tipo di reti possono includere anche tutti i tipi di dispositivi wearable oltre che i normali sensori. É opportuno anche definire la differenza fra sensori e attuatori, poiché i primi raccolgono dati, mentre i secondi compiono azioni (un classico di esempio di sensore potrebbe essere un sensore di umidità, mentre un attuatore potrebbe essere la siringa con comando remoto che inietta nel paziente una determinata sostanza).

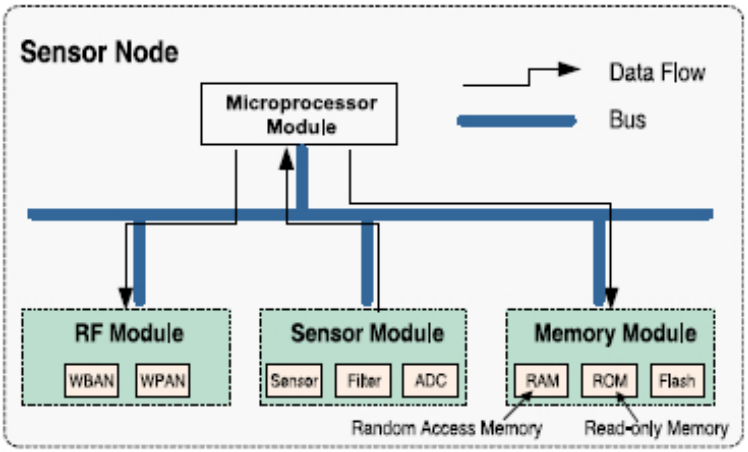


Figura 2.2: Architettura di un nodo sensore

Architettura di un nodo sensore

Sul corpo dell'utente sono posizionati quindi diversi 'nodi sensore', dei quali, a differenza dei dispositivi wearable che non hanno un'architettura comune l'uno con l'altro, è possibile definire uno schema generale. Come si può vedere in figura 2.2, un nodo sensore è composto da un microprocessore per la computazione, un modulo sensore che serve a raccogliere i dati, e a convertirli in digitale attraverso un convertitore, in modo che siano utilizzabili da una macchina, da un modulo radio utile per le comunicazioni ed un modulo della memoria, atto a mantenere le informazioni che ritiene utile conservare all'interno del sensore.

Tipi di sensore

I sensori si dividono in tre gruppi fondamentali:

- Sensori fisiologici: Misurano tutti i valori fisiologici personali, quindi il battito cardiaco, la pressione sanguigna, o possono anche fornire un'elettroencefalografia (EEG)
- Sensori biocinetici: Sono quei sensori che misurano il movimento, e fanno si che il sistema sia space-aware (consapevole dello spazio in cui si trova e nel quale si muove)

- Sensori d'ambiente: Misurano i vari valori dell'ambiente, come la temperatura o l'umidità

Per quanto riguarda le BAN i più diffusi sono sicuramente quelli del primo tipo, seguiti da quelli biocinetici, e raramente si possono trovare sensori d'ambiente all'interno di una rete di questo tipo.

Le principali differenze esistenti fra i sensori di una BAN e i sensori di una qualsiasi rete generica sono che i sensori della rete generica sono spesso omogenei, si distribuiscono in uno spazio molto esteso e gli errori sono abbastanza tollerabili (esistono più sensori che misurano un determinato fenomeno, e la caduta di un nodo specifico non è quindi sempre critica), mentre per quanto riguarda le BAN, i sensori sono posti sul corpo che si deve monitorare, ciò significa che la loro distribuzione è molto più raccolta, i sensori sono eterogenei e soprattutto la posizione di un sensore influisce parecchio su quale sarà poi la misura di un determinato valore. I sensori nella maggior parte dei casi possiedono quindi un package adeguato che ne permetta l'utilizzo anche in constesti in cui l'utente debba muoversi e potrebbe correre il rischio di spostare i sensori che ha sul corpo. Esistono inoltre numerosi modelli di sensore, ognuno con uno scopo diverso. In questa sezione verranno illustrati i principali, fornendo esempi di utilizzo.

- Accelerometro/Giroscopio: L'accelerometro è utilizzato per monitorare la posizione di un utente (in piedi, in ginocchio, seduto, sdraiato, ecc...) e riconoscerla sfruttandola per diversi scopi. I campi in cui questo sensore vede il maggior utilizzo sono la realtà virtuale, l'ambito medico, lo sport e il gaming. Il monitoraggio della postura nelle BAN solitamente si basa su accelerometri '3-axis' (sui tre assi). Il giroscopio viene invece utilizzato per stabilire l'orientazione del dispositivo per il quale è stato progettato. Si basa sulla legge fisica della conservazione del momento angolare ed è possibile trovarlo ormai in tutti gli smartphone.

- Sensore ECG (ElectroCardioGram): Il sensore ECG serve a misurare, con l'aiuto possibilmente di un grafico, il battito cardiaco. In realtà non è propriamente corretto definirlo come un unico sensore, dato che la maggior parte delle volte si tratta di più sensori che vengono piazzati in punti specifici del corpo umano in modo da misurare le differenze di potenziale. Vengono utilizzati nella quasi totalità delle BAN relative all'health-care.

- Sensore EEG (ElectroEncephaloGram): Misura l'attività elettrica nel cervello, tramite il posizionamento di vari dispositivi sulla superficie del cranio dell'utente. Nuove tecniche di progettazione di questo tipo di sensore (più orientate verso gli attuatori) permetteranno in futuro di controllare letteralmente l'attività del cervello di chi ne avesse bisogno. Una funzione molto interessante potrebbe essere quella di restituire la vista alle persone non vedenti con un mix di telecamere e attuatori posti all'interno della testa del paziente. Data l'invasività e delicatezza dell'operazione questo tipo di sensore è quello che più di altri ha bisogno di studi per quanto riguarda segnali di trasmissione dei dati e materiali idonei.

- Pulsiossimetro: Misura la saturazione del livello di emoglobina ossigenata nel sangue in modo non invasivo. Il dispositivo viene agganciato a un dito della mano, all'orecchio o a un dito del piede. Si basa sul passaggio di un fascio di luce da una parte all'altra del sensore. La misurazione viene effettuata tenendo conto della saturazione dell'emoglobina ossigenata in rapporto all'emoglobina totale nel sangue. Anche questo tipo di sensore è utilizzato spesso in applicazioni relative all'health-care e alla medicina.

- Sensori di umidità e temperatura: Sono sensori che, come si può facilmente intuire dal nome, misurano il livello di umidità e la temperatura dell'ambiente in cui si trovano. Raramente sono progettati come sensori indipendenti, ma sono ormai contenuti nella maggior parte dei dispositivi, come gli smartphone.

2.2.3 Comunicazioni

Come già anticipato in precedenza, la vera distinzione dei livelli di una BAN (intesa ora non solo come insieme di dispositivi posizionati sul corpo ma proprio come insieme totale di dispositivi e rete) avviene quando si parla di comunicazioni. Le comunicazioni si dividono in tre macro-livelli: il primo livello riguarda la cosiddetta comunicazione intra-BAN, ovvero la comunicazione che avviene all'interno delle BAN, fra i vari nodi sensore che comunicano fra di loro, il secondo è quello relativo alla comunicazione inter-BAN, ovvero alla comunicazione che avviene fra una BAN e l'altra, mentre l'ultimo livello di comunicazione è quello beyond-ban, ovvero tutte

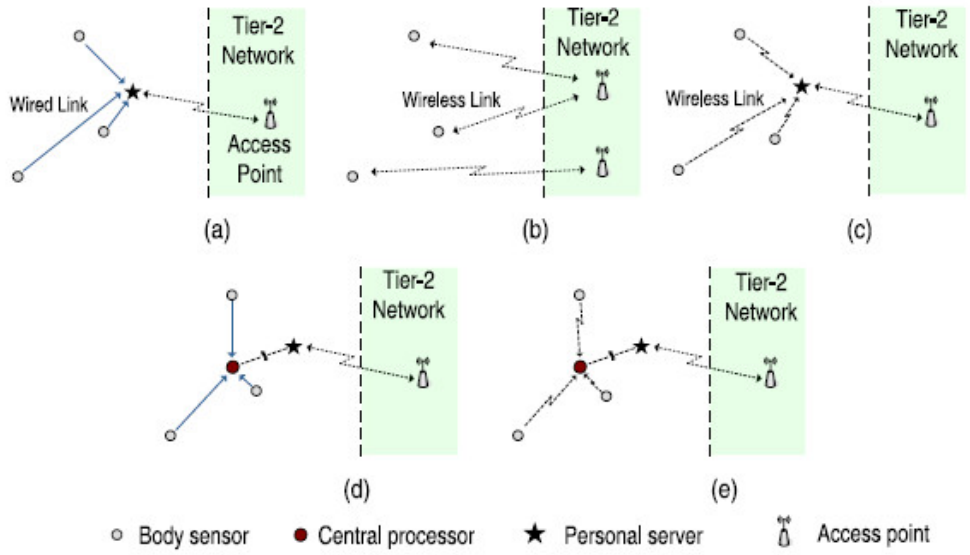


Figura 2.3: Configurazioni del primo livello di comunicazione

quelle interazioni che avvengono al di fuori delle body area network. Non esiste un confine assolutamente netto fra questi tre livelli, come invece si potrebbe intuire dalla figura 2.1, che rappresenta un esempio di un BAN per l'health-care.

Comunicazioni intra-BAN

Per comunicazioni intra-BAN, all'interno delle body area network, si può effettuare un'ulteriore distinzione, fra comunicazioni tra due sensori o dispositivi di misurazione e comunicazioni che avvengono invece fra i sensori e un PS (Personal Server), un dispositivo spesso anch'esso situato sul corpo monitorato (può essere uno smartphone o qualsiasi tipo di device in grado poi di comunicare con il secondo livello). I vari tipi di configurazione sono brevemente riassunti dalla figura 2.3.

Le comunicazioni fra i sensori sono solitamente comunicazioni radio a corto raggio (circa 2 metri), e possono essere di due tipi:

- Sensori wired: i sensori sono uniti tramite cavi ad un piccolo server personale, in grado di interagire con l'access point del secondo livello

tramite lo specifico protocollo

- Sensori wireless: i sensori sono essi stessi in grado di comunicare, utilizzando lo specifico protocollo necessario per utilizzare l'AP (Access Point), col secondo livello. Un altro tipo di architettura prevede invece che la rete wireless funzioni come l'esempio precedente, ovvero i sensori dialoghino con il PS, che poi tramite l'AP del secondo livello procede nella comunicazione.

I sensori possono poi essere semplici o complessi: quando i sensori sono semplici, i dati vengono inviati ad un processore centrale che ha poi il compito di inoltrarli al secondo livello, i dati trasmessi dai sensori sono etichettati come 'raw data' (dati grezzi); quando i sensori sono invece complessi (ovviamente sono più onerosi sia per quanto riguarda l'implementazione sia per quanto riguarda il costo degli stessi), effettuano una computazione interna prima di inviare i dati. Questo alleggerisce la computazione ed evita il cosiddetto effetto 'collo di bottiglia' per quanto riguarda il processamento dei dati nel PD o nel processore centrale.

Comunicazioni inter-BAN

Questo tipo di comunicazione si focalizza principalmente sulla distribuzione dei dati che vengono raccolti dai device che sono poi responsabili del loro utilizzo (un server, o un qualsiasi processore a cui potrebbero servire i dati). Si può effettuare un'ulteriore divisione di questo tipo di comunicazione, in base a quella che è l'architettura tramite la quale la comunicazione può funzionare: le architetture basate su un'infrastruttura e le architetture ad hoc.

Per quanto riguarda le architetture basate su un'infrastruttura fissa, spesso vengono utilizzate in ambienti ristretti, che difficilmente dovranno essere allargati in futuro e non necessitano di un'estensibilità particolare. É quindi un tipo di architettura che non permette un facile spread del sistema ad altri dispositivi e non si cura del fatto che i dati potranno in futuro essere distribuiti ad altri device che ne richiederanno l'utilizzo. Gli svantaggi che ne derivano sono quindi il fatto che difficilmente potrà essere aggiunto un altro nodo alla rete (intendendo come nodo un device che possa elaborare i dati), e che il sistema sia statico e non aperto e scalabile. I lati positivi

sono, grazie alla configurazione a stella che si viene così a creare, il fatto che gli accessi e la sicurezza vengano gestiti in modo centralizzato, e la semplicità di implementazione. Ovviamente al giorno d'oggi la tipologia di reti che si preferisce è quella del secondo tipo: quelle basate su un'architettura ad hoc. La caratteristica di queste reti è quella di essere aperte ed estendibili a piacimento, facendo in modo che i dati possano essere acceduti attraverso diversi access point, andando così ad allargare il dominio della rete. Gli AP possono essere aggiunti dinamicamente (anche se questo toglie alcune delle caratteristiche di sicurezza di cui sopra). I nodi possono essere divisi in due categorie per quanto riguarda questa architettura: esistono i sensori/attuatori, posti sul corpo, e i nodi router, che sono i responsabili poi della diffusione delle informazioni. Per quanto riguarda le tecnologie utilizzate, queste devono tenere conto che tutti i sensori utilizzano la stessa larghezza di banda e che ci potrebbero essere quindi delle collisioni durante le comunicazioni. Per ovviare al problema vengono utilizzati protocolli come ZigBee, che possono essere dotati anche di CSMA/CA (Carrier Sense Multiple Access dotate di Collision Avoidance). Questo tipo di architettura è utilizzata spesso in ambienti dove è necessario un rapido sviluppo di una rete dinamica e responsive, come ad esempio nelle strutture ospedaliere. La caratteristica fondamentale è quindi la possibilità di aggiungere access point in relazione a quali sono i bisogni dello specifico caso.

Per quanto riguarda le specifiche tecnologie che vengono utilizzate nella realizzazione di questo tipo di comunicazioni, si devono sicuramente evidenziare ZigBee e il Bluetooth. La particolarità del protocollo ZigBee è quella di funzionare in modo ottimale quando si tratta di reti cosiddette 'mesh' (ovvero reti nelle quali ogni nodo partecipa attivamente inviando dei dati). Ci sono anche altre tecnologie che vengono utilizzate spesso, come le WLAN o il cellular network, tra le quali la più utilizzata è sicuramente la prima, anche se, per quanto riguarda la seconda, il grande vantaggio deriva dal fatto che chiunque ormai è dotato di un cellulare e l'interfacciamento sia molto semplice. Il discorso relativo ai protocolli verrà comunque affrontato in modo più approfondito nel capitolo successivo.

Comunicazioni beyond-BAN

Questo livello di comunicazione prende in esame tutti gli specialisti che devono poter usufruire dei dati che vengono raccolti dalle reti di sensori in-

stallate sul corpo da analizzare. Queste comunicazioni sono spesso specifiche per ogni applicazione e permettono ad esempio ad un medico di conoscere i dati relativi ad un paziente accedendo un database. Se i due livelli precedenti erano specifici e si poteva dare quantomeno un'idea di quelle che possono essere le architetture più diffuse, per quanto riguarda questo livello di comunicazione gli scenari sono potenzialmente infiniti. O meglio, è quasi impossibile definire con certezza quali siano le comunicazioni oltre le BAN poiché la categoria è molto estesa. Grazie a questa comunicazione è possibile ad esempio intervenire a distanza su un utente, se un medico infatti viene a conoscenza del fatto che i dati di un determinato paziente indicano che si deve intervenire al più presto, può inviare un avviso al paziente e al personale addetto all'intervento.

2.3 Applicazioni

Se nel capitolo precedente era stato affrontato il discorso relativo alle applicazioni che possono essere fatte di un determinato sensore o device indossabile, qui vengono trattati aspetti che prendono anche in esame la necessità di dover rendere una BAN e tutti i suoi componenti partecipi all'interno di una rete. Le BAN sono ormai molto diffuse e vengono utilizzate per vari scopi. Permettono un monitoraggio che non ha più bisogno della presenza dell'entità monitorata e di chi poi va ad utilizzare i dati, ma ha semplicemente bisogno che le due parti siano in grado di comunicare fra di loro.

2.3.1 Ambito medico

Le body area network trovano grande diffusione nella medicina e nell'healthcare in generale. A dire il vero, lo scopo per cui sono nate è proprio questo. Come già accennato in precedenza, permettono ad un utente di essere monitorato a distanza da uno specialista, aumentando la possibilità di un intervento immediato se necessario. Questo è anche l'ambito in cui la ricerca relativa a materiali e comfort è più sviluppata. I sensori sono infatti spesso posizionati in regioni specifiche del corpo, e ben devono sapersi adattare ergonomicamente al paziente.

2.3.2 Fitness

I parametri e le funzioni vitali possono essere monitorate per poi essere inviate a specialisti in grado di poter leggere i dati, interpretarli ed utilizzarli per vari scopi (soprattutto, anche in questo caso, medici). Sono sempre più diffusi infatti questi sistemi di monitoraggio che utilizzano spesso come dispositivo di raccolta ed elaborazione dei dati uno smartphone, utile anche per raccogliere informazioni riguardanti la velocità e la posizione dello user. Spesso queste reti sono formate da sensori che controllano il battito cardiaco, per essere utilizzate per numerosi scopi di allenamento.

2.3.3 Ambito militare

Le BAN possono essere utilizzate anche per la rilevazione dei risultati di determinati addestramenti militari. In questo settore giocano anche un ruolo fondamentale le funzioni dello smartphone di cui sopra, per pianificare strategie ed azioni relative alla posizione dei soldati. In questo tipo di reti appaiono spesso dispositivi utili a lavorare con la realtà aumentata.

2.3.4 Gaming

Il mondo dei videogiochi è sempre più invaso da sensori (spesso non si trovano sul corpo, ma si tratta di sensori di movimento installati su sistemi dotati di telecamere). Le BAN potrebbero essere utilizzate per migliorare l'esperienza dei giocatori rilevandone movimenti e simulando l'immersione ad esempio in uno specifico ambiente. Più che per il gaming vero e proprio, la tecnologia delle BAN viene sfruttata dal lato dello sviluppatore, che in fase di progetto può utilizzare diversi sensori per simulare i movimenti che verranno poi effettuati all'interno del videogioco. Sempre più spesso infatti (un esempio riguarda i giochi di simulazione sportiva), vengono ingaggiati veri e propri atleti appartenenti al determinato sport trattato, ai quali viene applicata una BAN, per imitare poi i movimenti rilevati da complesse reti di accelerometri e giroscopi.

2.3.5 Information sharing

Alcune informazioni della vita di tutti i giorni potrebbero essere utilizzate per fornire servizi al cliente che ne richiede. Queste potrebbero essere

utili per diversi scopi: indagini di marketing relativi a quali possano essere i gusti dei clienti, reazioni a fronte di determinate sollecitazioni, o altro. Il tutto viene ottenuto tramite un adeguato mix di telecamere e sensori di EEG applicati.

2.3.6 Autenticazione e sicurezza

Tramite una BAN è possibile sviluppare strumenti per la sicurezza e l'autenticazione, un esempio possono essere le impronte digitali di una persona o il riconoscimento facciale. Ovviamente per fare ciò non serve solo un unico sensore, ma ci deve essere la possibilità poi di confrontare le credenziali inserite o mostrate con quelle attese, e per fare ciò deve essere possibile la comunicazione fra la BAN e ad esempio un server che gestisca i dati.

2.4 Consumo Energetico

I dati che vengono raccolti dai sensori devono essere poi inviati a chi li necessita (medici, esperti che li possano analizzare). Spesso però capita che il costo di trasmissione (in termini di banda ed energia) sia superiore a quello che i sensori possano permettere. Per questo vengono utilizzate tecnologie e protocolli che permettano di avere ottimi risultati a costi ridotti. In figura 2.4 è riportato un grafico che analizza il rapporto fra il data rate (espresso in bit al secondo, bps) e l'energia consumata utilizzando quello specifico protocollo per la trasmissione dei dati raccolti.

Per cercare di risolvere il problema, almeno per quanto riguarda il data rate, si può pensare di avere in primo luogo sensori 'intelligenti', che svolgano una parte della computazione al loro interno, e forniscano come risultati dei dati già elaborati, con dimensioni quindi ridotte, che devono poi essere inviati, e che saranno più facilmente trattabili essendo in quantità minore. Utilizzando meno banda per il trasferimento, e inviando meno dati, si abbassa anche il consumo di energia, sebbene questo tipo di reti dotate di sensori in grado di computare autonomamente siano molto più costose in termini economici.

Un altro aspetto da tenere in considerazione è quello dello storage dei dati prima che questi vengano elaborati. Spesso i dati che vengono raccolti non hanno bisogno di essere disponibili in real-time per il mondo esterno, ecco perché stanno continuando studi che prevedono di sviluppare memorie

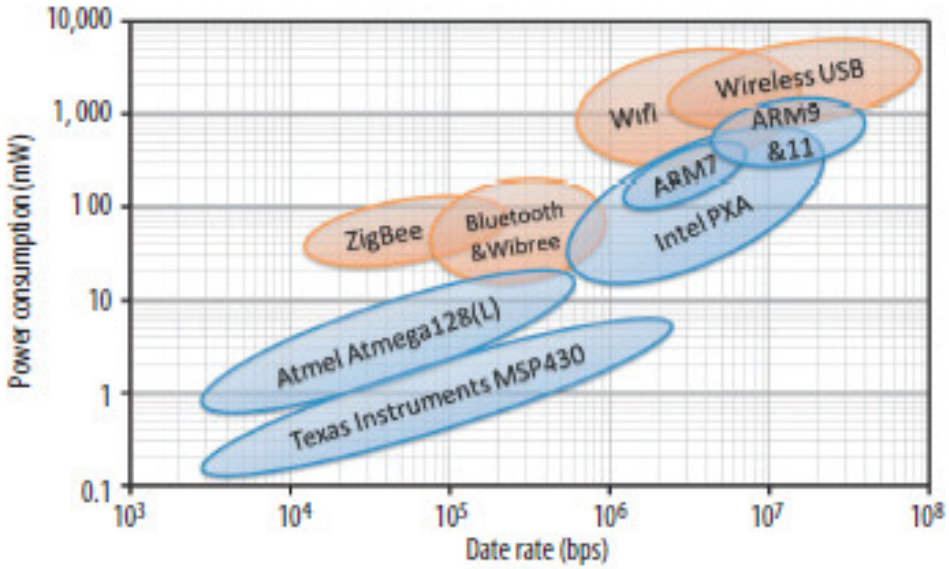


Figura 2.4: Consumo di energia in rapporto al data-rate

non volatili sempre meno onerose in termini di energia consumata, in grado di accumulare dati che, a fine monitoraggio, potranno essere utilizzati da chi li necessita. Questa particolare pratica prende il nome di on-node storage.

2.5 Conclusione

In questo capitolo si è discusso di quali siano gli utilizzi delle BAN nei vari ambiti e della loro architettura, oltre a definire a grandi linee il modo in cui i dispositivi sono collegati. Dopo un'introduzione generica a come i wearable comunicano nel prossimo capitolo verrà affrontato il discorso relativo ai protocolli.

Capitolo 3

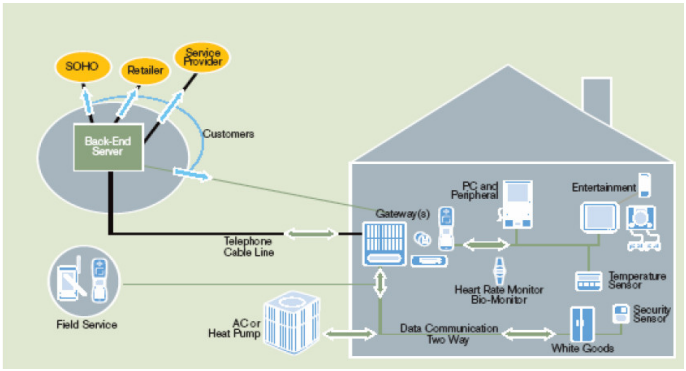
Protocolli

In questo capitolo verranno trattati i due principali protocolli di comunicazione quando si tratta di body area network. Si affronterà prima il discorso relativo a ZigBee, per poi passare a Bluetooth, soffermandosi sull'aggiornamento effettuato a Bluetooth 4.0 Low Energy, in modo da poter poi impostare un confronto fra i due protocolli e decidere quale dei due sia meglio utilizzare per affrontare problemi relativi alla creazione delle reti. I protocolli non verranno affrontati attraverso un punto di vista legato alle telecomunicazioni o all'elettronica, in modo da non uscire dall'argomento trattato dalla tesi.

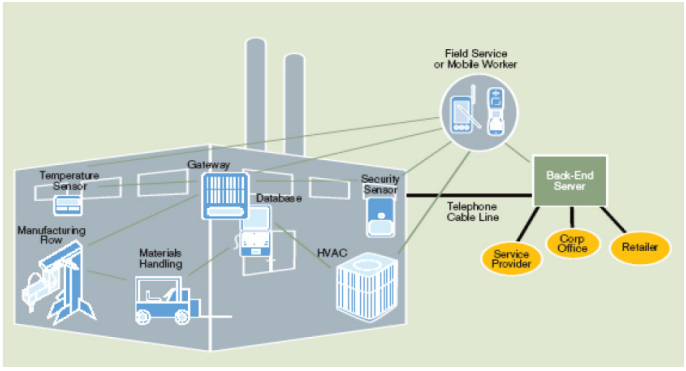
3.1 ZigBee

Lo standard di comunicazione ZigBee è l'evoluzione del wireless per quanto riguarda l'aggregazione di dispositivi wearable e la creazione di BAN. Deve il suo nome al caratteristico volo a zig-zag effettuato dalle api. É basato sullo standard IEEE 802.15.4 e il suo successo è dovuto a degli aspetti fondamentali che ne determinano la grande utilità ed efficacia:

- I network che vengono creati sono stabili, grazie al fatto che la velocità di trasmissione è piuttosto ridotta (ha un data-rate massimo di 250 kb/s). Questo non è un grande svantaggio solitamente quando si tratta di reti come le BAN, che non necessitano come già detto di grandi quantità di dati scambiati per essere operative.



(a) ZigBee in ambito domotico



(b) ZigBee in ambito industriale

Figura 3.1: Applicazioni di ZigBee

- Il consumo energetico è estremamente limitato, e questo permette a ZigBee di essere utilizzato in tutti gli ambiti che non debbano prevedere un alto consumo di energia, accontentandosi di una velocità di trasferimento dei dati non troppo alta, ad esempio la domotica.

Gli scenari applicativi di ZigBee sono molteplici, partendo dalla domotica e arrivando anche ad ambiti industriali, come riportato nella figura 3.1.

Il protocollo è stato ideato da Motorola, anche se successivamente si è formata un'associazione no profit, con membri quali la stessa Motorola, Samsung, Philips e Freescale, che si pone l'obiettivo di fornire uno standard di comunicazione efficace e a basso consumo, limitando per quanto possibile il bit-rate (rendendola quindi utile solo nei casi in cui la quantità di dati in-

<table>
<tr><th>Modello iso/osi</th><th>Semplificazione
Modello iso/osi</th><th>IEEE 802.15.4</th></tr>
<tr><td>Applicazione</td><td>Applicazioni utente</td><td rowspan="5">Applicazioni utente</td></tr>
<tr><td>Presentazione</td><td rowspan="3">Profilo applicazioni</td></tr>
<tr><td>Sessione</td></tr>
<tr><td>Trasporto</td></tr>
<tr><td>Livello di rete</td><td>Livello rete</td></tr>
<tr><td rowspan="2">Livello dati</td><td rowspan="2">Livello dati</td><td>Logical link control(MAC)</td></tr>
<tr><td>Media access control(MAC)</td></tr>
<tr><td>Livello fisico</td><td>Livello fisico</td><td>Livello fisico(PHY)</td></tr>
</table>

Figura 3.2: Modello ISO-OSI per IEEE 802.15.4

viati non debba essere troppo elevata). Questa associazione prende il nome di ZigBee Alliance. ZigBee, così come tutti i protocolli che si basano sull'IEEE 802.15.4, si preoccupa di definire solo i primi due livelli del modello ISO/OSI, ovvero quello fisico e quello di linea:

3.1.1 Banda di trasmissione

La banda adottata da ZigBee è l'ISM (Industrial, Scientific and Medical), che prevede che ci siano tre bande libere:

- 868-868.6 MHz: banda utilizzabile nella maggior parte dei paesi europei, con bit rate massimo di 20 Kb/s
- 902-928 MHz: banda utilizzabile maggiormente in Australia, Nuova Zelanda e nel continente americano, con bit rate massimo di 40 Kb/s
- 2400-2483,5 MHz: banda utilizzabile in tutto il mondo, con un bit rate massimo di 250 Kb/s

Lo standard IEEE 802.15.4 prevede che vi siano 27 canali, che sono numerati dallo 0 al 26. Di questi canali soltanto uno è allocato nella prima

banda, 10 sono allocati nella seconda, e i restanti 16 nella terza. ZigBee utilizza la terza banda libera disponibile.

Ovviamente ZigBee non è l'unico protocollo di comunicazione che sfrutta questa frequenza (si vedrà nella sezione successiva che lo fa anche Bluetooth ad esempio), ed è necessario quindi gestire le possibili 'collisioni' sfruttando la tecnologia del CSMA/CA (Carrier Sense Multiple Access con Collision Avoidance).

3.1.2 Utilizzo in ambito domotico ed industriale

Oltrepassando per un attimo il discorso relativo solo ed esclusivamente alle BAN, ci si può brevemente soffermare in un'esplorazione di quelle che sono le funzionalità di ZigBee per fare interagire dispositivi diversi. Per quanto riguarda ad esempio la domotica, con ZigBee è possibile controllare qualsiasi dispositivo o quasi. Per esempio è possibile controllare porte e finestre, sistemi di allarme, qualsiasi tipo di periferica video, condizionatori e addirittura l'accensione e lo spegnimento delle lampadine. Il tutto è reso più facile dal fatto che ZigBee sia una tecnologia wireless e permetta perciò di controllare numerose periferiche senza l'utilizzo di una rete cablata. Tutto ciò è utile anche e soprattutto in ambito industriale. Sono infatti numerosi gli utilizzi che possono essere fatti di questa tecnologia ad esempio all'interno di una fabbrica. Tutte le reti che prima dovevano essere 'wired', ora possono essere wireless, e basta quindi che un sensore o un dispositivo venga posizionato nel luogo utile al rilevamento di dati o all'attuazione di comandi.

ZigBee come già detto in precedenza risulta molto utile e molto efficiente quando si va ad affrontare il discorso relativo al consumo energetico, è questo infatti il vero punto di forza di questa tecnologia. Ovviamente il costo di questa qualità è quello di non poter utilizzare ZigBee in sistemi che necessitino l'invio di una quantità considerevole di dati in real-time, ma questo standard ben si adatta ad ambienti in cui non è necessario che le informazioni debbano essere particolarmente onerose.

3.1.3 Topologie di rete

Vediamo ora come si presenta la topologia di una rete composta da dispositivi che comunicano tramite ZigBee. La topologia di rete può riguar-

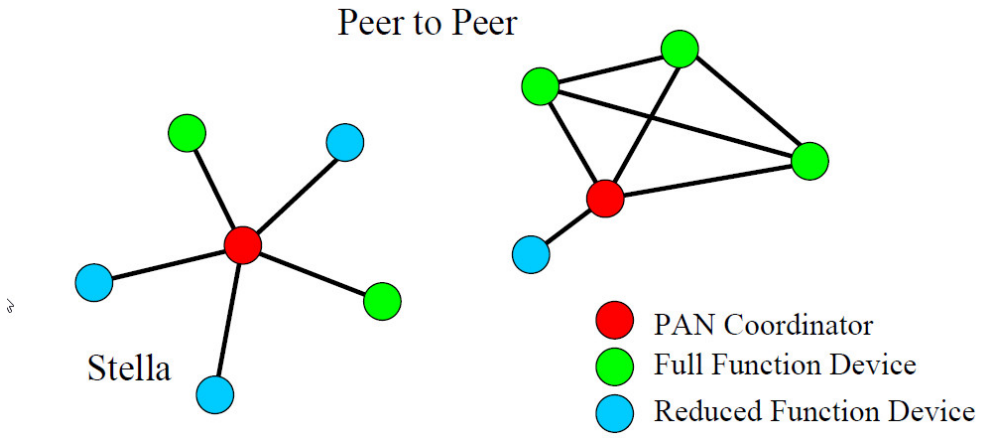


Figura 3.3: Struttura a stella e punto-punto

dare indifferentemente il primo o il secondo livello di comunicazione in una BAN (intra-BAN e inter-BAN), anche se il livello a cui si farà riferimento sarà soprattutto quello intra-BAN, solitamente soffermandosi sulla comunicazione che avviene fra i dispositivi e il Personal Server. In una rete che si basa sul protocollo ZigBee sono previsti due tipi di dispositivi:

- Full Function Device (FFD): Questo tipo di device ha al suo interno il livello del MAC con tutte le funzionalità e tutte le primitive
- Reduced Function Device (RFD): Questo tipo di device ha al suo interno un livello di MAC ridotto, che permette solo la realizzazione di dispositivi di rete

Ci sono poi due topologie distinte di rete:

- Rete a stella: Questa rete prevede che al centro ci sia un FFD, e che la rete poi si espanda facendo in modo che il punto centrale sia comunque sempre l'FFD. Attorno a questo device possono esserci sia FFD che RFD.
- Rete punto-punto (o peer-to-peer): In questo tipo di rete non c'è un dispositivo centrale, è quindi possibile la comunicazione fra vari dispositivi, uno con l'altro, anche se deve comunque essere presente

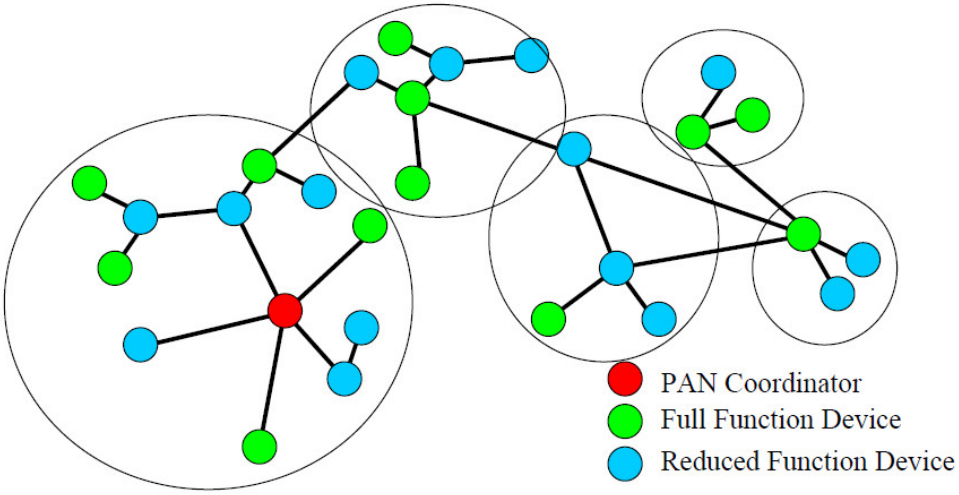


Figura 3.4: Struttura a cluster-tree

un coordinatore della rete. Un coordinatore della rete (coordinatore PAN) può essere un FFD. L'acronimo PAN sta per Personal Area Network.

Un'altra topologia supportata è quella a cluster-tree, un'architettura nella quale diverse sotto-reti interagiscono fra di loro per creare una macro-rete. La caratteristica di questo tipo di struttura è che le reti possono essere formate nel modo più consono agli scenari che si vengono a creare, ma un RFD può comunicare solo con RFD della sua stessa sotto-rete, e mai con RFD di altre sotto-reti. La struttura è illustrata in figura 3.4. Non è obbligatorio che ogni sotto-rete abbia un coordinatore PAN. Lo scambio fra i dati viene effettuato dai cluster head, che possono essere solo dispositivi FFD.

3.2 Bluetooth

Il nome di questa tecnologia deriva da Harald Blatand, Re della Danimarca attorno al X secolo. Il nome fu poi adattato all'inglese come Harald Bluetooth, il progetto cominciò appunto con questo nome e venne portato poi avanti senza cambiarlo. Il Bluetooth viene utilizzato per mettere in co-

municazione qualsiasi tipo di dispositivo dotato di una radio Bluetooth con un altro. Una radio Bluetooth ha dimensioni molto ridotte, è contenuta in un chip molto piccolo e necessita di una quantità di energia molto bassa per poter funzionare. Questo tipo di tecnologia prevede connessioni con dispositivi che si trovano a non più di 10 metri di distanza l'uno dall'altro. I dati possono essere inviati sia in modalità punto a punto, sia in modalità broadcast (punto-multipunto).

3.2.1 Frequenza di utilizzo e onde radio

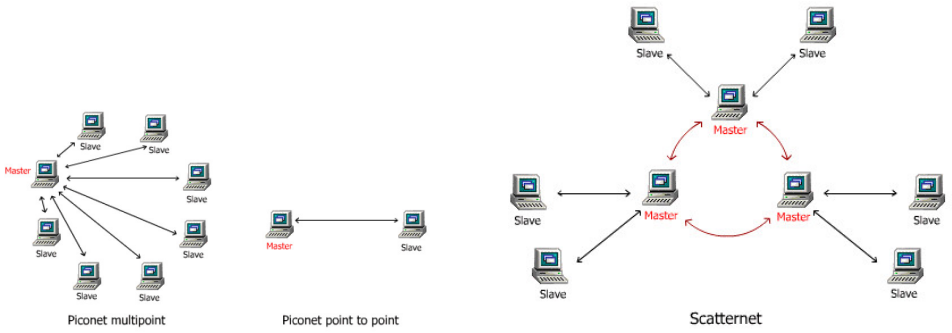
La comunicazione avviene attraverso onde radio, che hanno la particolarità di differire dai segnali scambiati, ad esempio, da due dispositivi che utilizzano la tecnologia ad infrarossi, perché le onde radio hanno la capacità di passare attraverso i materiali solidi, come per esempio la parete di un ufficio. Il Bluetooth opera sulla banda ISM (Industrial, Scientific and Medical), così come ZigBee. Più precisamente, sulla stessa identica banda di ZigBee, ovvero da 2,4 GHz a 2,483 GHz circa. Questa banda, come già accennato in precedenza, è molto affollata, e si potrebbero incontrare dunque altre comunicazioni che causerebbero interferenze ed errori. Per ovviare al problema, il protocollo Bluetooth utilizza una particolare soluzione, denominata hopping di frequenza a largo spettro (FHSS), che prevede che durante una comunicazione Bluetooth i dati passino automaticamente da una frequenza all'altra in modo molto rapido, circa 1600 hop al secondo (hps, hop per second). I dati vengono inviati a pacchetti.

3.2.2 Topologie di rete

Ogni volta che due o più dispositivi Bluetooth si connettono fra di loro formano una Piconet, che può essere di due tipi:

- Punto-punto: due dispositivi parlano fra di loro, e nessuno dei due ha il controllo sull'altro
- Punto-multipunto: Un dispositivo parla con più dispositivi (fino ad un massimo di 7, escluso quello principale). In questo tipo di collegamento il dispositivo centrale viene chiamato master, gli altri slave.

Per creare reti di dimensioni maggiori si possono unire varie Piconet fra di loro. Per fare ciò si deve fare in modo che gli intermediari della



(a) Struttura di una Piconet (b) Struttura di una Scatternet

Figura 3.5: Applicazioni di ZigBee

connessione siano sempre e comunque i vari master delle sottoreti. Questo nuovo tipo di configurazione prende il nome di 'Scatternet'.

Un esempio di questi tipi di collegamento è illustrato in figura 3.5

3.2.3 Bluetooth Low Energy

Nella sezione precedente non è stato affrontato il discorso dell'utilizzo della tecnologia Bluetooth nei vari campi di interesse. Questo perchè per quanto riguarda le body area network, si preferisce utilizzare una sua evoluzione: il Bluetooth LE (Low Energy), anche chiamato Bluetooth Smart. Questo protocollo è nato per contrastare lo strapotere di ZigBee che stava imponendosi come protocollo principale, grazie al bassissimo consumo di energia e alla grande versatilità. Tramite il Bluetooth Low Energy si possono creare reti molto estese e reattive, e questa tecnologia è utilizzata in diversi campi. Diventa, in questo caso, molto importante il discorso dei profili Bluetooth, ovvero il ruolo che ogni device può ricoprire in un'applicazione. Tra questi, due molto importanti sono i profili per l'health-care e quelli per l'attività sportiva e il fitness.

3.3 Differenze fra Bluetooth e ZigBee

Data la grande concorrenza di tecnologie come ZigBee, utili soprattutto per le BAN, Bluetooth si è evoluto nel tempo mettendo a punto una

soluzione per contrastare il suo più grande difetto: l'eccessivo utilizzo di energia. Nasce così come visto nella sezione precedente il Bluetooth a basso consumo energetico (Low Energy). É ora quindi possibile confrontare ZigBee e BLE per decretare quale delle due tecnologie sia più utile nella creazione e gestione di BAN. Ovviamente per questo confronto si ricorda ancora una volta che lo scopo è quello di rimanere il più possibile legati all'aspetto informatico di questo eleborato, e ciò esclude le considerazioni dovute ad aspetti inerenti alle telecomunicazioni.

Bluetooth BLE si comporta quasi come ZigBee per quanto riguarda il bit rate in confronto all'energia che viene consumata (si parla di bit per joule). Il problema di ZigBee riguarda alcuni aspetti di interoperabilità, cosa che invece Bluetooth gestisce discretamente bene, anche se c'è da aggiungere che una Piconet Bluetooth può supportare un massimo di 8 dispositivi (1 master e 7 slave). In definitiva, le differenze esistenti fra Bluetooth BLE e ZigBee portano a concludere che ZigBee sia più indicato per reti che hanno un grande numero di sensori (può supportare anche reti con oltre 65000 device), mentre Bluetooth (BLE) è indicato per reti di piccole dimensioni. Per quanto riguarda le BAN (formate da pochi sensori), la scelta del protocollo non influenza più di tanto l'efficienza della rete, soprattutto se si tratta, come spesso accade, di reti ad hoc, che non necessitano di prestazioni elevatissime. Prendendo in considerazione i pochi aspetti trattati comunque, il protocollo più indicato per creare BAN formate da pochi sensori rimane il Bluetooth (nella sua versione Low Energy o nella sua versione originale) mentre per quanto riguarda la creazione di reti di sensori/attuatori, come ad esempio le reti di controllo domotiche, il protocollo più indicato è senza dubbio ZigBee.

Capitolo 4

Android

Android è un sistema operativo ormai molto diffuso a livello mondiale, non solo su smartphone o tablet, ma su qualsiasi tipo di dispositivo. La sua rapida e crescente diffusione si deve certamente al fatto che sia completamente open-source e abbia quindi attirato tutti gli sviluppatori software che apprezzano questo tipo di tecnologia. Un altro motivo per il quale Android ha avuto una rapida diffusione, è sicuramente da ricercare nel mercato che sta dietro a questo sistema operativo. Android si trova infatti ormai sulla maggior parte degli smartphone e tablet accessibili al pubblico, e gode di una forte adattabilità ad una gamma molto estesa di dispositivi. Non sempre questo è però da interpretarsi solo come un vantaggio, ovviamente dal punto di vista dello sviluppatore, in quanto chi sviluppa su questo sistema operativo deve prestare particolare attenzione al fatto di lavorare su piattaforme che condividono sì Android, ma che sono estremamente eterogenee fra di loro. Tutto questo ha portato i produttori dei dispositivi ad adattare i loro prodotti in modo da favorirne la diffusione. Android si sposa in modo ottimo con l'applicazione in ambito wearable. Il sistema operativo nasce infatti come soluzione per sistemi embedded, dotati di basse prestazioni e scarsa memoria. Tutto ciò si combina in pieno con le tecnologie esposte nei capitoli precedenti, che spesso sono di dimensioni molto ridotte e non dispongono di una grande potenza computazionale. Il fatto poi che sia un sistema operativo estremamente estendibile, fa sì che Android si adatti alla perfezione a lavorare per garantire l'interazione e la comunicazione dei dispositivi wearable e dei sensori, portando con sé una grande facilità nell'implementazione delle BAN formate da dispositivi eterogenei

fra di loro.

In questo capitolo verrà illustrata brevemente la struttura di Android, e verranno illustrati i componenti fondamentali del sistema operativo. Si introdurrà poi il discorso relativo al protocollo Bluetooth associato a questo OS, che verrà utilizzato nel caso di studio affrontato nel capitolo successivo. Si fornirà infine un breve tutorial per (lato client) comunicare attraverso il protocollo Bluetooth con un dispositivo che lo permetta.

4.1 Introduzione ad Android

Alla base dei sistemi Android ci sono due principi fondamentali: il primo è quello che prevede la salvaguardia delle risorse e il secondo è la sicurezza. Android nasce in principio, come accennato sopra, come sistema operativo utile a lavorare in sistemi embedded, con a disposizione quindi pochissima memoria, che deve essere utilizzata con parsimonia. Una funzionalità molto importante offerta è quella di distruggere e ricreare parti di un'applicazione solo ed esclusivamente quando serve. Android è un sistema operativo basato su Linux, che pone molta attenzione a non fare interagire in modo diretto le varie applicazioni fra di loro, e gli assegna infatti un processo ciascuna. Ovviamente le applicazioni possono comunicare ed interagire, ma non potranno mai occupare lo spazio l'una dell'altra. I quattro elementi fondamentali di un'applicazione sono:

- Le activity: Sono una sorta di pagina tramite la quale vengono letti dei dati o si possono inserire input, sono l'elemento più importante
- I service: Sono la vera e propria parte computazionale. Svolgono un lavoro spesso in background senza la necessità di interagire con l'utente
- I content provider: Sono i componenti che meglio sintetizzano la proprietà della sicurezza. Permettono infatti che ci sia dialogo fra le applicazioni in modo sicuro
- I broadcast receiver: Sono componenti che reagiscono ad un determinato messaggio di sistema (broadcast). Sono utilizzati ad esempio per notificare la ricezione di un messaggio o di una chiamata

Infine c'è un altro componente che non viene considerato fondamentale ma riveste comunque una sua importanza: l'intent. Un intent serve fondamentalmente a far interagire due activity. Android, come già detto, ha come uno dei suoi punti saldi una corretta gestione della memoria, ed è anche spesso integrato in dispositivi come gli smartphone, che raramente vengono spenti consentendo quindi una corretta pulizia della memoria. Questo fa sì che il sistema operativo debba effettuare una scelta su quali processi mantenere in vita e quali no, in modo da liberare spazio utile. Il sistema è quindi in grado di riconoscere al momento opportuno quali siano i processi da mantenere in vita e quali invece possono essere arrestati, in modo da liberare la memoria per altre applicazioni. In Android i processi vengono così classificati:

- Processi in foreground: Sono quel tipo di processi che vengono attualmente visualizzati dall'utente, la cui activity è utilizzata al momento, e sono in un certo senso i processi più importanti in un determinato istante. Raramente vengono terminati, anche se può capitare in casi eccezionali

- Processi visibili: Sono processi che vengono in ogni caso visualizzati dall'utente, anche se l'interazione potrebbe essere terminata. Hanno una priorità alta e anche per quanto riguarda questi processi è raro che vengano arrestati

- Processi service: Sono processi, o meglio veri e propri service, che svolgono un qualche servizio in background, non sono ovviamente importanti come i primi due e non vi è interazione con l'utente, hanno perciò una priorità media

- Processi in background: Sono spesso activity non più visibili all'utente, che, magari involontariamente, è uscito da un'applicazione senza chiuderla correttamente. Possono quindi essere tranquillamente arrestati dal sistema.

- Processi empty: Sono quei processi che non hanno più alcun componente attivo, vengono sempre eliminati a meno di necessità di caching per permetterne poi una facile riapertura in futuro.

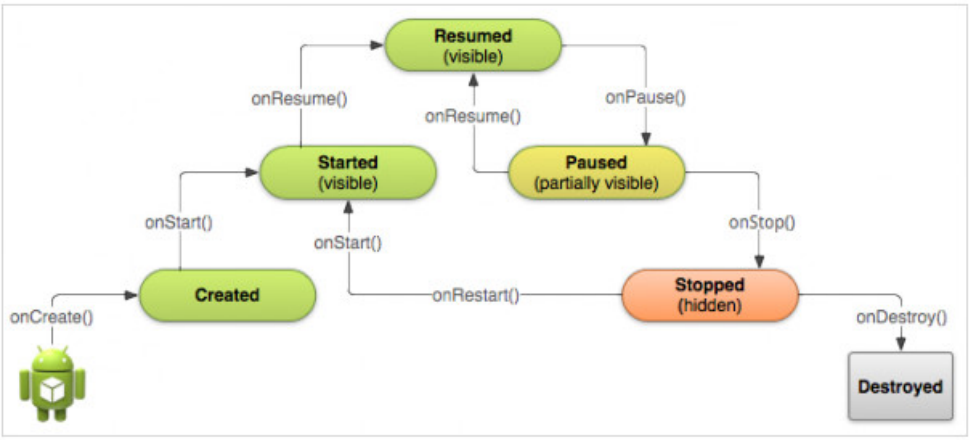


Figura 4.1: Ciclo di vita di un'activity

4.1.1 Il ciclo di vita di un'activity

La figura 4.1 rappresenta il ciclo di vita di un'activity all'interno di Android. L'activity, come si può notare, passa attraverso tutti gli stati almeno una volta, dalla creazione alla sua distruzione. L'immagine è anche contenuta nella documentazione ufficiale Android e risulta molto utile al momento della programmazione, visto che come si può intuire dalla semantica, ogni entrata o uscita da uno stato presuppone un'invocazione di un metodo, che deve essere implementato in relazione all'azione che si vuole che l'activity compia. Il colore degli stati è stato selezionato in relazione alla visibilità che quell'activity ha in quel determinato momento. Ad esempio, il colore verde sta ad indicare che l'activity è visibile, e l'interazione con l'utente è abilitata. Il colore giallo invece, presente nello stato 'Paused', indica come l'activity sia ancora parzialmente visibile, ma che non vi sia più interazione (momentaneamente) con l'utente. Lo stato 'Stopped' è colorato in rosso, e quando un'activity entra in questo stato è pronta per essere poi distrutta. In realtà è ancora possibile recuperare un'activity in stato 'Stopped', ma l'activity deve essere fatta partire di nuovo, e non può essere semplicemente 'resumed' come nel caso dello stato 'Paused'. Andando ad analizzare più in profondità la normale vita di un'activity si hanno due fasi. La prima avviene quando viene mandata in esecuzione, per renderla disponibile all'interazione con l'utente. Vengono così obbligatoriamente invocati tre metodi:

- onCreate(): l'activity viene creata, definendo un'interfaccia grafica, che sarà poi quella visualizzata dall'utente
- onStart(): dopo l'invocazione di questo metodo, l'activity diventa effettivamente visibile all'utente
- onResume(): l'activity in questo modo diventa totalmente a disposizione dell'utente, e pronta a ricevere qualsiasi tipo di input

Il percorso inverso di un'activity invece si ha quando l'utente smette di utilizzarla e ne apre un'altra, anche magari rimanendo nella stessa applicazione. I tre metodi fondamentali che vengono invocati sono:

- onPause(): dopo essere stato invocato, l'utente non può più interagire con l'activity
- onStop(): l'activity smette di essere visibile
- onDestroy(): l'activity viene distrutta, e non può più essere recuperata in alcun modo

I metodi devono essere solo definiti ed implementati dallo sviluppatore, poiché la loro esecuzione è gestita dal sistema operativo.

4.1.2 Le risorse in Android

Il codice Java all'interno di un'applicazione Android ha spesso bisogno di risorse esterne per poter esser eseguito (come stringhe, immagini, o altro ancora). Android prevede che tutte le risorse statiche debbano essere contenute in una particolare cartella del progetto, la cartella res, al cui interno sono contenuti tutti i valori di cui l'applicazione avrà bisogno. Gli esempi classici di risorse per un'applicazione Android sono:

- Layout: è l'architettura grafica delle activity, quella che viene imposta nel metodo onCreate()
- Values: è la categoria in ambito più generico, può contenere qualsiasi tipo di valore, anche se i più utilizzati sono stringhe, colori e dimensioni. Possono essere richiamati all'interno del codice, in alternativa alla loro definizione cosiddetta hard-coded, nella quale vengono definiti direttamente nel codice

- Drawable: contiene fondamentalmente immagini dei formati più comuni, ma potrebbe anche contenere immagini definite in XML

Le risorse sono definite in XML, e successivamente viene automaticamente generato del codice Java relativo ad esse. Possono quindi essere richiamate in due modi:

- In Java:

```
R.tipo_risorsa.nome_risorsa;
```

- In XML:

```
@tipo_risorsa/nome_risorsa
```

É anche possibile definire un ID univoco per ogni risorsa, in modo che sia possibile poi recuperarla tramite quest'ultimo. Questo in ogni caso non preclude la possibilità di creare ambienti di variabili o costanti direttamente nel codice Java.

4.1.3 L'app manifest

Un componente fondamentale dell'applicazione è l'app manifest. Si tratta di un file XML necessario per ogni applicazione. Fornisce al sistema operativo le informazioni di base di un'app, ed è necessario definire al suo interno una nuova activity prima di implementarla in Java. Gestisce anche i permessi per accedere a determinate API, come quelle relative al Bluetooth che verranno trattate in seguito, oltre ad avere tantissime altre funzionalità che verranno illustrate se e quando ce ne sarà espressamente bisogno.

4.2 Android e Bluetooth

In questa sezione si forniranno le istruzioni di base per poter lavorare con il protocollo Bluetooth in Android. In particolar modo si affronterà il discorso relativo al client Bluetooth, più che al server, dato che l'applicazione che verrà trattata in seguito necessita di questo aspetto. Verranno riportati esempi di codice di carattere generale, non necessariamente perciò utilizzati all'interno dell'applicazione trattata nel capitolo successivo.

4.2.1 Permessi Bluetooth

Nella sezione precedente è stato introdotto il discorso relativo all'app manifest. Questo componente risulta fondamentale per quanto riguarda i permessi necessari per poter lavorare con il Bluetooth. La riga di codice che è necessario aggiungere è:

```
<uses-permission android:name="android.permission.BLUETOOTH" />
```

Questo permesso è necessario per avviare un qualsiasi tipo di comunicazione, come la richiesta o l'accettazione di una connessione, e il trasferimento dei dati. Nel caso il dispositivo debba anche effettuare una discovery (un'esplorazione per la ricerca di dispositivi Bluetooth nelle vicinanze), è necessario un altro permesso:

```
<uses-permission
    android:name="android.permission.BLUETOOTH_ADMIN" />
```

4.2.2 Elementi fondamentali

Per quanto riguarda gli oggetti fondamentali che si devono conoscere quando si affronta il discorso Bluetooth con Android troviamo:

- `BluetoothAdapter`: Rappresenta l'adapter Bluetooth locale, ovvero la radio Bluetooth fisica. É l'entry-point di tutte le interazioni
- `BluetoothDevice`: Come si può intuire facilmente dal nome, è la rappresentazione di un qualsiasi dispositivo remoto dotato di radio Bluetooth. I suoi metodi possono essere utilizzati per richiedere una connessione o qualsiasi altra informazione relativa al dispositivo, come il suo indirizzo
- `BluetoothSocket`: Rappresenta l'interfaccia di una socket Bluetooth ed è molto simile ad una socket TCP. É il canale di comunicazione che permette lo scambio dei dati attraverso gli appositi stream (`InputStream` ed `OutputStream`)
- `BluetoothServerSocket`: É una socket aperta, in attesa di richieste, lato server. Quando un device richiede una connessione si ve-

de restituita una `BluetoothSocket` connessa, con la quale può poi interagire

4.3 Connessione con un dispositivo Bluetooth (lato client)

In questa sezione verrà brevemente illustrato come creare una connessione fra due dispositivi Bluetooth, in particolare come fare una richiesta di connessione lato client. Per un approfondimento relativo alla programmazione lato server, o riguardo a qualsiasi altro aspetto della programmazione Android, si rimanda al sito ufficiale, contenente un esauriente tutorial: `http://developer.android.com/`.

4.3.1 Attivazione della radio Bluetooth

Per ottenere una qualsiasi interazione con un dispositivo Bluetooth (lato client) è necessario passare attraverso due fasi. La prima è quella che riguarda l'attivazione del Bluetooth e la seconda è quella che riguarda la richiesta di connessione vera e propria. Per quanto riguarda l'attivazione ci sono due passi fondamentali:

1. La prima cosa da fare è recuperare un oggetto che rappresenta il componente fisico presente in ogni dispositivo dotato di Bluetooth, ovvero la radio Bluetooth. Per fare ciò nella quasi totalità dei casi, si richiede al dispositivo la sua radio di default, tramite il metodo `getDefaultAdapter()` della classe `BluetoothAdapter`. Un esempio di codice è riportato qui di seguito:

```
BluetoothAdapter btAdapter =
    BluetoothAdapter.getDefaultAdapter();
if (btAdapter == null) {
    //Il dispositivo non supporta il Bluetooth
}
```

   Il metodo `getDefaultAdapter()` restituisce, se possibile, un oggetto di tipo `BluetoothAdapter`, su cui poi si invocheranno altri metodi. Nel caso in cui il metodo ritorni `null`, questo significa che il device

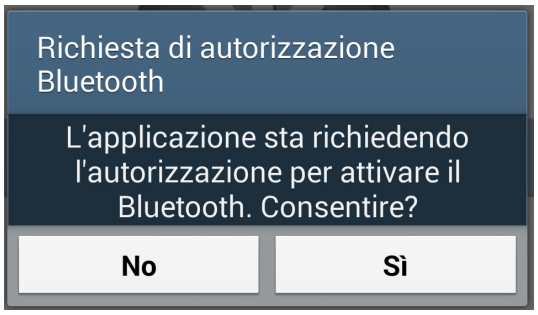


Figura 4.2: Dialog per l'attivazione Bluetooth

non supporta il Bluetooth, e non ha una radio Bluetooth per poter interagire con altri dispositivi. Nel caso in cui non sia presente la radio Bluetooth fisica, non è possibile utilizzare il Bluetooth in alcun modo, a meno che non si disponga di una radio Bluetooth esterna da poter attaccare al dispositivo.

2. La seconda cosa da fare è attivare il Bluetooth sul dispositivo. Solitamente infatti, la radio Bluetooth di molti dispositivi rimane spenta se inutilizzata, in attesa di essere accesa se necessario. Viene effettuato un controllo sull'adapter, nel caso fosse già acceso, e, in caso contrario, tramite un `Intent` (che, come già discusso in precedenza, è il metodo usato dalle activity per interagire), viene attivato.

```
if (!btAdapter.isEnabled()) {
    Intent enableBtIntent = new
        Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBtIntent,
        USER_DEFINED_REQUEST_ID);
}
```

L'`Intent` in oggetto crea una dialog, come mostrato in figura 4.2, tramite la quale l'utente può attivare il Bluetooth.

Analizzando il codice precedente, ci si deve soffermare su un aspetto fondamentale di Android. Quando l'`Intent` viene mandato in esecuzione, questo deve per forza essere fatto all'interno di un'activity, che dopo aver lanciato la dialog con relativa attivazione Bluetooth, continua la sua esecuzione, senza curarsi del fatto che l'attivazione

non sia ancora stata effettuata. Per ovviare a questo, come a tutti i problemi di questo genere, Android fa un grande utilizzo delle callback. Una volta che viene invocato un metodo come ad esempio `startActivityForResult()` nel codice poco sopra, l'`Intent` viene lanciato, e viene poi richiamato il metodo `onActivityResult()` all'interno dell'activity che ha lanciato l'`Intent`. Questo metodo deve perciò essere implementato in modo da compiere qualsiasi azione che sia relativa alla corretta (o non corretta) terminazione degli `Intent` invocati. Ovviamente il metodo nella sua forma completa ha degli argomenti in ingresso

```
protected void onActivityResult (int requestCode, int
    resultCode, Intent data)
```

e deve quindi tener conto di quale sia il ritorno che sta trattando. Ciò avviene attraverso il controllo del valore di ingresso `requestCode`, che nel caso preso come esempio sarà `USER_DEFINED_REQUEST_ID`, una costante intera definita dall'utente. Nel campo `resultCode` sarà contenuto il valore di `RESULT_OK` se l'operazione è andata a buon fine, e `RESULT_CANCELED` in caso contrario. Il terzo argomento è un oggetto di tipo `Intent`, contenente eventuali parametri di ritorno.

4.3.2 Ricerca dei dispositivi

La seconda fase dell'operazione è quella della ricerca dei dispositivi. Esistono due tipi di dispositivi dal punto di vista del device utilizzato: quelli associati e quelli non associati. I device associati sono device con cui in passato molto probabilmente c'è già stato uno scambio di informazioni. I dispositivi possono essere associati al proprio device tramite le funzioni di routine messe a disposizione di default. Spesso per l'associazione viene richiesto di inserire un codice pin, per ragioni di sicurezza. Volendo cercare i dispositivi con Bluetooth attivo nell'ambiente circostante, è necessario avviare una discovery. Per fare ciò bisogna avere a disposizione un oggetto di tipo `BluetoothAdapter` e fare in modo che questo sia attivo. Il codice Java è il seguente:

```
// Creazione di un set di BluetoothDevice
private Set<BluetoothDevice> devices = new Set<BluetoothDevice>();
```

```
// Creazione di un receiver per ACTION_FOUND
private final BroadcastReceiver receiver = new
    BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            BluetoothDevice device =
                intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            mArrayAdapter.add(device);
        }
    }
};
// Registrazione del receiver
IntentFilter filter = new
    IntentFilter(BluetoothDevice.ACTION_FOUND);
registerReceiver(receiver, filter);
// Inizio della discovery
btAdapter.startDiscovery();
```

Cominciamo dal meccanismo con il quale viene effettuata la discovery. Un receiver viene creato in Android per 'ricevere' qualcosa. In questo caso viene creato (e successivamente registrato) allo scopo di accorgersi se un dispositivo viene trovato attivo nel raggio d'azione della radio Bluetooth. Inizialmente viene definito un set di oggetti di tipo `BluetoothDevice` per avere a disposizione tutti i dispositivi che vengono trovati durante la discovery. Successivamente si va a definire quale dovrà essere il comportamento del receiver nel caso in cui il messaggio ricevuto sia quello di aver trovato un nuovo dispositivo (ovvero, in questo caso, `ACTION_FOUND`). Il comportamento viene definito all'interno del metodo

```
public void onReceive(Context context, Intent intent)
```

nel quale `context` è il contesto di esecuzione, in questo caso l'activity, e `intent` è un oggetto di tipo `Intent` che viene restituito (come già accennato in precedenza) per recuperare i parametri di ritorno, che in questo caso corrispondono al dispositivo Bluetooth trovato. Il dispositivo è contenuto negli extra dell'oggetto di tipo `Intent`. Questi extra possono essere recuperati tramite il metodo `getParcelableExtra()`, andando a analizzare solo quello di cui si ha bisogno, in questo caso tramite l'id `BluetoothDevice.EXTRA_-`

DEVICE. Successivamente il dispositivo viene aggiunto al set creato in precedenza, in modo da essere disponibile per un utilizzo futuro. Occorre ora registrare il receiver, e ciò viene effettuato utilizzando anche un filtro, che limiti in un certo senso i messaggi a cui il receiver è interessato (in questo caso solo BluetoothDevice.ACTION_FOUND. Non resta altro che cominciare la discovery tramite il metodo startDiscovery() dell'oggetto BluetoothAdapter creato in precedenza. La discovery deve essere poi arrestata al momento opportuno (per esempio quando l'activity viene chiusa) tramite il metodo opposto, cancelDiscovery().

Nel caso in cui invece il dispositivo di interesse sia già associato al device che vuole agire da client, la procedura risulta molto più semplice e rapida. Tutto ciò che è necessario fare infatti, è richiamare il metodo getBondedDevices() dell'oggetto BluetoothAdapter

```
private Set<BluetoothDevice> devices =
    btAdapter.getBondedDevices();
```

che ritorna un set di BluetoothDevice, pronti per essere utilizzati. Nel caso in esame, volendo agire da client, verrà scelto il dispositivo server col quale si intende interagire, e gli si richiederà una connessione.

4.3.3 Connessione

La fase successiva è quindi quella di recuperare i dati necessari per la richiesta di connessione al server. É necessario conoscere l'indirizzo del dispositivo a cui ci si vuole connettere. La procedura di base per la connessione ad un server Bluetooth prevede due passi fondamentali:

1. Utilizzando l'oggetto di tipo BluetoothDevice a cui ci si vuole connettere, si ottiene una BluetoothSocket, tramite l'invocazione del metodo createRfcommSocketToServiceRecord(UUID). Il parametro UUID è un intero identificativo del particolare servizio messo a disposizione dal server, ed è recuperabile in due modi. Il primo è quello di copiare l'UUID definito quando si è programmato il lato server, il secondo è quello di recuperarlo attraverso un metodo di get. Ovviamente, spesso i dispositivi da interfacciare non mettono a disposizione dell'utente finale il codice sorgente, ed è quindi necessario percorrere la

seconda strada. Un UUID può essere recuperato attraverso l'oggetto `BluetoothDevice` preso in considerazione, in questo modo:

```
int UUID = BluetoothDevice.getUuids()[i].getUuid();
```

dove `i` rappresenta l'indice dell'UUID di cui si ha bisogno (nella maggior parte dei casi ce n'è soltanto uno, perciò `i`=0).

2. Il metodo `createRfcommSocketToServiceRecord(UUID)`, se va a buon fine, restituisce una socket su cui è poi necessario effettuare una richiesta di `connect()`, in questo modo:

```
private BluetoothSocket btSocket;
try {
   btSocket = device.createRfcommSocketToServiceRecord(UUID);
} catch (IOException e) {
        //Eventuale gestione dell'eccezione
}
try {
   btSocket.connect();
} catch (IOException connectException) {
      //Eventuale gestione dell'eccezione (in questo caso la
          socket viene chiusa)
   try {
   btSocket.close();
    } catch (IOException closeException) {
        //Eventuale gestione dell'eccezione
    }
}
```

L'operazione di `connect()` ha un comportamento bloccante, ed una delle regole fondamentali di Android è che il flusso principale di controllo non debba mai bloccarsi, per non proibire l'interazione con l'utente. Tutte le operazioni di `connect()` e le successive operazioni di lettura e scrittura devono essere effettuate da un thread separato.

4.3.4 Lettura e scrittura

Una volta che la richiesta di `connect()` è stata accettata, non rimane altro che cominciare a leggere e scrivere sulla socket, con banali operazioni di read e write. Un esempio di codice è:

```
InputStream inStream = socket.getInputStream();
byte[] buffer = new byte[1024];
while (true) {
   try {
      // Lettura dei dati dallo stream
      inStream.read(buffer);
    } catch (IOException e) {
      break;
    }
}
```

La stessa cosa può essere effettuata per i processi di scrittura. É bene, ogni volta che si smette di utilizzare una socket, chiuderla, con il metodo `close()`.

Capitolo 5

Caso di studio su Android e WristOx2

In questo capitolo verrà discussa la realizzazione di un'applicazione, nella quale verranno utilizzati i concetti esposti in precedenza, sfruttando il dispositivo WristOx2, un pulsiossimetro dotato di un display wearable per il polso. Per la realizzazione di questa applicazione viene scelto Android, poiché, come discusso nell'introduzione del capitolo precedente, gode della proprietà di essere fortemente estendibile e di essere adattabile a lavorare con moltissimi dispositivi, anche fra loro eterogenei. La scelta del pulsiossimetro ricade invece sul WristOx2 (prodotto dall'azienda Nonin Medical), dal momento che il device è facilmente interfacciabile con un dispositivo e lavora tramite il Bluetooth, uno dei due protocolli più utilizzati nella creazione di questo tipo di reti. Sfruttando Android è possibile anche fare in modo che l'applicazione abbia la proprietà di essere fortemente estendibile, proprietà che verrà poi discussa nella sezione apposita.

5.1 Obiettivi

L'obiettivo della creazione dell'applicazione è quello di interfacciare il pulsiossimetro ad uno smartphone, applicando le conoscenze studiate nel corso della stesura di questo elaborato, su Android e sui protocolli di comunicazione. I due dispositivi dovranno interagire attraverso il protocollo Bluetooth (nella sua versione classica, non quella Low Energy). Dovrà essere possibile visualizzare i dati inviati dal pulsiossimetro sullo schermo dello

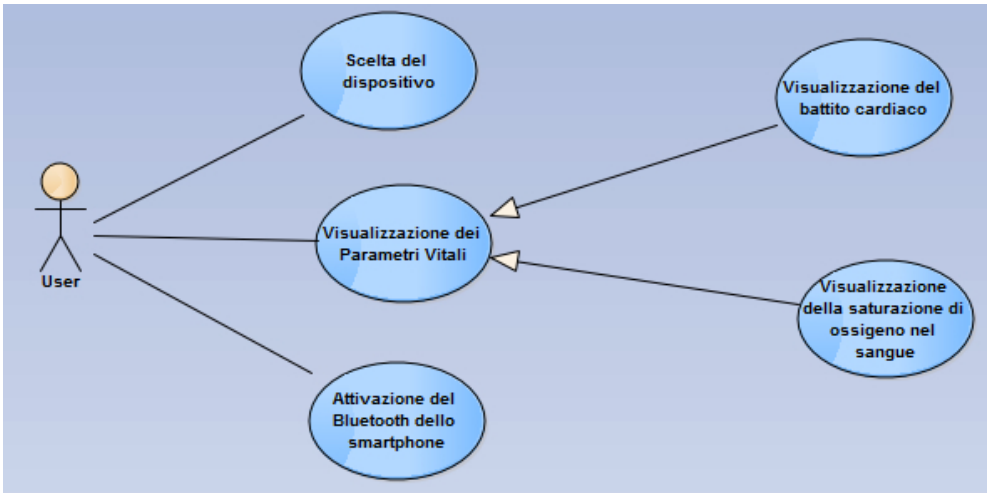


Figura 5.1: Casi d'uso

smartphone. Per la creazione dell'applicazione e per il testing viene utilizzato un dispositivo dotato del sistema operativo Android aggiornato alla versione 4.4.2.

5.1.1 Analisi dei requisiti

In questa sezione verranno analizzati i casi d'uso dell'applicazione, gli scenari, e verrà costruito il modello del dominio.

5.1.2 Casi d'uso

I casi d'uso, come riportato in figura 5.1 sono tre. L'utente può attivare la radio Bluetooth del dispositivo che sta utilizzando, può scegliere da quale dispositivo ricevere i dati (la scelta sarà obbligata, e presupporrà l'utilizzo del dispositivo WristOx2, in ogni caso per un utilizzo futuro dell'applicazione, sarà anche possibile l'interazione con altri device) e visualizzare i parametri vitali.

5.1.3 Scenari

Nome	Scelta del dispositivo
Descrizione	L'utente effettua una scelta di quale dovrà essere il dispositivo con cui interfacciarsi e da cui ricevere i dati
Precondizioni	Il dispositivo scelto deve essere associato allo smartphone
Scenario principale	Il dispositivo compare in una lista di device selezionabili, in modo da poter essere utilizzato come sender dei dati da parte dell'utente

Nome	Attivazione del Bluetooth dello smartphone
Descrizione	L'utente attiva il Bluetooth dello smartphone, in modo da poter successivamente interagire con il dispositivo
Precondizioni	Lo smartphone deve supportare le funzionalità Bluetooth
Scenario principale	La radio Bluetooth del dispositivo viene attivata

Nome	Visualizzazione dei parametri vitali
Descrizione	L'utente visualizza sullo schermo dello smartphone il battito cardiaco e la saturazione del livello di ossigeno nel sangue misurati dal dispositivo
Precondizioni	Lo smartphone ed il dispositivo devono essere connessi fra di loro ed essere in grado di scambiare informazioni
Scenario principale	I dati appaiono sul display dello smartphone

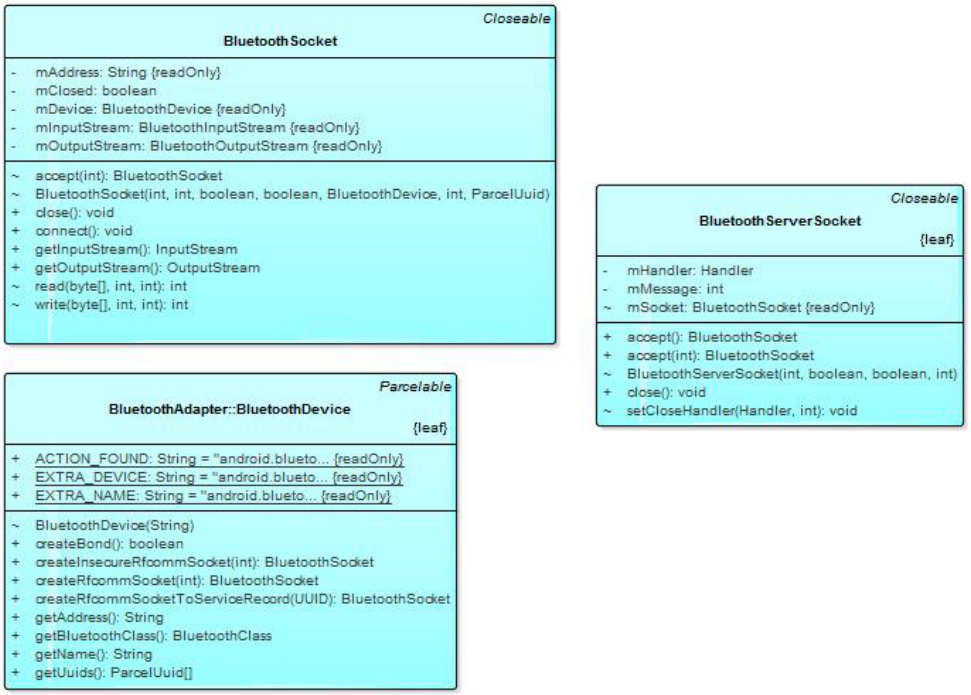


Figura 5.2: Modello del dominio

5.1.4 Modello del dominio

Il modello del dominio come si può osservare in figura 5.2 è formato dalle sole rappresentazioni delle socket e dei dispositivi Bluetooth (che rappresentano il WristOx2, il server, e lo smartphone, il client). La documentazione in dotazione con il dispositivo infatti, descrive come il WristOx2 sia del tutto assimilabile ad un server in attesa di richieste di connessione. Nell'ambito in cui viene sviluppata questa applicazione perciò, può essere considerato come un server Bluetooth, con la propria `BluetoothServerSocket`. La decisione di inserire già le rappresentazioni Android nel modello del dominio scaturisce dal fatto che questa è l'unica possibile soluzione adottabile (avendo a disposizione determinati device, ovvero lo smartphone e il WristOx2).

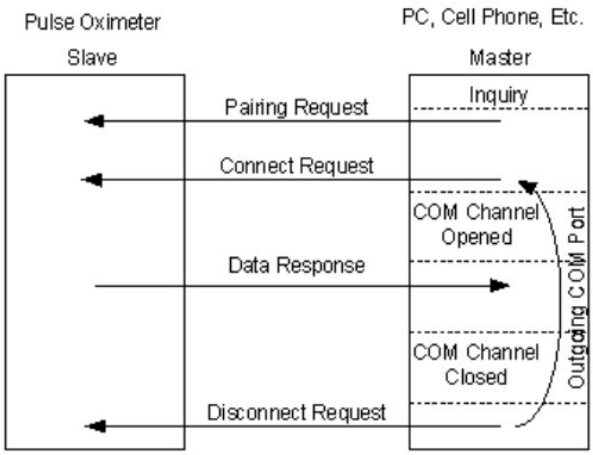


Figura 5.3: Comunicazione con il dispositivo

5.2 Analisi del problema

A questo punto risulta necessario analizzare il problema. Il dispositivo WristOx2 dovrà comunicare in qualche modo con lo smartphone, e dovrà farlo con l'unico protocollo di comunicazione comune ad entrambi, ovvero il Bluetooth. Per fare ciò è necessario modellare il WristOx2 come un oggetto di tipo `BluetoothDevice`, che possiede una socket in attesa di richieste di connessione, accettate un device alla volta. Dalla documentazione del dispositivo (figura 5.3) si capisce che la richiesta di connessione non è fatta dal WristOx2, ma deve essere lo smartphone, ricoprendo il ruolo di Master nella rete che si viene a creare, a richiedere una connessione. La richiesta di Pairing (associazione dei dispositivi) deve essere fatta una sola volta, fino a che il WristOx2 non viene associato ad un altro dispositivo, o dissociato dal dispositivo utilizzato fino a quel momento. La tecnologia richiesta per lo sviluppo dell'applicazione è Android, e questo limita nettamente la modellazione dei componenti, che devono rispettare le classi per il Bluetooth imposte dal sistema operativo.

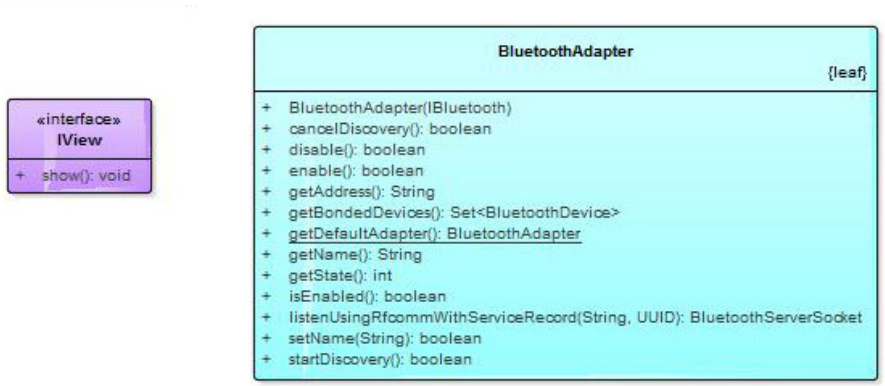


Figura 5.4: Nuove entità aggiunte

5.2.1 Struttura

É necessario, rispetto al modello del dominio, aggiungere due nuove entità. La prima, grazie al fatto di lavorare con la tecnologia Android, è già disponibile: si tratta di un componente che rappresenti la radio Bluetooth dello smartphone con cui si andrà a lavorare: `BluetoothAdapter`, mostrata in figura 5.4 insieme al componente grafico su cui visualizzare i dati.

5.2.2 Interazione

Nella figura 5.5 viene illustrato il diagramma di interazione dell'architettura logica, relativo alla lettura dei dati da parte dello smartphone.

5.2.3 Abstraction gap

Non esiste abstraction gap, dato che la tecnologia utilizzata è Android, ed esistono già tutte le funzionalità per supportare e progettare una connessione fra due dispositivi Bluetooth qualsiasi.

5.2.4 Struttura del pacchetto dati del WristOx2

Prima di proseguire nel progetto è necessario analizzare la struttura dei dati inviati dal WristOx2. La struttura è illustrata in figura 5.6.

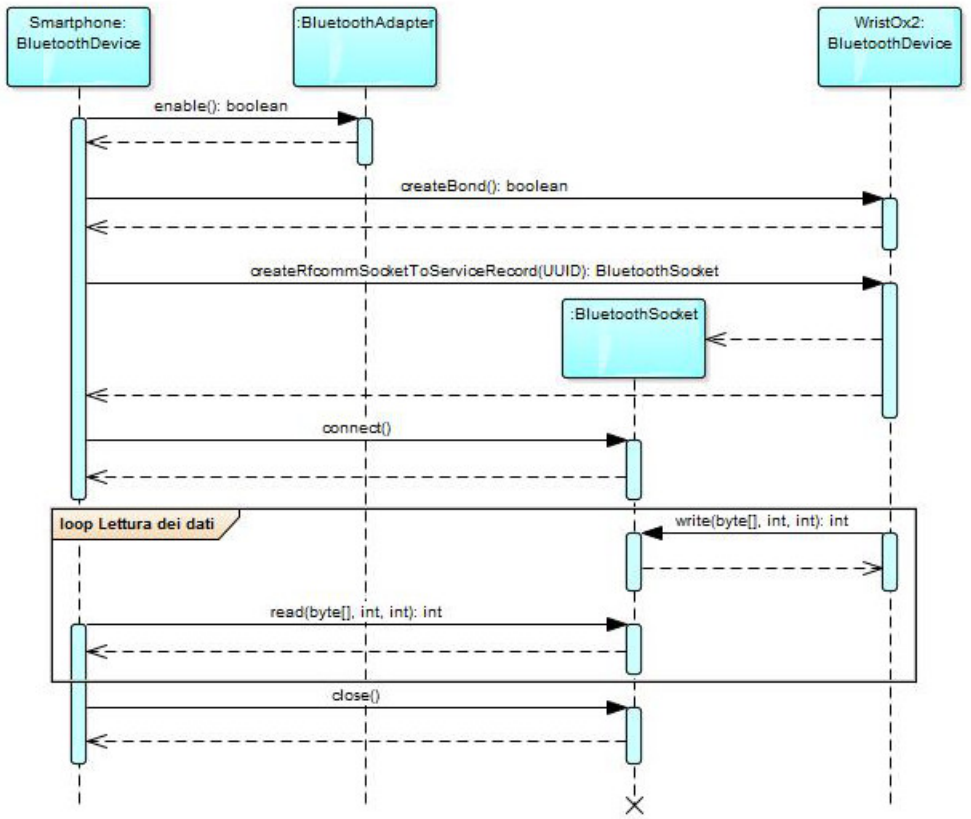


Figura 5.5: Architettura logica: diagramma di interazione

Packet	Frame				
	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
1	01	STATUS	PLETH	HR MSB	CHK
2	01	STATUS	PLETH	HR LSB	CHK
3	01	STATUS	PLETH	SpO_2	CHK
4	01	STATUS	PLETH	SREV	CHK
5	01	STATUS	PLETH	reserved	CHK
6	01	STATUS	PLETH	TMR MSB	CHK
7	01	STATUS	PLETH	TMR LSB	CHK
8	01	STATUS	PLETH	STAT2	CHK
9	01	STATUS	PLETH	SpO_2-D	CHK
10	01	STATUS	PLETH	SpO_2 Fast	CHK
11	01	STATUS	PLETH	SpO_2 B-B	CHK
12	01	STATUS	PLETH	reserved	CHK
13	01	STATUS	PLETH	reserved	CHK
14	01	STATUS	PLETH	E-HR MSB	CHK
15	01	STATUS	PLETH	E-HR LSB	CHK
16	01	STATUS	PLETH	E-SpO_2	CHK
17	01	STATUS	PLETH	E-SpO_2-D	CHK
18	01	STATUS	PLETH	reserved	CHK
19	01	STATUS	PLETH	reserved	CHK
20	01	STATUS	PLETH	HR-D MSB	CHK
21	01	STATUS	PLETH	HR-D LSB	CHK
22	01	STATUS	PLETH	E-HR-D MSB	CHK
23	01	STATUS	PLETH	E-HR-D LSB	CHK
24	01	STATUS	PLETH	reserved	CHK
25	01	STATUS	PLETH	reserved	CHK

Figura 5.6: Dati inviati dal WristOx2

I dati che interessano per quanto riguarda l'applicazione sono l'HR LSB per quanto riguarda il battito cardiaco, e l'SpO2 per quanto riguarda il livello di saturazione di emoglobina ossigenata nel sangue. Gli altri byte vengono utilizzati in questo caso per funzioni di sincronizzazione di ricezione dei pacchetti.

5.3 Progetto

5.3.1 Struttura

Le classi dell'applicazione saranno le seguenti, dovendo ora trasferire il progetto nell'ambiente Android, e dovendo perciò inserire `Activity` e GUI.

Come già accennato in precedenza nella sezione riguardante la programmazione in ambiente Android, tutte le computazioni bloccanti debbano essere effettuate in un flusso di controllo separato da quello principale. Proprio per questo motivo vengono creati oggetti che estendono la classe `AsyncTask`, una specie di thread mandato in esecuzione in modo asincrono, così da non bloccare l'`Activity` che lo utilizza. In questo caso l'oggetto creato è `ConnectAndUpdateTask`, responsabile della connessione e della lettura dei dati dalla `BluetoothSocket` ottenuta. Viene anche creato un listener per la gestione della pressione del nome di un determinato dispositivo Bluetooth da parte dell'utente, al momento della scelta. Andando ad analizzare tutti i componenti del progetto:

- `ConnectAndUpdateTask`: Come accennato poco sopra, è un thread asincrono che lavora sulla socket, andando ad effettuare una richiesta di `connect()` e gestendo poi la lettura dei dati e la loro successiva visualizzazione attraverso una GUI

- `MainActivity`: l'activity principale dell'applicazione, dalla quale è possibile selezionare un'immagine raffigurante il logo del protocollo Bluetooth per avviare la connessione al dispositivo e la ricezione dei dati

- `BondedDevicesActivity`: l'activity che viene visualizzata al momento della scelta del dispositivo. Attraverso una `ListView` è possibile per l'utente selezionare il dispositivo con cui intende lavorare

- `BondedDeviceClickListener`: il listener che serve a gestire il click di una determinata voce della `ListView` contenente i device disponibili

- `BluetoothActivity`: L'activity che permette di visualizzare i dati in arrivo dal WristOx2

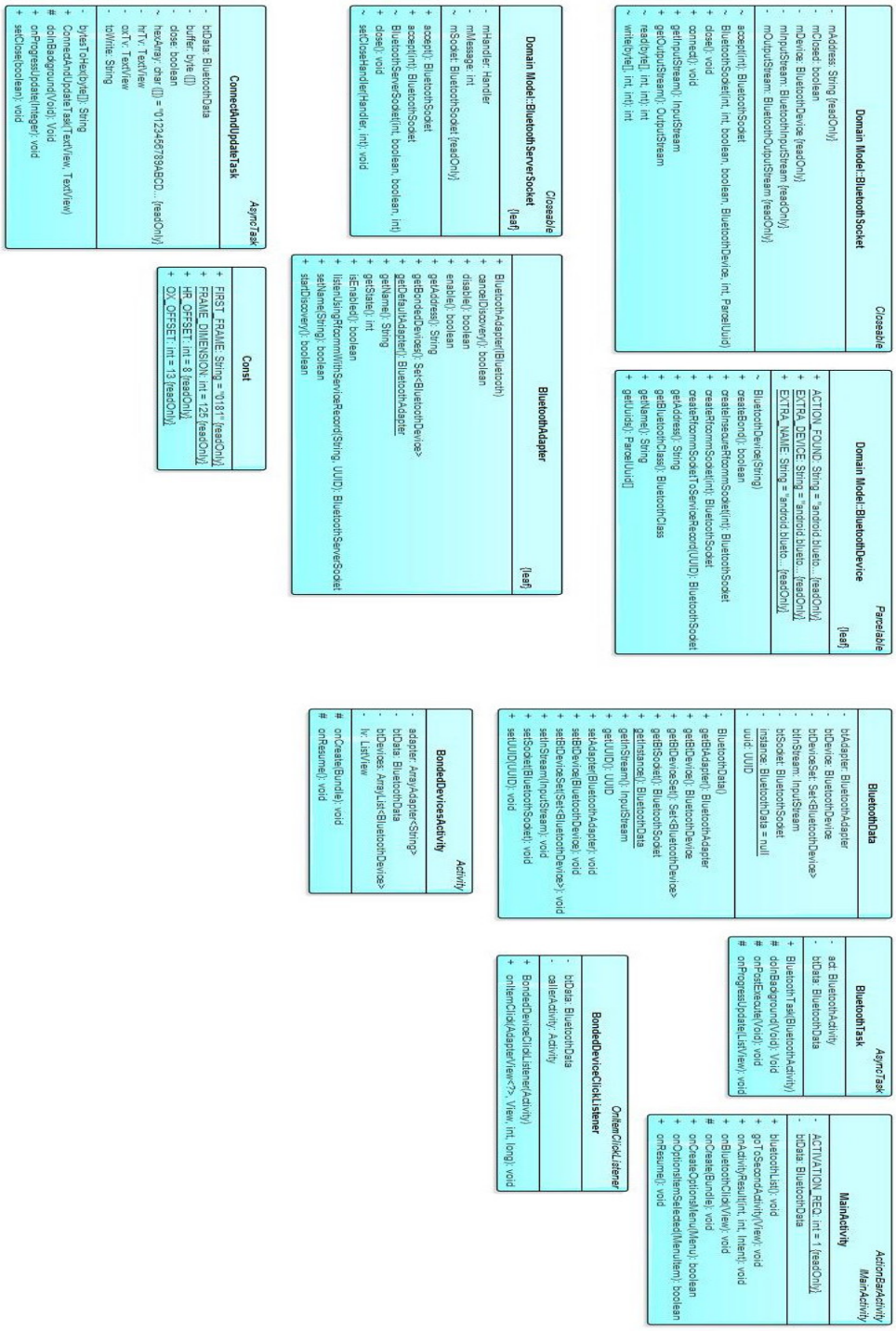


Figura 5.7: Struttura di progetto

5.3.2 Interazione

L'interazione di progetto, al fine di comprenderla meglio, viene divisa in tre sezioni. Come prima analisi, si tratta l'interazione che porta alla visualizzazione della lista dei device associati. Nella figura 5.8 viene mostrato come l'activity principale vada a recuperare un'istanza di `BluetoothData` poi poi settare l'adapter a quello principale, richiamando il metodo della classe `BluetoothAdapter`: `getDefaultAdapter()` . Fatto ciò, viene effettuato il controllo sullo stato di attivazione o meno del servizio di Bluetooth, e se trovato non attivo, l'activity lancia un `Intent` fornito da Android che permette di richiedere all'utente di attivare il servizio. Una volta che il servizio è stato attivato (o se è già attivo) viene lanciata la seconda activity.

La seconda parte dell'interazione prende in esame la visualizzazione della lista dei device associati allo smartphone. Il tutto viene effettuato attraverso una `ListView` a cui è associato un oggetto di tipo `ArrayAdapter`. Questo oggetto non è assolutamente correlato con l'adapter incontrato quando si parla della radio Bluetooth dei dispositivi, ma permette semplicemente di aggiornare dinamicamente e facilmente un oggetto di tipo `ListView`. L'interazione è disponibile in figura 5.9. Per andare con ordine, l'activity recupera anch'essa un'istanza di `BluetoothData`, in modo da avere gli stessi dati a disposizione di tutte le activity. Crea poi tre oggetti, un `ArrayList` di `BluetoothDevice`, una `ListView` (che in realtà non viene creata, ma viene recuperata tramite il suo id dalla classe che rappresenta le risorse), ed un `ArrayAdapter`, di cui si parlava in precedenza. Poi viene ottenuto il `Set` dei dispositivi associati attraverso il metodo `getBondedDevices()`, e viene depositato all'interno di `BluetoothData`. Il `Set` viene iterato, andando a recuperare tutti i device, che vengono poi mostrati nella `ListView` con l'aiuto dell'`ArrayAdapter` associato. Poi, attraverso l'utilizzo di un opportuno listener, viene deciso con quale dispositivo si andrà a lavorare in base alla selezione effettuata dall'utente, e viene lanciata l'activity successiva (per una questione di dimensione e sintesi, nel diagramma non è indicato l'`Intent` necessario affinché l'activity successiva sia lanciata).

La terza activity è quella che serve per leggere i dati in arrivo dal dispositivo. Oltre ad andare a recuperare la sua istanza dei dati Bluetooth, l'activity crea un oggetto di tipo `ConnectAndUpdateTask`, che vada ad effettuare le operazioni bloccanti come la `connect()` e la `read()`. Vengono così, dopo aver ottenuto la `BluetoothSocket` su cui leggere i dati, letti

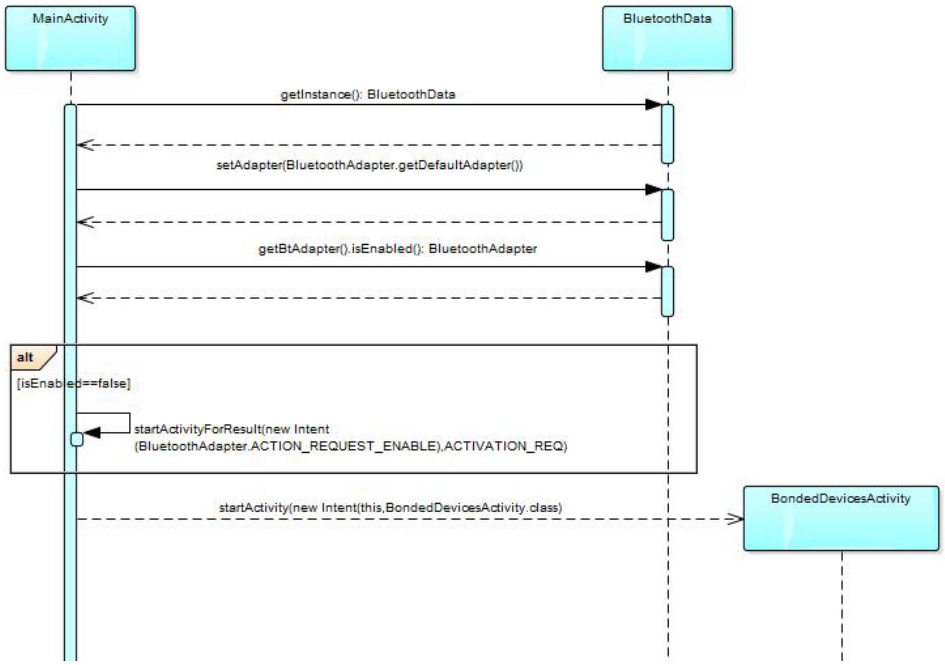


Figura 5.8: Attivazione Bluetooth

volta per volta i dati in arrivo dal WristOx2. Dopo una selezione sui dati (basata sui byte di sincronizzazione, l'activity visualizza, tramite il metodo `onProgressUpdate()` del task, in due `TextView` utilizzate a questo scopo, i dati di battito cardiaco e saturazione di emoglobina ossigenata nel sangue. Una volta che la lettura è terminata (in termini software, quando viene invocato il metodo `onPause()` sull'activity) la socket viene chiusa.

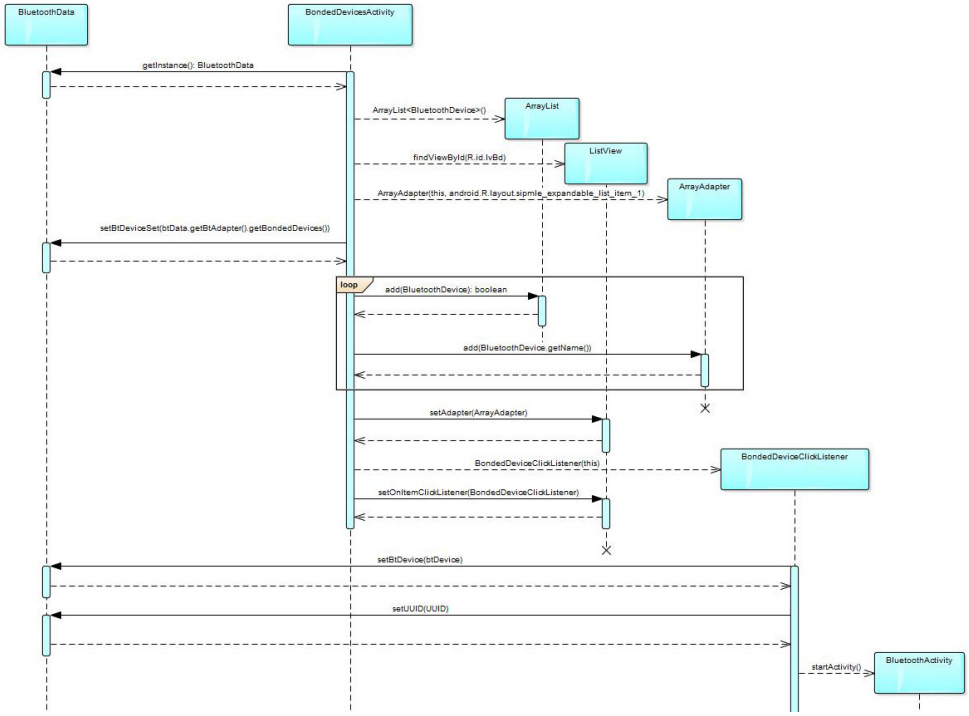


Figura 5.9: Visualizzazione della lista dei dispositivi e scelta del device da cui leggere i dati

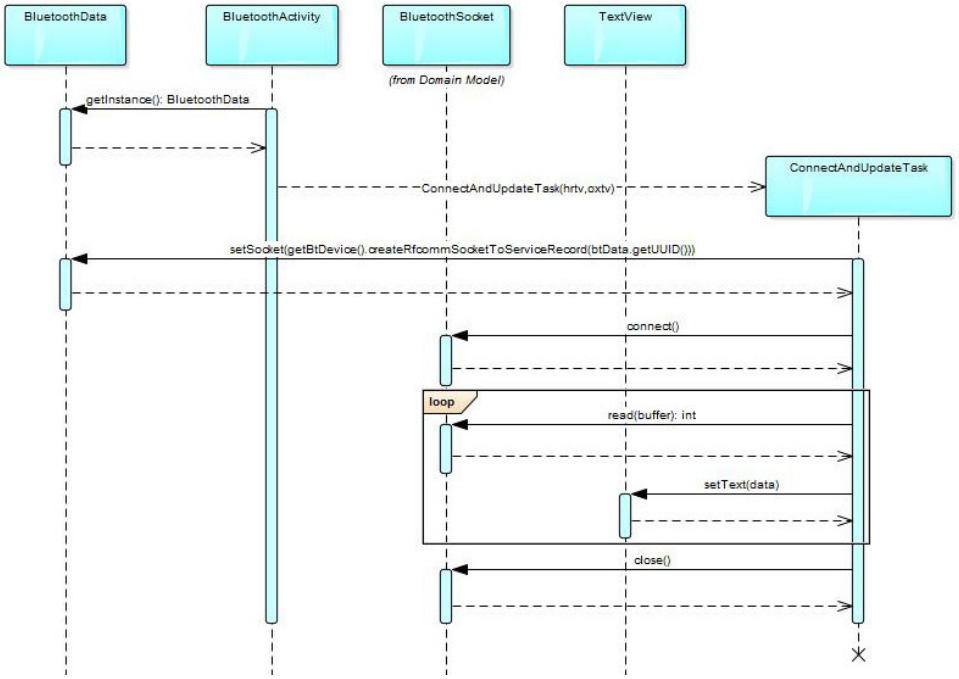


Figura 5.10: Lettura e visualizzazione dei dati

5.4 Implementazione

Al fine di una più semplice comprensione del software, viene fornita di seguito l'implementazione dei componenti fondamentali dell'applicazione.

`MainActivity`:

```
public class MainActivity extends ActionBarActivity implements
    IMainActivity{

   private BluetoothData btData;
   private static final int ACTIVATION_REQ = 1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btData = BluetoothData.getInstance();
    }

    public void onBluetoothClick(View v){
      ImageButton btn = (ImageButton)findViewById(R.id.btButton);
      btn.setImageResource(R.drawable.bluetooth_logo_gray);
      if (btData.getBtAdapter() == null) {
          Toast.makeText(this, "Il device non supporta il
              Bluetooth", Toast.LENGTH_LONG).show();
      }
      if (!btData.getBtAdapter().isEnabled()) {
          Intent enableBtIntent = new
              Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
          startActivityForResult(enableBtIntent, ACTIVATION_REQ);
      } else {
         bluetoothList();
      }
    }

    public void onResume(){
      super.onResume();
      btData.setAdapter(BluetoothAdapter.getDefaultAdapter());
      ImageButton btn = (ImageButton)findViewById(R.id.btButton);
      btn.setImageResource(R.drawable.bluetooth_logo);
```

```
    }

    @Override
    public void onActivityResult(int requestCode, int resultCode,
        Intent data){
      if(requestCode==ACTIVATION_REQ && resultCode==RESULT_OK){
         bluetoothList();
      }
    }

    public void bluetoothList(){
      Intent i = new Intent(this,BondedDevicesActivity.class);
      startActivity(i);
    }

}
```

BondedDevicesActivity:

```
public class BondedDevicesActivity extends Activity{

   private ListView lv;
   private ArrayList<BluetoothDevice> btDevices;
   private ArrayAdapter<String> adapter;
   private BluetoothData btData;

   @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.bd_layout);
        btData = BluetoothData.getInstance();
        btDevices = new ArrayList<BluetoothDevice>();
    }

   @Override
    protected void onResume() {
      super.onResume();
      btData.setAdapter(BluetoothAdapter.getDefaultAdapter());
      lv = (ListView)findViewById(R.id.lvBd);
      adapter=new ArrayAdapter<String>(this,
```

```
            android.R.layout.simple_expandable_list_item_1);
        btData.setBtDeviceSet(btData.getBtAdapter().getBondedDevices());
        for(BluetoothDevice bt : btData.getBtDeviceSet()){
            adapter.add(bt.getName());
            btDevices.add(bt);
        }
        lv.setAdapter(adapter);
        lv.setOnItemClickListener(new
            BondedDeviceClickListener(this));
    }
}
```

BondedDeviceClickListener:

```
public class BondedDeviceClickListener implements
    OnItemClickListener{

   private BluetoothData btData;
   private Activity callerActivity;

   public BondedDeviceClickListener(Activity caller){
      btData = BluetoothData.getInstance();
      callerActivity = caller;
   }

   @Override
   public void onItemClick(AdapterView<?> parent, View view, int
       position, long id){
      for(BluetoothDevice bt: btData.getBtDeviceSet()){
         if(bt.getName().equals((String)parent.getItemAtPosition(position))){
            btData.setUUID(bt.getUuids()[0].getUuid());
            btData.setBtDevice(bt);
         }
      }
      Intent startMonitoring = new
          Intent(callerActivity,BluetoothActivity.class);
      callerActivity.startActivity(startMonitoring);
   }

}
```

BluetoothActivity:

```
public class BluetoothActivity extends Activity implements
    IBluetoothActivity{
    private BluetoothData btData;
    private ConnectAndUpdateTask updateTask;

   @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.bt_layout);
    }

   @Override
   protected void onResume(){
      super.onResume();
        this.updateTask = new
            ConnectAndUpdateTask((TextView)findViewById(R.id.hrText),
         (TextView)findViewById(R.id.ox_text));
        btData = BluetoothData.getInstance();
       connectAndUpdate();
   }

   protected void onPause(){
      try{
         super.onPause();
         this.updateTask.setClose(true);
      } catch(Exception e){
      }
   }

   public void connectAndUpdate(){
      this.updateTask.execute();
   }
}
```

ConnectAndUpdateTask:

```
public class ConnectAndUpdateTask extends
    AsyncTask<Void,Integer,Void>{

   private TextView hrTv,oxTv;
   private BluetoothData btData;
   private byte[] buffer;
   private String toWrite;
   private boolean close;

   public ConnectAndUpdateTask(TextView hrTv, TextView oxTv){
      this.hrTv=hrTv;
      this.oxTv=oxTv;
      btData=BluetoothData.getInstance();
      buffer = new byte[Const.FRAME_DIMENSION];
      toWrite=null;
      close=false;
   }

   @Override
   protected Void doInBackground(Void... params) {
      BluetoothSocket tmp = null;
        try {
            tmp = btData.getBtDevice()
               .createRfcommSocketToServiceRecord(btData.getUUID());
            btData.setSocket(tmp);
        } catch (Exception e) { }
         btData.getBtAdapter().cancelDiscovery();
        try {
            btData.getBtSocket().connect();
        } catch (IOException connectException) {
            try {
                btData.getBtSocket().close();
            } catch (IOException closeException) { }
        }
      try {
         this.btData.setInStream(btData.getBtSocket().getInputStream());
      } catch (IOException e1) {
         e1.printStackTrace();
      }
```

```
    while(!this.close){
          try {
              btData.getInStream().read(buffer,0,Const.PACKET_DIMENSION);
          toWrite = bytesToHex(buffer);
          int heartRate,oxygenSat;
          int index = toWrite.indexOf(Const.FIRST_FRAME);
          heartRate=Integer.parseInt("" +
              toWrite.charAt(Const.HR_OFFSET*2+index) +
              toWrite.charAt(Const.HR_OFFSET*2+1+index), 16);
          oxygenSat=Integer.parseInt("" +
              toWrite.charAt(Const.OX_OFFSET*2+index) +
              toWrite.charAt(Const.OX_OFFSET*2+1+index), 16);
          if((index+27)<toWrite.length() &&
              toWrite.substring(index+10,
              index+14).equals(Const.NOT_FIRST_FRAME) &&
              toWrite.substring(index+20,
              index+24).equals(Const.NOT_FIRST_FRAME)){
             publishProgress(heartRate,oxygenSat,buffer.length);
          }
          } catch (Exception e) {
          }
    }
    try{
       btData.getBtSocket().close();
    } catch(Exception e){
       e.printStackTrace();
    }
    return null;
}

@Override
public void onProgressUpdate(Integer...integers){
    hrTv.setText(""+integers[0]+" bpm");
    if(integers[1]>0 && integers[1]<100){
       oxTv.setText(""+integers[1]+"%");
    }
}

final char[] hexArray = "0123456789ABCDEF".toCharArray();
```

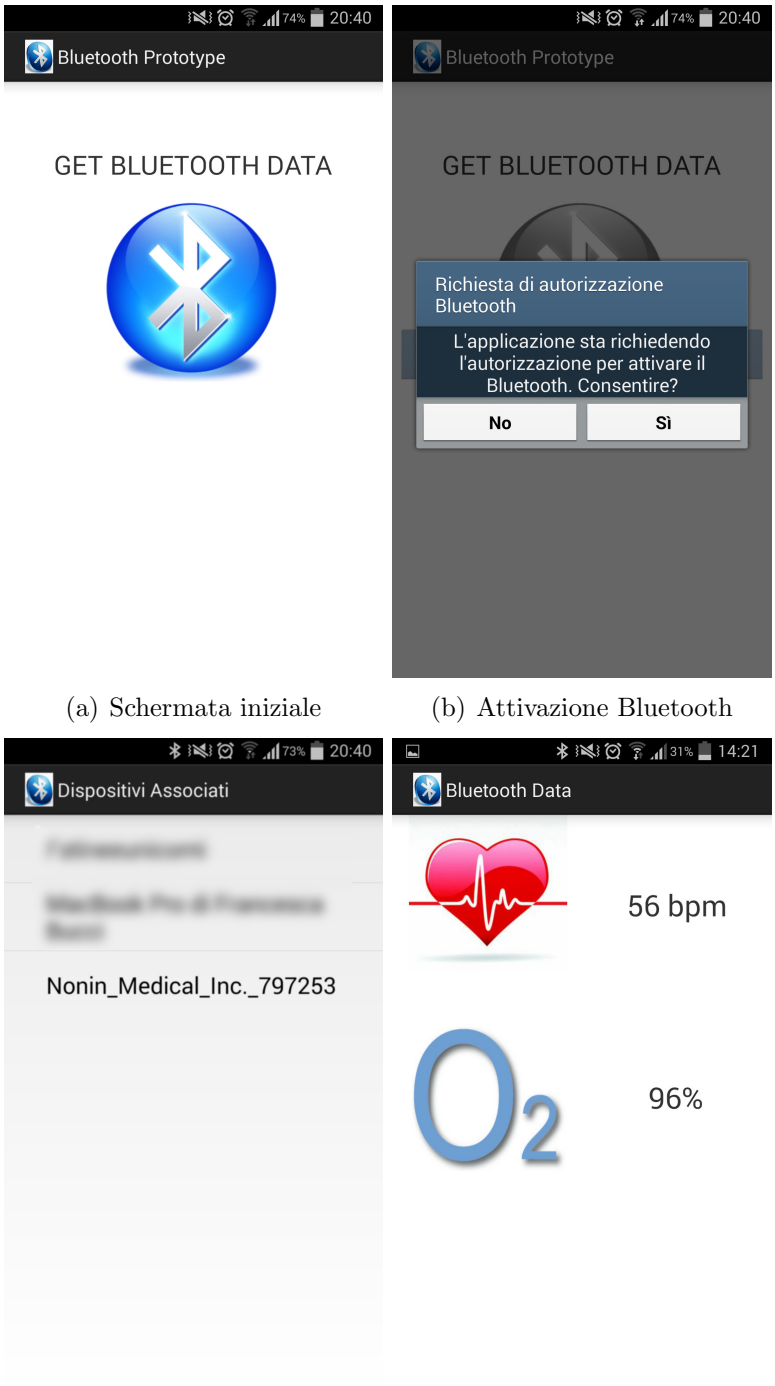
```
   private String bytesToHex(byte[] bytes) {
      char[] hexChars = new char[bytes.length * 2];
      for (int j = 0; j < bytes.length; j++) {
      int v = bytes[j] & 0xFF;
      hexChars[j * 2] = hexArray[v >>> 4];
      hexChars[j * 2 + 1] = hexArray[v & 0x0F];
      }
      return new String(hexChars);
   }

   public void setClose(boolean todo){
      this.close=todo;
   }

}
```

5.5 Test

I test che vengono effettuati confermano che l'applicazione riesce a supportare l'invio di una considerevole quantità di dati, in ogni caso sufficienti per considerare la visualizzazione dei dati in real-time. Per quanto riguarda il primo test, l'applicazione viene lanciata quando il servizio di Bluetooth dello smartphone è disattivato, il risultato (quello atteso) è che viene richiesta all'utente l'attivazione come si può osservare in figura 5.11.

Il secondo test viene effettuato uscendo dalla schermata di visualizzazione dei dati e riaprendola: l'applicazione, dopo aver chiuso la socket, si rimette correttamente in ascolto dei dati.

Il terzo test riguarda la scelta del dispositivo. Viene correttamente visualizzata la lista dei dispositivi associati, ovviamente contenente anche il WristOx2, che può essere così selezionato, in modo da cominciare l'interazione.



(a) Schermata iniziale (b) Attivazione Bluetooth

(c) Lista dei dispositivi (d) Visualizzazione dei dati

Figura 5.11: Test effettuati

5.6 Estendibilità

L'applicazione è molto estendibile per diverse ragioni. La prima è che si basa sul sistema operativo Android, che ha una struttura che permette facilmente di disaccoppiare la parte grafica da quella logica, e permette, meccanismo che è stato utilizzato in questa applicazione, di eseguire operazioni technology-dependent (come la lettura dei dati dal WristOx2) attraverso thread separati. L'applicazione ad esempio, potrebbe essere estesa a qualsiasi altro dispositivo semplicemente definendo una nuova activity ed uno o più task responsabili della lettura dei dati.

Oltre ad essere fortemente estendibile, è anche compatibile sulla maggior parte dei dispositivi basati su Android. É stata infatti testata con successo su un Nexus 7, sfruttandola per l'integrazione con un altro progetto.

L'obiettivo è quello infatti di sfruttare lo smartphone utilizzato come il PS (personal Server) della BAN, facendo in modo che i dati possano poi essere inviati ad altri componenti software che ne richiedano l'utilizzo.

Capitolo 6

Conclusione

6.1 Considerazioni finali

Giunti al termine di questo elaborato, diventa opportuno riassumere brevemente l'elenco dei concetti assimilati ed affrontati durante la stesura di questa tesi. Partendo dall'inizio, l'impressione avuta durante la ricerca ed il lavoro sul materiale è quella che, non solo il mondo del wearable, ma in generale quello dell'informatica, sia in costante evoluzione ed in continuo cambiamento. Questo presuppone un aggiornamento personale costante su quelle che sono le nuove tendenze per quanto riguarda le varie tecnologie. Nello specifico campo del wearable, c'è la seria possibilità che i nuovi device (come i Glass) migliorino la vita di chi li possiede, aprendo nuovi orizzonti nello sviluppo software e hardware. Le Body Area Network sono già importantissime, soprattutto a livello medico, e permettono un'interazione ed un controllo del paziente a distanza che prima di oggi erano impensabili, lasciando comunque aperte le porte di ulteriori evoluzioni. Personalmente non è viva in me l'idea forte avanzata da qualcuno, che il costante e pervasivo sviluppo di questo tipo di reti trasformerà di qui a breve gli uomini in cyborg, macchine incapaci di pensare ma semplicemente di compiere azioni. Questo sviluppo verrà certamente aiutato da Android, un sistema operativo che come si è osservato nelle pagine precedenti, presenta scenari di evoluzione potenzialmente infiniti, in innumerevoli campi di sviluppo. Concludendo, non si può con certezza affermare dove arriverà l'evoluzione di queste tecnologie, la sfida per il futuro rimane quella di ampliare ancora questo orizzonte.

6.2 Ringraziamenti

Durante questi tre anni ci sono stati diversi tipi di momenti. Alcuni belli, altri meno, ma ciò non mi ha impedito di arrivare al termine di questo percorso triennale, e di acquisire, grazie ad esso, un numero grandissimo di conoscenze ed abilità in molti ambiti, spaziando ovviamente dall'informatica alla fisica (come è ovvio che sia), ed arrivando anche ad essere più determinato e sicuro nell'affrontare le sfide che la vita mi porta, e mi continuerà a portare, ad affrontare.

Sicuramente non avrei mai concluso questo percorso da solo, con le mie uniche forze. Tantissime persone hanno contribuito a formarmi, motivarmi o anche più semplicemente farmi divertire, per far sì che non si riducesse tutto ad un mera questione didattica, basata semplicemente su un voto.

Vorrei perciò ringraziare queste persone, a partire dai professori, tutti quelli che mi hanno fatto capire quanto sia importante possedere un'abilità o una conoscenza, ma quanto sia ancora più importante riuscire a condividerla con gli altri, trasmettendomi la passione di sapere e saper insegnare.

Vorrei ringraziare il Professor Alessandro Ricci, che è diventato, da quando ho affrontato il suo corso, il mio punto di riferimento all'interno della facoltà. Ho intrapreso con lui un percorso di tirocinio e un altro che mi ha portato alla stesura di questa tesi. Da lui ho imparato ad avere sete di conoscenza, a non fermarmi alla prima difficoltà, poiché nulla è impossibile, anche quando lo sembra all'inizio, che con lo studio e la pratica (preceduti da un'attenta analisi) si può ottenere ciò che si vuole.

Vorrei ringraziare tutti i miei amici, compagni di facoltà e non, che mi hanno aiutato in questi tre anni a non sentire il peso dei pomeriggi passati a studiare, perché sapevo che quando avrei finito loro per me ci sarebbero stati, per uscire o anche semplicemente per fare due chiacchiere, per staccare la spina.

Vorrei ringraziare i miei genitori, che hanno sempre creduto in me e continueranno a farlo, e io spero di continuare a dimostrargli che la loro fiducia è ben riposta.

Infine, vorrei ringraziare la persona che ritengo sia stata la più importante in questi tre anni (non me ne vogliano gli altri). Questa persona è la mia ragazza, Francesca, che in ogni istante mi ha sempre sostenuto, in qualunque scelta io facessi. Durante la fine del mio percorso mi è stata ancora più vicino, quando sembrava che terminare questi tre anni fosse impossibile. Ed ora che questi tre anni sono finiti, cerco, come ho sempre fatto, di ripagarla allo stesso modo, cercando di aiutarla, di motivarla e di sostenerla in qualsiasi scelta lei faccia o vorrà fare in futuro.

Grazie a tutti,

Edoardo

Bibliografia

- Mann, Steve - Wearable Computing. In: Soegaard, Mads and Dam, Rikke Friis (eds.). The Encyclopedia of Human-Computer Interaction, 2nd Ed.. Aarhus, Denmark: The Interaction Design Foundation. Available online at `https://www.interaction-design.org/encyclopedia/wearable_computing.html` - 2013
- Min Chen, Sergio Gonzalez, Athanasios Vasilakos, Huasong Cao, Victor C. M. Leung - Body Area Networks: A Survey - Springer Science, 2010
- Mark A. Hanson, Harry C. Powell Jr., Adam T. Barth, Kyle Ringgenberg, Benton H. Calhoun, James H. Aylor, John Lach - Body Area Sensor Networks: challenges and opportunities - IEEE Computer Society, 2009
- Matti Siekkinen, Markus Hiienkari, Jukka K. Nurminen, Johanna Nieminen - How Low Energy is Bluetooth Low Energy? Comparative Measurements with ZigBee/802.15.4 - 2012
- Bluetooth Website - `http://bluetooth.com/`, 2014
- Tutorial sui Nodi Wireless, `http://www.ing.unibs.it/~wsnlab/download/zigbee/sisti/WSNlab_tutorial04_ZigBee.pdf`
- Android Developer Website - `http://developer.android.com/`, 2014
- Pagina ufficiale del prodotto WristOx2, `http://www.nonin.com/OEMSolutions/WristOx23150-OEM`