POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

Migliorare l'assistenza in strutture per disabili: una soluzione IoT

Relatori
Fulvio Corno
Luigi De Russis

Candidato
Monge Roffarello Alberto

Ottobre 2015

Indice

Elenco delle tabelle

Elenco delle figure

Capitolo 1

Introduzione

1.1 Motivazioni

L'Organizzazione Mondiale della Sanità definisce la disabilità come:
"Termine generale che comprende handicap, limitazione nelle attività e restrizione nella partecipazione. Un handicap è un problema relativo ad una funzione o ad una struttura del corpo; limitazione nelle attività è una difficoltà che il soggetto incontra nell'eseguire un compito o un'azione; restrizione nella partecipazione è un problema che il soggetto incontra nell'essere coinvolto pienamente nelle situazioni della vita. La disabilità quindi è un fenomeno complesso, che riflette l'interazione fra il corpo della persona e la società in cui la persona vive".[1]
La persona che ne soffre ha una ridotta capacità d'interazione con l'ambiente sociale rispetto a ciò che è considerata la norma, pertanto è meno autonomo nello svolgere le attività quotidiane e spesso si trova in condizioni di svantaggio nel partecipare alla vita sociale. Il concetto di disabilità è cambiato nel tempo e secondo la nuova classificazione dell'Organizzazione Mondiale della Sanità [2] (approvata da quasi tutte le nazioni afferenti all'ONU) è diventato un termine ombrello che identifica le varie difficoltà a cui un disabile è sottoposto. La classificazione considera la disabilità come il risultato dell'interazione fra la salute dell'individuo e fattori legati al contesto, come lo stato di salute e l'ambiente circostante. I disabili possono essere di vario tipo e una prima suddivisione può essere fatta tra disabilità motoria, sensoriale e psichica. La prima caratterizza qualsiasi problematica che limiti la funzione fisica degli arti, diminuendo la capacità motoria dell'individuo. I problemi sensoriali invece sono quelli che compromettono uno o più dei cinque sensi, principalmente la vista e l'udito. L'ultima tipologia di disabilità indicata rientra invece in una classe più ampia, quella dei problemi intellettivi, che possono comparire a qualsiasi età della persona. La disabilità intellettiva è un concetto che spazia dal ritardo mentale al deficit cognitivo più lieve e specifico. Un disturbo mentale, chiamato anche

disturbo psichiatrico, è un'affezione patologica che colpisce la sfera cognitiva, affettiva, comportamentale o relazionale di una persona in modo sufficientemente forte da rendere problematica la sua integrazione nella società. Quando il disturbo diventa particolarmente importante si parla spesso di malattia mentale.
La stima del numero delle persone disabili nel mondo è molto complicata, anche perché il termine stesso è molto generico e può racchiudere a seconda delle interpretazioni persone diverse. Nonostante questo i demografi concordano sul fatto che la popolazione mondiale di persone con disabilità è molto grande: nel 2012 l'Organizzazione Mondiale della Sanità ha stimato una popolazione mondiale di 6,5 miliardi di persone di cui quasi 650 milioni, ovvero il 10%, sono stati stimati essere moderatamente o gravemente disabili [3].
Esistono due visioni che cercano di affrontare il problema della disabilità in generale. Il modello medico vede la disabilità come un problema della persona, direttamente causata da malattie, traumi o altre condizioni di salute e che richiede quindi assistenza medica individuale da parte di professionisti. Il focus è sulla singola persona e l'obiettivo è quello di "agire" su di essa per "modificarla" in modo da ovviare alla sue limitazioni. Il modello sociale invece vede il tema della disabilità come un problema di comunità: secondo questo approccio è la società che con le sue discriminazioni e i suoi pregiudizi rende le persone disabili. Esso pone la questione della piena integrazione degli individui disabili nel mondo, focalizzandosi sulle barriere fisiche o sociali che essi sperimentano. La gestione del problema richiede un'azione sociale, ed è responsabilità collettiva apportare le modifiche ambientali necessarie per permettere l'integrazione. Da questo punto di vista la parità di accesso alle strutture e ai mezzi di trasporto per una persona disabile è una questione di diritti umani di grande rilevanza. L'obiettivo è quello di sviluppare una tecnologia assistiva per i disabili, ovvero prodotti e servizi per migliorare la loro autonomia. Nonostante la discriminazione sia un cancro sociale difficilmente estirpabile, con il tempo la tutela del disabile è aumentata. Grazie anche alla dichiarazione di Madrid del 2002 [4], che sposta l'asse di interesse da una visione eminentemente medico - scientifica ad una prettamente sociale, e alla convenzione ONU sui diritti delle persone con disabilità [5], che richiama esplicitamente diversi principi della Dichiarazione Universale dei Diritti Umani, negli ultimi anni l'attenzione verso questo tema è aumentata e si stanno facendo passi avanti verso un più equo trattamento di queste persone. L'obiettivo è promuovere condizioni di vita dignitose e un sistema di relazioni soddisfacenti nei riguardi di persone che presentano difficoltà nella propria autonomia personale e sociale, in modo che esse possano sentirsi parte di comunità e di contesti relazionali dove poter agire, scegliere, giocare, studiare, lavorare e vedere riconosciuto il proprio ruolo e la propria identità. In questo campo gli ingegneri e progettisti hanno due principali aree di intervento e responsabilità:

- "Design for all": progettare e costruire dispositivi ed ambienti accessibili utilizzabili dal maggior numero possibile di utenti, indipendentemente dalle loro

caratteristiche (inclusi quindi i disabili).

- Progetto di "tecnologie assistive": progettare e costruire dispositivi che permettano di ovviare a barriere fisiche o sociali già esistenti, estendendo opportunità e utilizzabilità ai disabili.

Nell'ambito delle tecnologie assistive per le persone con problemi psichici e psicofisici si inizia a parlare di "Ambient Assisted Living AAL" (AAL), termine coniato nei primi anni 2000 per descrivere un insieme di soluzioni tecnologiche destinate a rendere attivo, intelligente e cooperativo l'ambiente nel quale viviamo, efficace nel sostenere la vita indipendente, capace di fornire maggiore sicurezza nello svolgimento delle attività di tutti i giorni. Il ruolo degli ausili (in senso lato) è aiutare la persona a superare le proprie difficoltà, compensando o evitando le carenze funzionali. Gli ausili possono essere di varia natura: servizi di assistenza personale, aiuti da parte di animali, medicinali, strumenti utilizzati dal personale sanitario e molto altro. Un "Assistive Product", secondo la norma norma ISO 9999:2011, è un qualunque prodotto (inclusi dispositivi, equipaggiamento, strumenti hardware e software) per prevenire, compensare, monitorare, rilevare o neutralizzare menomazioni e limitazioni dell'attività di una persona. L'utilizzo della tecnologia porta notevoli vantaggi: essa può essere impiegata in modo non intrusivo per rilevare segnali fisiologici delle persone e può permettere il monitoraggio continuo dei pazienti, anche a casa loro, diminuendo di fatto i costi necessari. Gli studi in ambito AAL si concentrano principalmente in due direzioni: da un lato si cerca di migliorare la qualità della vita casalinga per chi ha bisogno di attenzioni continue, dall'altro si cerca di supportare chi deve prendersi cura di queste persone negli ospedali. Tuttavia pochi studi sono stati svolti per supportare gli assistenti sanitari che si prendono cura di persone con disabilità mentali e fisiche in strutture abitative apposite. In Italia e nel mondo queste residenze sono molto comuni, e offrono una valida alternativa per chi non ha la possibilità di essere curato o accudito a casa propria. Non tutti i disabili hanno infatti una famiglia alle spalle, e pur avendola non tutte hanno la possibilità di fornire il giusto supporto per l'assistenza casalinga: si pensi ad esempio a una persona sola che è vittima di una disabilità acquisita a causa di un incidente. D'altra parte le strutture ospedaliere non potrebbero accogliere tutte queste figure, e offrirebbero comunque un tipo di permanenza che mal si adatterebbe con le necessità di queste persone.

1.2 Le RAF e l'assistenza ai disabili

Le Residenze Sanitarie Assistenziali (RSA), introdotte in Italia a metà degli anni novanta, sono strutture non ospedaliere, ma comunque a impronta sanitaria, che ospitano per un periodo variabile da poche settimane al tempo indeterminato persone non autosufficienti, che non possono essere assistite in casa o che necessitano

di specifiche cure mediche di più specialisti nonchè di una articolata assistenza sanitaria. Si distinguono dagli ospedali e dalle case di cura, rivolti ai pazienti sofferenti di una patologia acuta, e dalle case di riposo, destinate agli anziani almeno parzialmente autosufficienti. Una tipologia di RSA che spesso è specializzata nell'accogliere persone con disabilità è la Residenza Assistenziale Flessibile (RAF): strutture che forniscono assistenza e cura a persone con disabilità psichica e psicofisica grave, in alternativa o in sostituzione alla famiglia, garantendo prestazioni sanitarie e riabilitative e attività di potenziamento e mantenimento delle capacità del disabile. Queste residenze sono gestite molto spesso da cooperative sociali, particolari tipi di società cooperativa che gestiscono servizi socio-sanitari ed educativi, oppure attività di vario genere finalizzate all'inserimento nel mercato del lavoro di persone svantaggiate. La tesi si avvale della collaborazione della Cooperativa Sociale P.G. Frassati, che gestisce diverse strutture di questo tipo nella zona di Torino e con il cui contributo è stato possibile capire di più su queste realtà. In particolare in queste residenze vengono ospitate quelle persone per le quali la vita totalmente indipendente non è possibile, ma che non hanno bisogno di cure 24 ore al giorno e sono ancora troppo giovani per vivere in una casa di riposo. In questi luoghi, a differenza degli ospedali, non ci sono medici o infermieri che lavorano a tempo pieno, ma operatori sanitari che assicurano la cura e il supporto necessario agli ospiti nel compiere quelle azioni caratterizzanti la loro vita quotidiana: la cucina, la pulizia della casa, gli hobby, le passioni comuni e molto altro. Viene promossa l'indipendenza e la dignità di una vita che deve assomigliare a quella che vivrebbero nella società comune: per questo motivo viene lasciato agli ospiti un certo grado di libertà. Le strutture di solito comprendono le camere per gli ospiti e alcuni ambienti in comune, come la cucina e le sale svago. Gli ospiti delle RAF possono richiedere assistenza chiamando gli operatori sanitari con la voce oppure utilizzando appositi pulsanti presenti almeno in ogni camera e nei bagni, come previsto dalla legge italiana.
Nonostante il regime di semi indipendenza, il controllo dei pazienti non deve comunque mancare: sono infatti comuni situazioni critiche che si possono verificare accogliendo persone con problemi mentali e fisici, come attacchi epilettici, cadute e problemi di salute di varia natura. L'attenzione deve essere alta anche per il fatto che spesso gli ospiti si possono muovere liberamente nella struttura, e addirittura in certi casi possono lasciare temporaneamente la RAF. Le strutture, inoltre, si possono differenziare in base alle persone che ospitano: i disabili con problemi psichici sono relativamente autosufficienti, in molti casi sono inseriti nella società e hanno un lavoro che svolgono quotidianamente, mentre i disabili motori hanno bisogno di aiuto anche nelle piccole azioni di ogni giorno. D'altro canto i disabili mentali possono soffrire di attacchi epilettici, e gli assistenti devono continuamente controllarli per essere pronti ad intervenire fornendo le cure necessarie. Esistono tuttavia elementi comuni a tutte le residenze gestite dalla cooperativa Frassati: durante il giorno vi

sono due assistenti presenti, durante la notte uno solo, un infermiere è disponibile per un'ora al giorno mentre un dottore è disponibile solo su richiesta. Questo comporta che la rilevazione di situazioni di pericolo per i pazienti dev'essere la più affidabile e veloce possibile, in modo da avere il tempo di richiedere un'eventuale assistenza esterna. Uno dei principali limiti a cui al giorno d'oggi queste residenze sono sottoposte è che il monitoraggio continuo degli inquilini non è possibile: il compito di rilevare situazioni di rischio è lasciato agli operatori di turno o al sistema di allarme manuale che in taluni casi può non essere usufruibile dagli ospiti. Questo va in contrasto con la preoccupazione più grande per chi deve garantire tutte le cure necessarie agli ospiti di una RAF, quella di fornire assistenza ogni qual volta ve ne sia bisogno. Allo stato attuale esistono casi in cui questo non è possibile. Si pensi ad esempio al caso in cui una persona sia in preda a un attacco epilettico e non sia in grado di richiedere aiuto: è molto probabile che questo tipo di problema non sia rilevato in un tempo sufficientemente breve, con tutte le complicazioni e le problematiche che ne derivano.

1.3 Struttura della tesi

Nei prossimi capitoli verranno approfonditi gli aspetti di questa tesi, partendo dalle motivazioni che hanno spinto l'intraprendere di questo lavoro fino alla descrizione del sistema che si è scelto di progettare e implementare per andare incontro alle esigenze riscontrate.

- Nel capitolo 2 *"Requisiti e obiettivi"*, partendo dai risultati ottenuti dal confronto con gli operatori sanitari della cooperativa P.G. Frassati si estrarranno i requisiti e gli obiettivi che questa tesi si prefigge di raggiungere.
- Nel capitolo 3 *"I componenti del progetto"* verranno elencati i dispositivi, i linguaggi di programmazione e i framework adottati che servono per rendere concreto un sistema di aiuto per gli operatori sanitari che lavorano nelle RAF.
- Nel capitolo 4 *"Progettazione"* verrà esposto il progetto del sistema, comprendente l'architettura, le sue parti principali, e il sistema di comunicazione adottato tra i diversi dispositivi.
- Nel capitolo 5 *"Implementazione"* l'attenzione verrà posta sulla parte concreta della tesi, descrivendo le implementazioni delle varie parti del progetto.
- Nel capitolo 6 *"Test con utenti reali"* si descriveranno le modalità con cui il progetto è stato testato.
- Nel capitolo 7 *"Risultati dei test e obiettivi raggiunti"* si analizzeranno i test svolti, andando a confrontare gli obiettivi raggiunti con quelli che si erano prefissati a inizio lavori.

- Nel capitolo 8 *"Conclusioni e sviluppi futuri"* si andrà ad analizzare la riuscita del progetto, andando ad indicare gli sviluppi futuri che il prototipo sviluppato dovrà sicuramente avere per un ancor più efficace utilizzo.

Capitolo 2

Requisiti e obiettivi

In questo capitolo si vogliono definire i requisiti e gli obiettivi che questa tesi si prefigge di raggiungere e da dove essi nascono. Gli obiettivi verranno anche schematizzati in una tabella riassuntiva per essere poi facilmente confrontati con i risultati dei test condotti. In questo modo si avrà una visione complessiva della riuscita o meno del progetto, da cui anche eventuali sviluppi futuri potranno trarne vantaggio.

2.1 Le necessità di cui tenere conto: i focus groups

Questa tesi nasce e prende spunto dal documento *"Supporting caregivers in assisted living facilities for person with disabilities: a user study"* [6] in cui gli autori hanno cercato di contribuire agli studi in ambito AAL per le residenze non ospedaliere definendo una serie di linee guida per la progettazione di un sistema che possa aiutare gli operatori sanitari nel lavoro quotidiano, dal monitoraggio degli ospiti delle RAF all'assicurare loro ogni tipo di assistenza necessaria.
L'obiettivo ultimo degli autori è stato quello di capire le necessità e le preoccupazioni con cui gli operatori sanitari che lavorano nelle RAF si devono confrontare ogni giorno, il modo che hanno di operare attualmente nei momenti in cui sorgono problemi e difficoltà, e come la tecnologia potrebbe aiutarli nel loro lavoro quotidiano. Le domande iniziali che hanno dato origine alla ricerca sono state le seguenti:

- Come può l'introduzione di un sistema che coinvolga tecnologie mobile/wearable supportare il lavoro quotidiano degli operatori sanitari nelle RAF?
- Quali sono le necessità, le difficoltà e i desideri che attualmente hanno gli operatori sanitari che lavorano nelle RAF?
- Quali sono i particolari bisogni che possono favorire o meno l'adozione di dispositivi mobile/wearable per supportare le attività quotidiane degli operatori sanitari?

I risultati ottenuti derivano da uno studio compiuto nel nord Italia in collaborazione con la cooperativa P.G. Frassati di Torino, in cui attraverso una particolare forma di indagine, i focus group, si sono intervistati circa 30 operatori sanitari di tre differenti RAF per persone con disabilità fisiche e cognitive. I focus group sono una particolare tipologia di intervista a gruppi, in cui i componenti sono lasciati liberi di interagire tra di loro. La tecnica è molto utilizzata poiché permette di capire meglio le attitudini, le credenze, le esperienze e le reazioni che hanno i componenti del gruppo, e le informazioni sono ottenute in modo molto più flessibile rispetto ad altre tecniche, come le interviste individuali. Per ogni gruppo, composto da circa 10 persone, è stata prevista una singola sessione da 90 minuti che si è svolta nella RAF stessa, per meglio capire il contesto, le necessità, i problemi e gli strumenti con cui gli operatori sanitari sono in contatto. Si è deciso di mantenere tre gruppi separati affinché i componenti si conoscessero meglio e fossero più liberi di parlare ed esporre le proprie idee. Queste sessioni si sono svolte dopo pranzo, l'unico periodo della giornata in cui gli assistenti sono più liberi di incontrarsi visto che gli ospiti delle RAF hanno l'abitudine di riposarsi dopo mangiato. Ogni focus group è iniziato con una introduzione di un membro del gruppo di ricerca, che ha spiegato le motivazioni dello studio e ha incoraggiato i componenti a parlare e a confrontarsi tra di loro. Dopo una fase di conoscenza reciproca, ai partecipanti sono state poste alcune domande aperte che hanno guidato il dibattito, con i moderatori che hanno prestato attenzione a non dare opinioni personali per evitare di influenzare il gruppo. Nello specifico alcune delle domande poste sono state:

- Potete spiegarci com'è strutturata una vostra giornata tipica?
- Nelle vostre attività quotidiane, avete escogitato qualche piano per facilitare o velocizzare il vostro lavoro?
- Cosa vorreste controllare degli ospiti della RAF quando non li avete strettamente sotto controllo (di notte per esempio)?
- Quali dispositivi tecnologici (tablet, smartphone, etc.) avete già proposto agli ospiti o avete utilizzato voi stessi durante il lavoro? Come vi siete trovati?
- In generale, avete delle richieste particolari? Per esempio, quali tipi di dispositivi vorreste utilizzare?
- Gli ospiti possono richiedere la vostra assistenza? Se sì, come fanno?
- Quando un ospite richiede assistenza, qual è un tempo ragionevole entro cui potete raggiungerlo?

2.2 Le linee guida del progetto

Analizzando più in dettaglio il documento da cui questa tesi prende spunto, in particolare guardando ai risultati ottenuti, ci si prefigge di progettare, sviluppare e testare un sistema di supporto agli operatori sanitari che lavorano nelle RAF che coinvolga tecnologie mobile/werable. L'avere a disposizione un confronto diretto con le persone a cui si rivolge questo progetto è stato di fondamentale importanza, sia perché è stato più facile individuare le linee guida da seguire, sia perché sapere che questo lavoro risponde a delle necessità concrete ha dato ad una spinta motivazionale in più. Come prima parte del lavoro sono state così estratte le informazioni che hanno guidato l'implementazione di questo progetto.
Uno dei primi requisiti che gli assistenti sanitari della cooperativa P.G. Frassati hanno tenuto a sottolineare, in tutti e tre i focus group, è la necessità di avere le mani libere durante lo svolgersi del loro lavoro. Questo è ragionevole, per il fatto che in ogni momento essi devono essere pronti a reagire a potenziali situazioni di rischio per gli ospiti della struttura, come in caso di attacchi epilettici o cadute. Il sistema sviluppato non deve dunque essere di intralcio per chi lo usa, e inoltre deve essere funzionante ovunque siano gli inquilini della RAF e gli operatori sanitari. Per questi motivi il progetto non prevede smartphone o tablet aggiuntivi (se non quelli già in dotazione agli utilizzatori) e propone l'utilizzo dei cosiddetti "wearable device", una delle ultime frontiere in campo tecnologico in cui i dispositivi, dotati di microprocessore interno, possono essere indossati dall'utente. Nei gruppi si è fatto notare che tutti i dispositivi introdotti dovrebbero essere resistenti all'acqua e agli urti, perché l'attenzione dev'essere focalizzata sugli inquilini della RAF e non sull'integrità degli oggetti. Inoltre, per andare incontro alle necessità di tutti i possibili utilizzatori, non si vogliono introdurre elementi inutili e complicati che potrebbero essere un deterrente per l'utilizzo del sistema: esso deve essere semplice da usarsi, anche per quelle persone non abituate al contatto continuo con la tecnologia. Tutti i dispositivi introdotti (possibilmente pochi) devono essere facilmente utilizzabili e portabili, e non devono costituire una preoccupazione per chi li usa, distogliendolo dai sui compiti principali. Occorre infatti fornire un supporto per l'operatore durante le sue attività, in modo immediato e ovunque egli sia, anche in movimento. L'usabilità è dunque un fattore fondamentale, e le interfacce grafiche devono essere quanto mai semplici e intuitive.
Studiando le necessità emerse dai focus group, comprese le limitazioni che attualmente ci sono nella gestione delle situazioni critiche, si è giunti alle linea guida riguardanti la modalità di allerta degli operatori sanitari in caso di necessità, il tutto con lo scopo di offrire un livello di assistenza ottimale. Gli ospiti delle RAF hanno bisogno di aiuto molte volte durante il giorno: le persone con disabilità fisica possono avere problemi anche in situazioni semplici come chiudere una porta o raggiungere un oggetto, mentre per situazioni di disabilità mentale grave possono essere comuni

gli attacchi epilettici. Allo stato attuale un ospite può richiedere aiuto a voce o con il sistema manuale di allarme: premuto un pulsante in una posizione fissa di una camera si scatena un allarme sonoro che viene avvertito in tutta la struttura, provocando disagio soprattutto nelle ore notturne. Oltre a questo allarme sonoro, un pannello luminoso con l'indicazione della camera dalla quale proviene la richiesta di aiuto si illumina nell'ufficio degli operatori sanitari in modo tale da permettere loro di sapere qual è il luogo del problema. Tuttavia esistono situazioni in cui la richiesta assistenziale non è possibile, né a voce né tramite allarmi sonori: un ospite potrebbe essere uscito in cortile, oppure potrebbe non essere in grado di raggiungere il pulsante per chiedere aiuto. Si pensi ad esempio a una persona in preda ad un attacco epilettico: di certo non potrà premere alcun bottone, e nemmeno urlare per chiedere aiuto, solo una sorveglianza continua da parte degli assistenti (che chiaramente non sarà mai perfetta) può evitare che la persona sia lasciata sola per troppo tempo. Attualmente nelle RAF soggette a questo tipo di problema sono stati escogitati dei metodi per sopperire all'inefficienza del controllo fisico continuo: l'utilizzo di "baby monitor" per quegli inquilini ad alto rischio di attacchi epilettici ne è un esempio, anche se non costituisce una soluzione ottimale, sia per problemi di privacy sia per il fatto che un assistente non può guardare un monitor in continuazione. Le chiamate perse rappresentano dunque il maggior problema non risolto nelle RAF e il sistema da progettare deve assolutamente tenerne conto, cercando comunque di non essere intrusivo: una richiesta di assistenza non deve disturbare gli altri inquilini della struttura. Occorre dunque migliorare il modo con cui una persona possa richiamare l'attenzione degli operatori, i quali non devono essere sottoposti allo stress di dover controllare continuamente tutti gli ospiti. Ecco allora che un sistema che sia veramente di aiuto per chi deve svolgere questo lavoro dovrebbe permettere, per quanto possibile, un monitoraggio continuo, automatico e ubiquo dei soggetti, riconoscendo alcune situazioni di pericolo e notificandolo automaticamente agli operatori sanitari: tutto questo ha fatto subito pensare all'utilizzo di dispositivi wearable dotati di appositi sensori.
A detta degli assistenti intervistati, che sono stati molto esigenti su questo punto, l'eventuale introduzione di un qualche sistema tecnologico non deve diminuire il livello di guardia nella gestione di un allarme. Sebbene l'utilizzo di un allarme rumoroso non sia la soluzione migliore, con tale metodo l'attenzione è subito attirata, e se prima occorreva che l'operatore sanitario andasse fisicamente nella camera di chi aveva richiesto aiuto per disattivare il segnale rumoroso, anche nel sistema da progettare non dovrà essere possibile ignorare facilmente un allarme. La paura comune è infatti quella che, con la possibilità di disattivare facilmente e remotamente un allarme, aumenti il numero di richieste non considerate, poichè un operatore potrebbe facilmente dimenticarsene. Inoltre servirà un metodo per indicare chi sta richiedendo aiuto, ruolo che attualmente è coperto dal pannello luminoso presente

nell'ufficio. Il protocollo che i dispositivi introdotti utilizzeranno dovrà allora riprodurre in qualche modo questa ridondanza nella gestione delle situazioni critiche in modo da assicurare che tutte le richieste vengano effettivamente prese in considerazione con la giusta attenzione, e dovrà segnalare con semplicità agli operatori la persona o il luogo in cui è necessario dirigersi. La robustezza e l'affidabilità avranno un valore fondamentale e la comunicazione dovrà essere il più possibile stabile e funzionante.
Un'ulteriore linea guida è quella del fornire la possibilità agli stessi operatori sanitari di richiedere aiuto in caso di pericolo. Essi lavorano a coppie durante il giorno, ma durante la notte sono soli. La ragione è che durante il giorno gli ospiti hanno bisogno di molte più attenzioni, mentre quando dormono la situazione è più tranquilla. Tuttavia dai focus group emerge la paura che gli assistenti hanno in caso dovesse succedere loro qualcosa quando sono soli, situazione che comprometterebbe sia la loro sicurezza sia quella di tutti gli ospiti della RAF. Il sistema di aiuto dovrebbe allora prevedere anche per gli operatori sanitari un modo semplice e veloce per notificare agli altri assistenti, magari a casa, una situazione di pericolo.
L'ultimo concetto, più generale dei precedenti, ma alla base della progettazione, indica che l'accuratezza del sistema non è un punto cruciale: gli operatori sanitari sono ben disposti a considerare qualche falso allarme se il sistema è in grado di segnalare automaticamente le situazioni di pericolo che prima dovevano essere rilevate di persona. Deve essere preferita dunque la segnalazione superflua di qualche richiesta di assistenza non necessaria rispetto alla mancata segnalazione. Su questo tutti i partecipanti ai focus group erano totalmente in accordo, poiché il desiderio di fornire assistenza ogni qual volta ve ne sia bisogno è il vero e proprio motore di questo progetto.

2.3 Gli obiettivi da raggiungere

Sulla base delle linee guida estratte grazie al documento relativo ai focus group è stato facile definire gli obiettivi che si vogliono raggiungere con questa tesi. Essendo abbastanza numerosi e di tipologie diverse si è scelto di riassumerli nella tabella 2.1. Gli obiettivi sono raggruppati secondo una macro area e ognuno di essi ha una priorità che va da 3 (priorità minima) a 1 (priorità massima): nella progettazione del sistema si tenderà a considerare con maggiore attenzione gli obiettivi più importanti. Nel capitolo 7 si andranno ad analizzare i risultati raggiunti, anche sulla base di test con utenti reali, e si potrà avere un riscontro sulla soddisfazione o meno dei traguardi prefissati.

Tabella 2.1: Obiettivi della tesi

Tipo	**Numero**	**Descrizione**	**Priorità**
Ubiquità	LG 1.1	Il sistema deve essere utilizzabile in qualunque condizione si trovino gli utilizzatori. In particolare dovrà funzionare anche quando gli operatori sanitari e gli ospiti della RAF sono in movimento.	1
	LG 1.2	Il sistema deve essere utilizzabile ovunque all'interno della struttura. Anche quando un ospite esce nel cortile il sistema deve essere in grado di rilevare e notificare eventuali situazioni di pericolo agli operatori sanitari.	1
	LG 1.3	I dispositivi utilizzati dal sistema devono essere facilmente portabili dagli operatori sanitari e soprattutto devono essere resistenti agli urti e all'acqua, in modo tale da non costituire una preoccupazione quando vengono utilizzati nelle situazioni critiche.	2
Usabilità	LG 2.1	Il sistema deve permettere agli operatori sanitari di avere sempre le mani libere, in modo tale da non essere d'intralcio con le loro attività quotidiane, soprattutto quando hanno il bisogno di soccorrere un ospite della RAF.	1
	LG 2.2	Il sistema di richiesta aiuto non deve essere intrusivo e non deve disturbare gli altri inquilini della struttura, soprattutto durante le ore notturne. Occorre dunque trovare una nuova modalità di avvertire gli operatori sanitari.	1
	LG 2.3	Il sistema non deve introdurre oggetti che siano di intralcio nel lavoro quotidiano degli operatori e che potrebbero essere facilmente dimenticati.	2
	LG 2.4	Il sistema deve essere facile da utilizzare per chiunque e gli elementi a stretto contatto con l'utente devono fornire un'interfaccia grafica intuitiva e semplice, in modo da non essere un deterrente per l'utilizzo.	2

	LG 2.5	Il protocollo utilizzato per la comunicazione tra i vari elementi del sistema deve rispondere alle esigenze espresse nei focus group dagli operatori sanitari e replicare i comportamenti richiesti. In particolare il sistema deve segnalare con chiarezza il luogo o la persona che sta richiedendo assistenza.	2
	LG 2.6	Il sistema non deve introdurre oggetti superflui per gli ospiti della RAF, come smartphone e tablet, poiché per le persone con disabilità più gravi c'è il rischio che vengano utilizzati male o rotti.	2
	LG 2.7	Nell'ottica di mantenere bassi i costi il sistema deve cercare di utilizzare i dispositivi già in dotazione di operatori sanitari e ospiti delle RAF senza introdurne troppi di nuovi.	2
Robustezza	LG 3.1	Il sistema deve essere robusto nell'identificare situazioni di pericolo. In particolare sono preferibili falsi allarmi a mancanze di segnalazione. L'affidabilità nel senso di rilevazione corretta non è dunque un fattore fondamentale: quello che conta è assicurare che tutte le situazioni di pericolo siano notificate.	1
	LG 3.2	Il sistema deve prevedere dei meccanismi per cui non sia facile ignorare un allarme da parte degli operatori. Per questo motivo il protocollo utilizzato dal sistema dovrebbe replicare il procedimento che attualmente occorre seguire per disattivare un allarme, ovvero quello di doversi recare per forza nella camera dell'ospite che ha richiesto aiuto.	2
	LG 3.3	Il sistema deve essere intrinsecamente sicuro e affidabile, senza che vengano perse richieste di aiuto. La comunicazione tra i dispositivi è un elemento fondamentale e deve essere affidabile.	2

Supporto per gli operatori	LG 4.1	Il sistema deve introdurre effettivi benefici per gli operatori sanitari. In particolare deve permettere il richiamo della loro attenzione in caso di pericolo senza che essi siano costretti a un controllo continuo degli ospiti.	1
	LG 4.2	Il sistema deve riconoscere in automatico, senza un'esplicita richiesta, eventuali situazioni di pericolo per gli ospiti della RAF.	1
	LG 4.3	Il sistema di supporto deve prevedere la possibilità di richiesta aiuto anche per gli operatori sanitari. Questo dovrebbe servire soprattutto di notte, quando gli assistenti sono soli.	3

Per concludere, l'intento finale del sistema è di aiutare entrambe le figure presenti in una RAF:

- Dal punto di vista dell'ospite, la sua permanenza diventa più sicura, in quanto può richiedere assistenza anche quando è in movimento nella struttura e le sue condizioni sono costantemente monitorate.
- Dal punto di vista dell'operatore, il sistema dovrebbe migliorarne la condizione lavorativa. Infatti egli non dovrà più effettuare quel controllo costante degli ospiti a cui oggi non può rinunciare.

Si presume inoltre che anche il tempo di intervento diminuisca, soprattutto nel caso di attacchi epilettici, quando difficilmente l'ospite può richiamare l'attenzione in altri modi: il monitoraggio tramite il dispositivo wearable permette l'immediato invio di una richiesta di assistenza.

Capitolo 3

I componenti del progetto

Nella prima fase del ciclo di vita del sistema sono stati scelti i componenti da utilizzare per lo sviluppo. Partendo dalle linee guida definite in precedenza, in particolare quelle relative all'ubiquità e all'usabilità, si è subito pensato all'utilizzo di dispositivi wearable. Tali oggetti allo stato attuale non possono comunicare direttamente tra di loro, ma hanno bisogno di un dispositivo mobile di supporto. Se disponibili verranno allora utilizzati gli smartphone già in dotazione a assistenti e ospiti, altrimenti occorrerà procurarsene di nuovi. Come ultimo elemento, non meno importante, occorre prevedere la presenza di un server che coordini la comunicazione tra i vari elementi e che implementi gran parte della logica del progetto.
In questo capitolo verrano prima descritti i dispositivi fisici adottati, per poi porre l'attenzione sui linguaggi di programmazione e i framework adottati.

3.1 I dispositivi utilizzati

3.1.1 Gli smartwatch Pebble

I "wearable device" rivestono una parte fondamentale del progetto, poiché sono quelli a stretto contatto con l'utente e dovrebbero essere in grado di attuare il monitoraggio continuo e automatico delle condizioni degli ospiti in qualunque posto della RAF essi si trovino (LG 1.1 e LG 1.2).
Per lo sviluppo del prototipo sono stati utilizzati quattro smartwatch prodotti dalla Pebble Technology Corporation [7] già a disposizione del Politecnico di Torino. Pebble è una società nata nell'anno 2011 che ha iniziato a produrre questi orologi grazie al sistema del crowfounding, un processo collaborativo di un gruppo di persone che utilizza il proprio denaro in comune per sostenere gli sforzi di singoli soggetti e organizzazioni. Secondo il Framework for European Crowdfunding, *"l'ascesa del crowdfunding negli ultimi dieci anni deriva dal proliferare e dall'affermarsi di applicazioni web e di servizi mobile, condizioni che consentono a imprenditori, imprese e*

creativi di ogni genere di poter dialogare con la crowd per ottenere idee, raccogliere soldi e sollecitare input sul prodotto o servizio che hanno intenzione di proporre" [8]. Attualmente l'ultima versione dello smartwatch è chiamata Pebble Time: essa presenta numerosi miglioramenti e aggiunte rispetto alla prima generazione dei dispositivi. Gli orologi della precedente versione sono dotati di un display LCD in bianco e nero, una CPU programmabile, una memoria, un motore di vibrazione, un sensore di luce ambientale, un accelerometro e hanno la capacità di comunicare tramite Bluetooth. Gli orologi Pebble Time posseggono invece un display a colori, hanno una batteria più performante e dispongono anche di un microfono. Oltre alle applicazioni utilizzabili direttamente sul Pebble, per poter comunicare in rete o utilizzare dati provenienti da Internet gli smartwatch hanno bisogno di uno smartphone o di un tablet di supporto. La comunicazione è possibile sia con dispositivi Android che iOS, utilizzando il Bluetooth 2.1 o il Bluetooth 4.0 (Bluetooth Low Energy), e le due parti rimangono collegate grazie all'applicazione ufficiale Pebble da installarsi sul dispositivo mobile.
Tutte queste caratteristiche estendono l'utilizzo di questi smartwatch, oltre alla semplice visualizzazione del tempo, all'interazione con le notifiche degli smartphone, al monitoraggio di attività, ai giochi e altro ancora.
Uno dei principali vantaggi nell'utilizzare questi dispositivi è dato dalla loro batteria: a seconda dell'utilizzo la durata non scende comunque sotto diversi giorni e tutte le librerie e le API fornite sono ottimizzate per essere a basso consumo. Esistono comunque operazioni che diminuiscono questo effetto, come l'uso dell'accelerometro, di cui questo prototipo fa largo uso: dopo vari test l'incidenza sulla batteria risulta comunque accettabile. Altra caratteristica interessante è data dalla robustezza di tali dispositivi (LG 1.3). In particolare è garantita la loro resistenza all'acqua sotto opportune condizioni: gli orologi hanno una impermeabilità di 5 atm, il che significa che possono essere immersi sia in acqua dolce che salata fino a 40 metri (130 piedi).
Un elemento ampiamente sfruttato nel sistema è l'accelerometro. La causa dell'accelerazione misurabile può essere la gravità o un certo tipo di movimento fisico: la torsione di un polso, il tocco delle dita e quant'altro. Anche una spinta della mano verso l'alto o verso il basso può essere misurata da un accelerometro. L'accelerometro del Pebble (figura 3.1) è in grado di effettuare misure a una data frequenza e di trasmettere, tramite le API fornite, più campioni raggruppati per risparmiare batteria. Esso è calibrato per misurare l'accelerazione in un range che va da -4G a +4G, i dati raccolti sono forniti secondo uno spazio a tre dimensioni e ogni valore è misurato in milli-Gs.
A partire da febbraio 2014, l'App Store di Pebble ha visto la nascita di un numero crescente di applicazioni, che possono riguardare notifiche per e-mail, chiamate, messaggi di testo, notifiche social, monitoraggio attività (movimento, sonno, stime di calorie bruciate) e molto altro. Sfruttando anche le capacità del dispositivo mobile associato le possibilità di sviluppo aumentano in modo esponenziale.

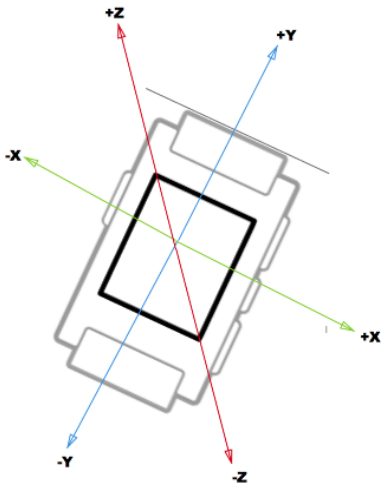


Figura 3.1: Accelerometro integrato dello smartwatch Pebble

3.1.2 I dispositivi Android

Come visto in precedenza per comunicare tra di loro gli smartwatch hanno bisogno di un dispositivo mobile di supporto come uno smartphone o un tablet, e dovendo sviluppare un sistema distribuito la comunicazione è un elemento fondamentale. Uno smartphone è un telefono cellulare con un avanzato sistema operativo che combina le caratteristiche di un personal computer con altri elementi utili per l'uso mobile, mentre un tablet è una versione semplificata e portabile di un personal computer. La scelta è ricaduta sui dispositivi che abbiano come sistema operativo Android [9], che di fatto è quello più utilizzato al mondo in ambito mobile, lasciando le altre piattaforme come iOS a eventuali sviluppi futuri. Android è un sistema operativo open source e facile da utilizzare, installato su più di un miliardo di dispositivi in tutto il mondo: telefoni e tablet, orologi, TV, auto e altri dispositivi. Realizzato da Google, è basato su kernel Linux, e può dunque essere considerato a tutti gli effetti al pari di una distribuzione GNU/Linux per sistemi embedded. La licenza (Licenza Apache) sotto cui è distribuito consente di modificare e distribuire liberamente il codice sorgente. Inoltre Android dispone di una vasta comunità di sviluppatori che realizzano applicazioni con l'obiettivo di aumentare le funzionalità dei dispositivi. Queste applicazioni sono scritte soprattutto in linguaggio di programmazione Java. Android utilizza la Dalvik virtual machine con un compilatore just-in-time per l'esecuzione di Dalvik dex-code (Dalvik Executable), che di solito viene tradotto da codice bytecode Java. Più di recente, nella versione *4.4 KitKat*, è stato introdotto in via sperimentale un nuovo runtime system software, Art (Android RunTime), che ha sostituito definitivamente la virtual machine Dalvik con il rilascio di Android 5.0. Attualmente il sistema operativo arriva alla versione 5.0 chiamata *Lollipop*, ma per

poter utilizzare i dispositivi già in dotazione di assistenti e ospiti si è pensato di scrivere applicazioni compatibili con versioni più vecchie, in particolare a partire dalla 4.0 *Ice Cream Sandwich*. Sicuramente alcune funzionalità potranno sembrare non ottimizzate, come l'utilizzo del Bluetooth classico al posto del più recente Bluetooth Low Energy, ma si è preferito non imporre l'utilizzo di nuovi dispositivi o l'aggiornamento di quelli esistenti, soprattutto nell'ottica del mantenere i costi limitati e di non aggiungere ulteriori dispositivi agli operatori (LG 2.3 e 2.7). L'interfaccia utente di Android è basata sul concetto di *direct manipulation* per cui si utilizzano gli ingressi mono e multi-touch come strisciate, tocchi e pizzichi sullo schermo per manipolare gli oggetti visibili sullo stesso. Sensori hardware interni come accelerometri, giroscopi e sensori di prossimità sono utilizzati da alcune applicazioni per rispondere alle azioni da parte dell'utente: la regolazione dello schermo da verticale a orizzontale a seconda dell'orientazione del dispositivo ne è un esempio. Android ha un rapido ciclo di aggiornamento, con la distribuzione di nuove versioni ogni sei-nove mesi. Gli aggiornamenti sono in genere di natura incrementale, piuttosto che revisioni complete del sistema ogni due o tre anni (pratica comune per i sistemi operativi desktop come Windows). Tra una major release e l'altra vengono messi a disposizione aggiornamenti intermedi per risolvere problemi di sicurezza e altri bug del software.

3.2 I framework e i linguaggi di programmazione adottati

Il progetto prevede l'utilizzo di linguaggi di programmazione e framework differenti poiché i dispositivi coinvolti sono diversi. Uno dei punti critici dello sviluppo è stato infatti integrare tutte le tecnologie adottate e far comunicare senza problemi dispositivi di natura diversa. In questa sezione verrano presentate le tecnologie software utilizzate per i vari componenti che costituiscono il sistema.

3.2.1 Tecnologie per il server

Il server è stato implementato interamente in linguaggio Java, in particolare come un'applicazione web Java Enterprise Edition, la cui struttura base è rappresentata in figura 3.2. Una applicazione Java EE è formata da un'insieme di classi che risiedono su un server remoto. Un programma, detto container, è in esecuzione sul server e monitora le connessioni entranti. Tutte le volte che viene ricevuta una richiesta il server istanzia uno o più oggetti incaricati all'elaborazione e inoltra la risposta generata al mittente. Esistono diversi container in commercio, sia a pagamento che open source: per il progetto è stato utilizzato Apache Tomcat. Il contenitore gestisce il protocollo HTTP e utilizza un template standard per descrivere le applicazioni

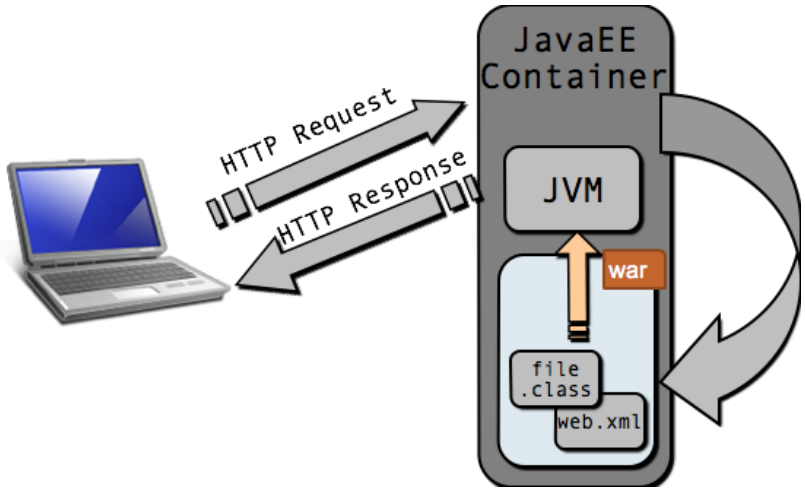


Figura 3.2: Schema di una applicazione Java EE

web, che sono distribuite sotto forma di un archivio .war.
I componenti elementari di una applicazione web sono diversi, e vengono istanziati e controllati in tutto il loro ciclo di vita dal container, al fine di elaborare richieste entranti e produrre risposte per l'utente. Di seguito vengono presentati i componenti principali, con le loro caratteristiche.

- *Servlet*: gli elementi più importanti sono sicuramente i servlet, classi Java responsabili di gestire le richieste provenienti dal contenitore, elaborarle e produrre una risposta. Il contenitore crea un'unica istanza del servlet e ne invoca il metodo di inizializzazione. Ad ogni richiesta il contenitore invoca il metodo di servizio del servlet utilizzando un thread differente.

- *Filtri*: i filtri sono invece componenti Java che trasformano le richieste inviate a un servlet o le risposte da esso generate, aggiungendo un livello di funzionalità che può essere inserito dinamicamente nel contesto dell'applicazione. In questo modo è possibile descrivere comportamenti trasversali all'applicazione separandoli dal dominio applicativo, definendo funzionalità comuni a più oggetti in un unico posto. Un classico esempio può essere la gestione dell'autenticazione, che senza l'uso di questi componenti andrebbe ripetuta in diversi punti dell'applicazione, cosa che potrebbe portare a facili errori di coerenza. La logica relativa ai comportamenti trasversali in questo modo non è ripetuta e le altre classi sono più pulite perché contengono solo le loro funzionalità primarie.

- *Pagine JSP*: altri componenti importanti sono le pagine JSP, che in realtà non sono nient'altro che dei servlet mascherati da file di testo. Vengono utilizzate per creare dinamicamente contenuti web quando le risposte sono per lo più

testuali, perché offrono la possibilità di annegare codice Java all'interno di una struttura HTML. Il client non vede mail la pagina JSP iniziale, ma solo i risultati della sua esecuzione: esse infatti sono elaborate dal contenitore che le trasforma in codice eseguibile.

- *Listener*: gli ultimi elementi principali di una applicazione Java EE sono i listener, che permettono a una applicazione di reagire opportunamente ai cambiamenti di stato dei singoli componenti. L'applicazione deve registrare opportuni oggetti che svolgono il ruolo di ascoltatori (listener) di tali eventi.

Representational State Transfer - REST

L'evoluzione di Internet impone la realizzazione di sistemi distribuiti scalabili, in grado di garantire l'interoperabilità tra piattaforme differenti. Nel sistema da sviluppare questa esigenza è molto forte, e per la parte principale del progetto il server offre un servizio RESTful al quale i dispositivi di operatori sanitari e ospiti della RAF possono interfacciarsi per poter comunicare. REST è uno stile con il quale realizzare architetture distribuite particolarmente adatto per applicazioni distribuite che durano a lungo nel tempo. Il concetto è stato introdotto da Roy Fielding nella sua tesi di dottorato nell'anno 2000. Presuppone la creazione di una architettura client-server priva di stati, in cui il server ospita uno o più servizi che gestiscono un insieme di informazioni, ciascuna delle quali ha un nome su scala globale (URL), può avere molteplici rappresentazioni tra loro semanticamente equivalenti (testo semplice, XML, JSON, etc.) e supporta operazioni elementari di tipo CRUD (Create Read Update Delete). Le informazioni associate a una URL possono essere relative a singole voci, collezioni di voci o risultati di computazioni funzionali.
Il protocollo HTTP costituisce un ambiente naturale nel quale le richieste di tipo CRUD possono essere mappate sulle caratteristiche del trasporto: ogni richiesta consiste di un'azione (verbo) e di una URL (nome), come mostrato in tabella 3.1.

Tabella 3.1: Mapping dei verbi HTTP secondo lo schema REST

Operazione	**Metodo**	**Oggetto**
Crea (voce singola)	POST	URL della collezione
Leggi (elenco di voci)	GET	URL della collezione
Leggi (voce singola)	GET	URL dell'elemento
Aggiorna (voce singola)	PUT	URL dell'elemento
Cancella (voce singola)	DELETE	URL dell'elemento

Nel progetto, il paradigma REST è stato implementato con il framework Jersey, implementazione di riferimento della specifica Java API for RESTful Web Services (JAX-RS). Disponibile con licenza open source, può essere integrato in prodotti commerciali o essere esteso con comportamenti specializzati. Mette a disposizione un servlet specializzato per servire le richieste rivolte a oggetti Plain Old Java Object (POJO) opportunamente annotati.

Il paradigma Model View Controller

Si è scelto di non utilizzare framework particolari soprattutto per la parte REST per cercare di ottimizzare al massimo le performance. Il server è stato costruito seguendo il paradigma *Model View Controller* (figura 3.3), che semplifica la gestione e la manutenibilità del codice e permette allo sviluppo di essere più logico e intuitivo. Con esso le applicazioni sono costruite mediante un approccio stratificato, in cui

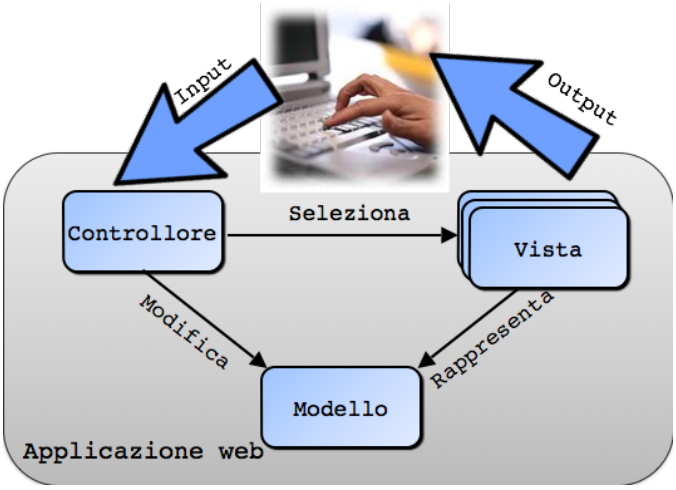


Figura 3.3: Il paradigma Model View Controller

ciascuno strato interagisce con gli strati adiacenti e limita le dipendenze reciproche definendo i punti di contatto sotto forma di interfacce. L'approccio enfatizza la modularità della soluzione e il principio della separazione delle responsabilità.

- Il *modello* contiene i dati dell'applicazione e impone delle regole sulla loro evoluzione. Viene costruito con vari livelli che vanno dallo strato responsabile della comunicazione con la sorgente dati allo strato di servizio, che definisce il contatto tra l'utente e l'applicazione, coordina le attività e delega i vari compiti agli oggetti del dominio applicativo.

- La *vista* è un oggetto specializzato nel mostrare quanto contenuto nel modello, di cui contiene al proprio interno un riferimento. Di solito le viste non modificano il modello, e nel progetto corrente, seppur poco usate, prendono la forma di pagine JSP.
- Il *controllore* invece è l'oggetto che mette in comunicazione le azioni eseguite sull'interfaccia con le operazioni offerte dal modello. Il controllore interpreta le azioni svolte dall'utente e agisce sul modello attraverso le interfacce di servizio, eventualmente scegliendo la nuova vista da mostrare.

Per ogni tipo di richiesta viene definito un URL opportuno su cui viene mappato il controllore responsabile della sua gestione.

La persistenza delle informazioni: MySQL, Hibernate e JPA

Nel progetto la persistenza delle informazioni è molto importante ed è implementata dal server in collaborazione con una base dati. Come database è stato scelto MySQL, il più diffuso al mondo tra quelli open source. MySQL è un Relational Database Management System (RDBMS) composto da un client a riga di comando e un server. Entrambi i software sono disponibili sia per sistemi Unix e Unix-like che per Windows. Un RDBMS è un sistema di gestione di database (DBMS) che si basa sul modello relazionale inventato da E.F. Codd nei Laboratori di Ricerca IBM. Questo modello si basa sulla teoria degli insiemi e sulla logica del primo ordine ed è strutturato intorno al concetto matematico di relazione (detto anche tabella). Per il suo trattamento ci si avvale di strumenti quali il calcolo relazionale e l'algebra relazionale. L'assunto fondamentale del modello relazionale è che tutti i dati sono rappresentati come relazioni e manipolati con gli operatori dell'algebra relazionale o del calcolo relazionale, da cui appunto il nome. Il modello relazionale consente al progettista di database di creare una rappresentazione consistente e logica dell'informazione. La consistenza è ottenuta inserendo nel progetto del database appropriati vincoli, che normalmente sono riassunti in quello che è chiamato schema logico. La teoria comprende anche un processo di normalizzazione in base al quale viene selezionato tra le diverse alternative lo schema maggiormente adatto a rappresentare le entità.
Nel campo della programmazione Java l'approccio ad oggetti si presta molto bene a rappresentare sistemi software complessi, ma manca un meccanismo generalizzato per rendere persistenti le informazioni. Per contro, le basi dati relazionali sono ben consolidate, ma richiedono lo sviluppo di grandi quantità di codice di configurazione per adattarne i contenuti al modello ad oggetti. Per superare questo disadattamento si può utilizzare il concetto di *Object-Relational Mapping* (ORM), uno strato software che converte i tipi di dato da un formato ad oggetti ad uno relazionale e viceversa. Ad ogni riga del database si associa un oggetto di classe opportuna, introducendo un meccanismo per cui oggetto e riga si mantengono sincronizzati, almeno

in momenti specifici. Un ORM è dunque un approccio concettuale alla modellazione, interrogazione e trasformazione dei dati che rende trasparenti le operazioni di conversione e rappresentazione a basso livello delle informazioni all'interno della base dati. Esso è orientato ai fatti e le corrispondenze tra oggetti e basi dati sono rese esplicite, così come le dipendenze tra gli oggetti stessi. L'ORM genera internamente le istruzioni necessarie per accedere alla base dati garantendo efficienza, correttezza, scalabilità, manutenibilità e flessibilità. Ogni linguaggio ad oggetti dispone di molte alternative implementative con leggere variazioni sintattico-semantiche. Nel caso di Java si parla comunemente di *Java Persistence API* (JPA), una specifica Java EE la cui implementazione più nota è il framework *Hibernate*. Ad ogni entità presente nel modello logico viene associata una classe Java, tramite configurazioni XML o annotazioni sulle classi. Per ogni proprietà degli oggetti è possibile mappare l'opportuna colonna della tabella del database, e se un'entità è legata ad un'altra tramite un'associazione di cardinalità superiore ad uno, essa offrirà, nella classe Java, una proprietà di tipo Collection.
In una base dati, le relazioni sono intrinsecamente bidirezionali, mentre nel campo della programmazione ad oggetti le relazioni sono modellate tramite puntatori, intrinsecamente unidirezionali. Se, per la logica applicativa, è utile poter navigare una relazione in entrambe le direzioni, occorre che essa sia definita in entrambe le entità, indicando le eventuali tabelle di appoggio cui fare riferimento e le colonne su cui effettuare le operazioni di join.
Un concetto importante in questo campo è quello della sessione. Una sessione rappresenta una singola unità di lavoro: tipicamente, in un'applicazione web, coincide con l'elaborazione di una richiesta e la generazione della relativa risposta. Nell'ambito della sessione si svolgono una o più transazioni. Al loro interno Hibernate accede al DBMS sia in lettura (a seguito di ogni singola query) sia in scrittura, nel momento in cui si esegue il commit. Dal punto di vista della persistenza, un oggetto Java istanza di una classe mappata, può assumere tre stati differenti:

- *Transient*: appena creato attraverso il costrutto new, l'oggetto non ha alcun legame di persistenza.

- *Persistent*: quando viene associato ad una sessione attraverso opportuni metodi un oggetto diventa persistente. Eventuali modifiche verranno rese persistenti alla chiusura della transazione.

- *Detached*: dopo la chiusura della transazione, l'oggetto perde il legame con il contesto di persistenza.

Hibernate utilizza un suo particolare linguaggio di query chiamato HQL, simile a SQL, ma che supporta anche l'ereditarietà e il polimorfismo e supera il fatto che esistono molti dialetti SQL non standard.

3.2.2 Google Cloud Messaging (GCM)

HTTP è un protocollo sviluppato secondo il paradigma richiesta/risposta. Non prevede dunque la possibilità di comunicazioni asincrone, modalità strettamente necessaria ai fini di questo progetto. Per inserire tale funzionalità si è utilizzato Google Cloud Messaging (GCM), un servizio fornito da Google che permette di inviare dati dal server ai dispositivi degli utenti senza che questi facciano alcuna richiesta. Il servizio GCM, la cui struttura implementativa è rappresentata in figura 3.4, gestisce tutti gli aspetti di accodamento dei messaggi e di consegna alle applicazioni client in esecuzione sui dispositivi, ed è completamente gratuito. Con tale servizio si possono spedire messaggi a dispositivi singoli o a gruppi di utenti attraverso una connessione affidabile e ottimizzata per essere a basso consumo di batteria. Un'implementazione GCM include un server di Google di connessione, un application server nel proprio ambiente che interagisce con il server Google via HTTP o XMPP, e almeno un'applicazione client.

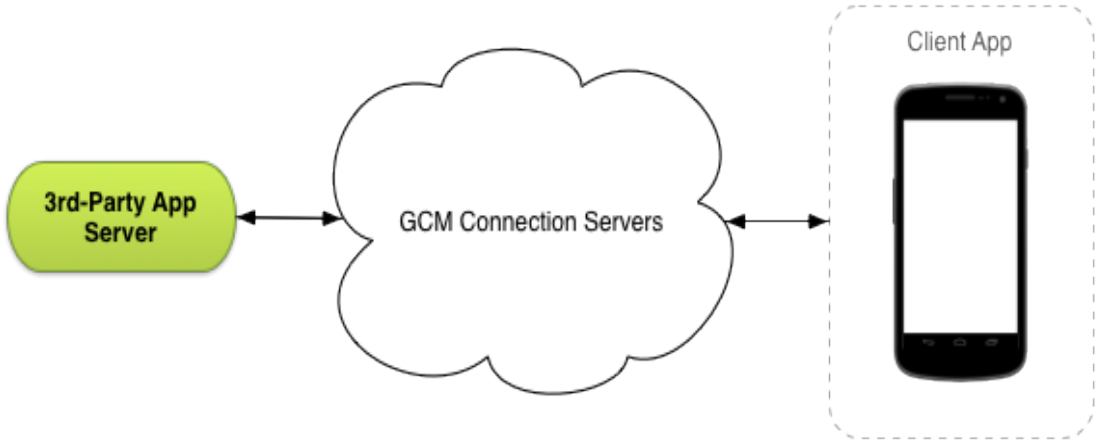


Figura 3.4: Struttura di un'implementazione GCM

3.2.3 Tecnologie per gli smartwatch

Gli smartwatch Pebbe hanno il pregio di essere forniti con software totalmente open source: è possibile utilizzare l'SDK scaricabile direttamente sul sito web ufficiale per iniziare a implementare facilmente nuove applicazioni. Come i dispositivi hardware anche l'SDK è presente in due versioni, una per gli orologi di prima generazione e una, più ricca, per quelli di seconda generazione. Esiste anche la possibilità di utilizzare l'ambiente di sviluppo online offerto da Pebble, molto intuitivo e facile da utilizzare. Le librerie esistenti permettono di scrivere codice in C o in Javascript e come parte principale sviluppano un particolare protocollo per la comunicazione tra

l'orologio e il dispositivo mobile associato. Il kit di librerie e di API per l'interazione tra le applicazioni Pebble e smartphone o tablet è disponibile sia per la piattaforma Android che per iOS e fornisce tre diverse implementazioni, di cui la principale è basata su un protocollo che prevede la comunicazione bidirezionale basata su messaggi che devono essere confermati ad ogni invio.
Per il progetto si è scelto una programmazione interamente in linguaggio C, cercando di sviluppare il codice con una modalità simile alla programmazione ad oggetti, ovvero utilizzando il concetto di *Abstract Data Type (ADT)* per le entità che modellano il sistema. Un tipo di dato astratto, in informatica e specificamente nel campo della programmazione, è un tipo di dato le cui istanze possono essere manipolate con modalità che dipendono esclusivamente dalla semantica del dato e non dalla sua implementazione. Nei linguaggi di programmazione che consentono la programmazione per tipi di dati astratti, un dato viene definito distinguendo nettamente la sua interfaccia, ovvero le operazioni che vengono fornite per la sua manipolazione, e la sua implementazione interna, ovvero il modo in cui le informazioni di stato sono conservate e in cui le operazioni manipolano tali informazioni al fine di esibire, all'interfaccia, il comportamento desiderato. La conseguente inaccessibilità dell'implementazione viene spesso identificata con l'espressione incapsulamento. Questo comportamento è stato ottenuto in linguaggio C mediante l'uso appropriato dei file di header, che sono stati fatti corrispondere all'interfaccia degli oggetti che modellano il sistema.

3.2.4 Tecnologie per i dispositivi mobili

Avendo scelto come target i dispositivi Android lo sviluppo di applicazioni è stato fatto attraverso l'apposito framework. Le applicazioni Android sono caratterizzate da una certa dualità: parti dinamiche scritte in Java e parti statiche scritte in XML. Tipico delle parti statiche possono essere quelle caratteristiche che non cambiano durante l'esecuzione dell'applicazione, come per esempio le caratteristiche usate per il design dell'interfaccia grafica. Tipico delle parti dinamiche sono invece gli aspetti programmatici come per esempio la gestione degli eventi. Il linguaggio per applicazioni Android è in realtà un dialetto del linguaggio Java così come è diversa anche la virtual machine di runtime (Dalvik virtual machine anziché JVM). Nella tipica applicazione Android non c'è un entry point (il classico metodo "main") da dove normalmente un programma inizia la sua elaborazione: tutto è pensato per essere un componente pilotato dagli eventi ("Event Driven"). Una applicazione è infatti formata da un insieme di componenti invocabili individualmente. Questo paradigma fa sì che il programmatore sviluppi delle routine il più possibile indipendenti. Un vantaggio è che il sistema operativo potrà ottimizzare le risorse, ad esempio rinunciando a caricare componenti non supportati o non prioritari perché inutilizzati.
Dal punto di vista dello sviluppo gli elementi fondamentali forniti da Android sono:

- Activity
- Intent e Intent Filter
- Broadcast Receiver
- Service

Una *activity* è un componente che gestisce una schermata dell'applicazione con cui l'utente può interagire per compiere determinate azioni. Ognuno di questi elementi produce una finestra nella quale è responsabilità del programmatore disegnare una interfaccia utente appropriata. Una applicazione di solito è composta da più *activity* che sono interconnesse tra di loro e di cui una di esse prende il ruolo di finestra principale da cui è possibile raggiungere tutte le altre. Il sistema infatti mantiene uno stack di finestre con cui è resa possibile la navigazione. Ci sono numerosi eventi che una *activity* può ricevere e che si possono catturare con la registrazione di opportune callback. Ad esempio alla chiusura sarebbe bene rilasciare tutte le risorse acquisite, poiché la potenza di elaborazione a disposizione di un dispositivo mobile è limitata.
Un *intent* è un messaggio con il quale si può richiedere una azione ad un altro componente dell'applicazione. Per rendere l'idea è comune utilizzare un messaggio del genere per avviare una nuova *activity*: essa è avviata passando un *intent* che ne descrive le caratteristiche a un opportuno metodo factory. Un *intent* contiene informazioni che il sistema usa per determinare il componente da istanziare, più informazioni che lo stesso elemento istanziato può usare per compiere le proprie elaborazioni. Gli *intent filter* sono un metodo per mostrare al resto del sistema le azioni che un'applicazione può compiere: un'altra applicazione, conoscendole, può decidere di utilizzarle.
Un *broadcast receiver* è un componente atto a rispondere a messaggi broadcast, ovvero *intent*, provenienti dal sistema Android o dall'applicazione stessa. Registrati tali componenti su determinati eventi, al verificarsi di essi il *broadcast receiver* avrà la possibilità di compiere determinate elaborazioni e istanziare eventuali nuovi componenti.
Un *service* è un componente che permette di eseguire operazioni lunghe in background e che non prevede una user interface. Quando un servizio è fatto partire da un altro componente compie le proprie azioni per un tempo indefinito. Un servizio normale vive nel main thread dell'applicazione: per far sì che venga utilizzato in un thread separato, e dunque che il suo ciclo di vita sia indipendente da quello dell'applicazione, si può usare un *intent service*.

Capitolo 4

Progettazione

In questo capitolo è esposto il progetto del sistema, la cui definizione è stata una delle prime parti di questo lavoro e ha rivestito un ruolo fondamentale. L'evoluzione di questa fase è stata aiutata e guidata dall'avere a disposizione i risultati dei focus group. Nonostante durante la fase di sviluppo alcuni aspetti siano per forza di cose cambiati il progetto iniziale è rimasto una forte fonte di ispirazione e anche l'implementazione stessa del sistema ne ha beneficiato: la progettazione, insieme ai test svolti, è stata sicuramente una delle parti più importanti. Verrà prima descritta l'architettura generale per poi scendere più nel dettaglio nelle varie componenti del sistema.

4.1 Architettura generale

La progettazione di un sistema di supporto per gli operatori sanitari nelle RAF è iniziata basandosi sui requisiti e sugli obiettivi definiti nei capitoli precedenti.
Lo spunto principale attraverso cui la progettazione ha preso il via è stata la necessità di sviluppare un sistema che avverta in modo automatico gli operatori sanitari in caso di situazioni di pericolo per gli ospiti delle residenze (LG 1.2 e 4.2). Dovendo dunque effettuare una forma di monitoraggio continuo degli inquilini ci si è rivolti al campo della sensoristica, prevedendo dei dispositivi wearable direttamente a contatto con gli utenti.
Nella prima parte del progetto si è cercato di capire quali situazioni critiche per gli utenti il sistema dovesse rilevare e in che modo. Gli ospiti delle RAF analizzate soffrono comunemente di due tipi di problemi: i disabili psichici hanno il problema degli attacchi epilettici, mentre per i disabili fisici più gravi sono comuni le cadute. Come si può notare tutte e due le situazioni sono legate al movimento della persona e si è dunque pensato di effettuare il monitoraggio affidandosi a sensori di movimento. Sebbene esistano in commercio dispositivi con numerosi sensori a bordo, quello

più comune, che anche i dispositivi di fascia più bassa posseggono, è l'accelerometro. Esso è presente anche sugli smartwatch Pebble e il suo uso per la rilevazione automatica di richieste di assistenza si è rilevato un ottimo compromesso sia dal punto di vista funzionale che per il mantenimento di costi bassi in previsione di una espansione futura del sistema (LG 2.7). Per aumentare la flessibilità è prevista anche una modalità di richiesta di assistenza manuale tramite lo smartwatch che simuli il campanello di allarme già esistente, ma con il vantaggio di essere mobile, sempre a portata dell'ospite della RAF. In questo modo non solo la persona bisognosa di aiuto non avrà la necessità di raggiungere fisicamente il punto della stanza dove era presente il campanello di allarme (cosa che potrebbe anche non riuscire a fare), ma potrà addirittura richiedere assistenza anche al di fuori della camera (LG 1.1 e 1.2). Affinché il sistema gestisca gli allarmi in modo robusto e non permetta a un operatore sanitario di ignorare facilmente una richiesta, per segnalarne la fine sarà necessario operare direttamente con lo smartwatch dell'ospite della RAF: l'operatore dovrà dunque recarsi fisicamente dall'ospite in difficoltà per evadere una richiesta (LG 3.2).
Passando poi ad analizzare la figura dell'operatore sanitario l'attenzione è stata posta sul requisito relativo alla non intrusività del sistema, ovvero al fatto che una richiesta di assistenza non debba recare disturbo agli altri ospiti della RAF (LG 2.2). Per questa ragione si è ipotizzato l'uso degli smartwatch anche per gli operatori sanitari, con il compito di visualizzare le richieste di aiuto in corso, comprendenti il tipo e la persona richiedente. In questo modo le richieste di assistenza arriveranno direttamente e senza provocare disturbo ai polsi degli assistenti, che potranno prenderle in carico. In sostanza questi dispositivi andranno a sostituire gli altri due elementi che insieme al campanello nelle stanze formano il sistema di richiesta assistenziale attuale: l'allarme sonoro e il pannello luminoso nell'ufficio che visualizza la camera del richiedente aiuto. Anche in questo caso il sistema non vincolerà più un operatore sanitario ad essere in uno specifico luogo per sapere dove dirigersi in caso di necessità, ma sarà effettivamente funzionante in qualunque punto della RAF egli si trovi (LG 1.1). Come richiesto nei focus group, tramite questi smartwatch gli operatori sanitari potranno inviare una richiesta di aiuto agli altri colleghi, anche se questi dovessero essere a casa (LG 4.3).
La progettazione della comunicazione tra i dispositivi è stata la fase successiva. Premettendo che, come già visto, i dispositivi wearable hanno bisogno di uno smartphone o di un tablet di supporto per poter comunicare in rete, e che tra smartwatch e dispositivo mobile la comunicazione avviene tramite Bluetooth, la parte principale è stata definire con quale modalità i dispositivi di una persona possano comunicare con quelli di un'altra. Dovendo mantenere traccia degli allarmi in corso e di quelli già evasi, questi ultimi soprattutto per scopi di documentazione e di ricerca, è stato necessario prevedere un server centrale che coordini il lavoro e si interfacci con un database per la persistenza delle informazioni. In questo modo la comunicazione si

spezza in due tronconi: ci sarà il tratto tra ospite della RAF e server, e il tratto tra il server e l'operatore sanitario. Avendo a disposizione un nodo centrale è stato anche più facile implementare la comunicazione del tipo "uno a molti" che in questo sistema ha molta rilevanza: si pensi al caso che un ospite richieda aiuto, il messaggio arrivato al server dovrà essere veicolato a tutti gli operatori sanitari in quel momento in ascolto. Per la comunicazione tra gli utenti si è scelto il protocollo HTTP e la rete Internet in generale: in questo modo, se al server viene assegnato un indirizzo IP pubblico, la comunicazione può avvenire anche se i dispositivi non sono sotto la copertura di una stessa rete wireless e l'affidabilità e la robustezza aumentano.
Il sistema può essere schematizzato come nella figura seguente (figura 4.1). Come si può notare ogni utente ha a disposizione due dispositivi, lo smartwatch Pebble e il proprio smartphone o tablet. Questi ultimi comunicano con il server, che si occupa di ricevere e instradare i vari messaggi dopo opportune elaborazioni, di gestire gli utenti e di interfacciarsi con un database per il salvataggio delle informazioni.

Figura 4.1: Architettura generale del sistema

4.1.1 La modellazione delle entità

Affinché il sistema possa prendere forma occorre definire il contesto in cui esso opera, andando a stabilire quali sono le entità trattate. Questa fase della progettazione è indispensabile anche per la persistenza delle informazioni: la costruzione di un database parte sempre dalla modellizazione delle entità e delle relazioni che intercorrono tra di loro. La base dati comprende quattro diversi concetti:

- *Caregiver*: rappresenta un operatore sanitario nel sistema.
- *Patient*: rappresenta un ospite della RAF nel sistema.
- *PatientAlarm*: modella una richiesta di assistenza da parte di un ospite della RAF.
- *CaregiverAlarm*: modella una richiesta di aiuto da parte di un operatore sanitario.

Un semplice diagramma entità-relazioni che rappresenta la base dati è raffigurato nella figura 4.2.

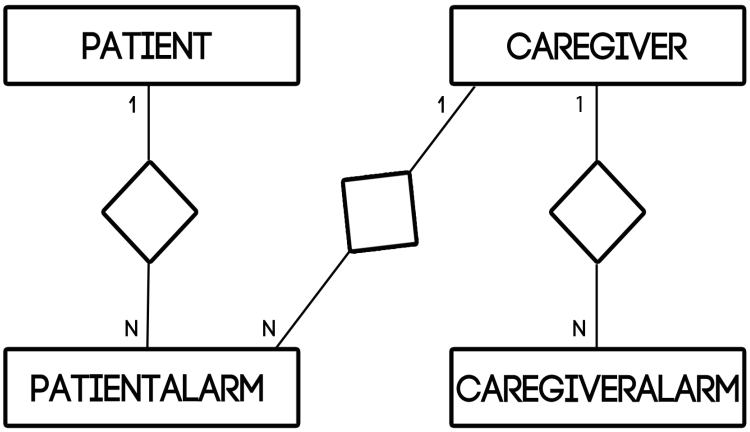


Figura 4.2: Diagramma entità-relazioni del sistema

Nella programmazione ad oggetti questi concetti sarebbero di sicuro raggruppabili secondo l'ereditarietà e il polimorfismo, ma si è preferito mantenerli separati proprio per essere compatibili al massimo con la progettazione del database e mantenere più alte possibili le performance. Una richiesta di assistenza di un ospite è collegata con l'ospite richiedente ed eventualmente con un operatore sanitario che si preoccupa di servirla. Da notare che un ospite può naturalmente richiedere più volte aiuto, così come un assistente può servire più di una richiesta.

L'elemento fondamentale trattato dal sistema è l'assistenza agli ospiti della RAF: nella fase di progettazione si è anche definito come vengono modellate le richieste di aiuto.

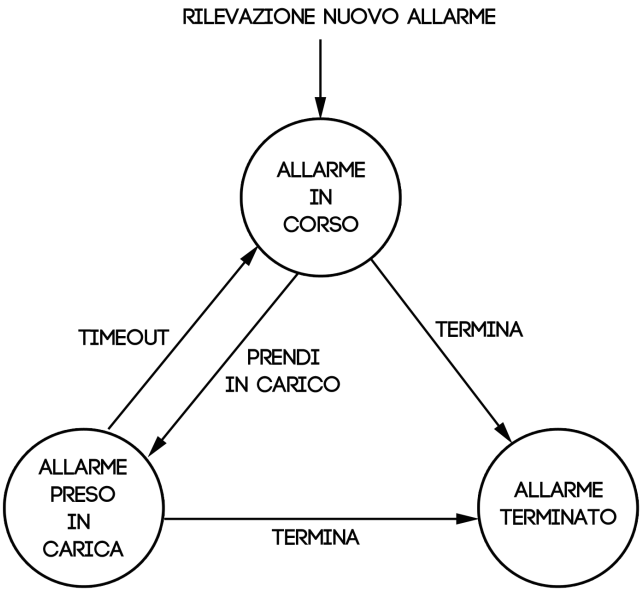


Figura 4.3: Diagramma a stati degli allarmi

Un allarme potrà assumere uno di questi tre stati:

- *Allarme in corso*: una situazione anomala per l'ospite, che sia stata rilevata automaticamente o sia una richiesta manuale, è in corso se nessun operatore sanitario ha preso in carico l'allarme tramite il suo smartwatch né tantomeno nessuno di essi ha terminato la richiesta interagendo con lo smartwatch dell'ospite stesso. Un allarme in corso è notificato continuamente a tutti gli operatori sanitari del turno.

- *Allarme preso in carica*: una richiesta di assistenza è presa in carico quando un operatore sanitario lo notifica tramite il suo smartwatch. Un allarme preso in carica non è terminato, e torna nello stato di allarme in corso dopo un certo periodo di tempo se l'operatore non ne segnala la fine interagendo con lo smartwatch dell'ospite. La notifica agli operatori sanitari di un allarme che è stato preso in carica è temporaneamente disabilitata.

- *Allarme terminato*: una richiesta di assistenza termina quando un operatore sanitario lo segnala tramite lo smartwatch dell'ospite della RAF. Una richiesta terminata non è più notificata a nessun operatore sanitario e risiede solamente nel database per scopi di documentazione.

A seconda delle azioni compiute dagli ospiti della RAF e dagli operatori sanitari sul sistema gli allarmi possono transitare da uno stato ad un altro (figura 4.3). Un allarme ad esempio può essere preso in carica da un operatore sanitario, oppure può ritornare pendente se preso in carica e non terminato entro un certo periodo.

4.1.2 Scenari di funzionamento

Prima di entrare nel merito dei singoli dispositivi è bene esporre i casi d'uso del sistema. Esistono quattro scenari principali in cui il sistema si può trovare: la richiesta di assistenza da parte di un ospite della RAF, la presa in carico di una richiesta di assistenza in corso, la terminazione di una richiesta di assistenza in corso e la richiesta di aiuto da parte degli operatori sanitari stessi. In ognuno di questi casi il sistema segue un ben determinato algoritmo, implementato grazie ai vari componenti presenti.

La richiesta di assistenza

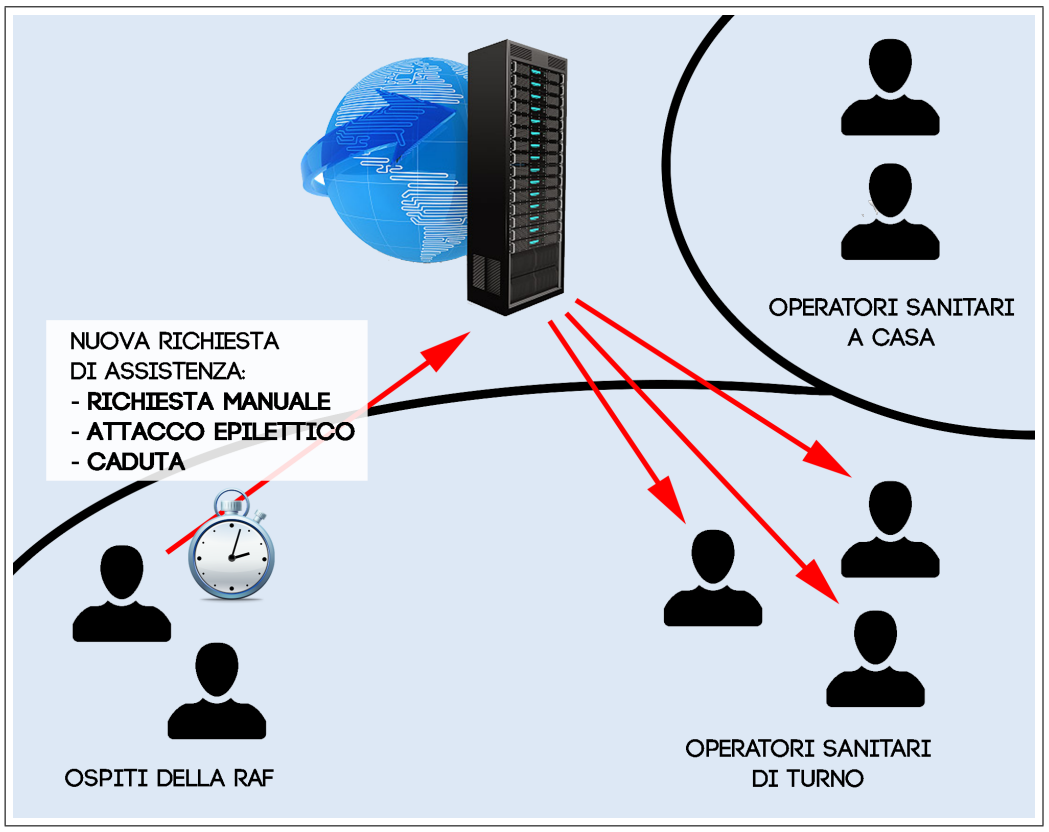


Figura 4.4: La richiesta di assistenza di un ospite della RAF

Il caso relativo a una nuova richiesta di assistenza da parte di un ospite della RAF è forse il più importante. Come si può vedere dalla figura 4.4 la richiesta è inviata al server che si occupa di propagarla a tutti gli operatori sanitari di turno. I dispositivi dell'ospite reimmettono periodicamente nel sistema la stessa richiesta di assistenza, che si propaga nuovamente con le stesse modalità, in modo tale che finché essa è in corso l'attenzione degli assistenti sanitari venga attratta in modo esaustivo (LG 3.2).

La presa in carico

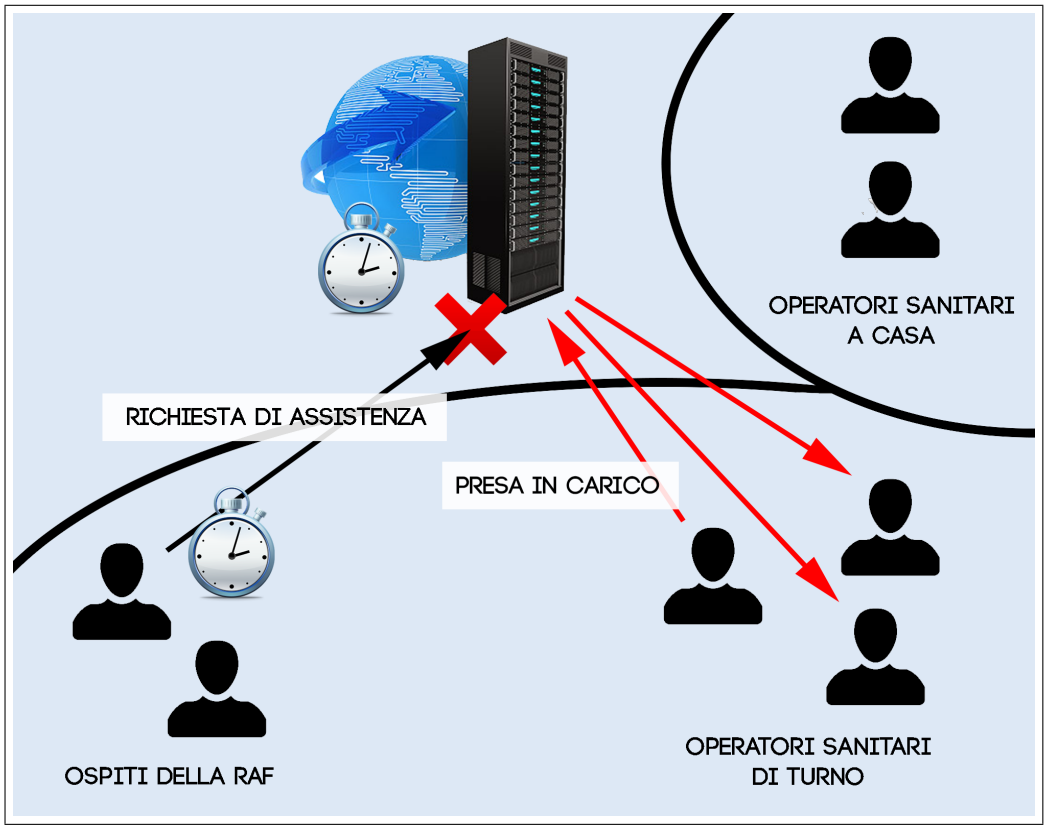


Figura 4.5: La presa in carico di una richiesta di assistenza

Un operatore sanitario, alla presenza di una o più richieste di assistenza in corso, può decidere di prenderle in carico. La notifica passa dal server, che modifica i dati relativi a quell'allarme già esistente e lo comunica a tutti gli altri assistenti, in modo tale che essi possano aggiornare le loro visualizzazioni. Da questo momento in poi, anche se l'ospite continua a inviare periodicamente l'allarme, questo viene scartato dal server, che non lo propaga. L'allarme infatti è passato dallo stato "in corso" a quello "preso in carica" e la sua segnalazione è temporaneamente disabilitata. Come

si può notare dalla figura 4.5 adesso anche sul server è stato istanziato un timer: se dopo un certo periodo di tempo l'allarme "preso in carica" non è ancora passato nello stato "allarme terminato" il server lo fa ritornare nello stato "in corso" e la sua propagazione è riabilitata. Gli operatori sanitari a questo punto torneranno a visualizzare la richiesta.

La terminazione

Quando un operatore sanitario giunge al cospetto dell'ospite che ha richiesto aiuto ha la facoltà di terminare l'allarme in corso. La notifica giunge al server il quale modifica i dati relativi all'allarme esistente. La propagazione di tale richiesta di assistenza è definitivamente disabilitata poiché i dispositivi dell'ospite, accorgendosi che l'allarme è terminato, cessano l'invio disabilitando il timer. La notifica è propagata dal server a tutti gli altri operatori sanitari, che potranno aggiornare le loro visualizzazioni (figura 4.6).

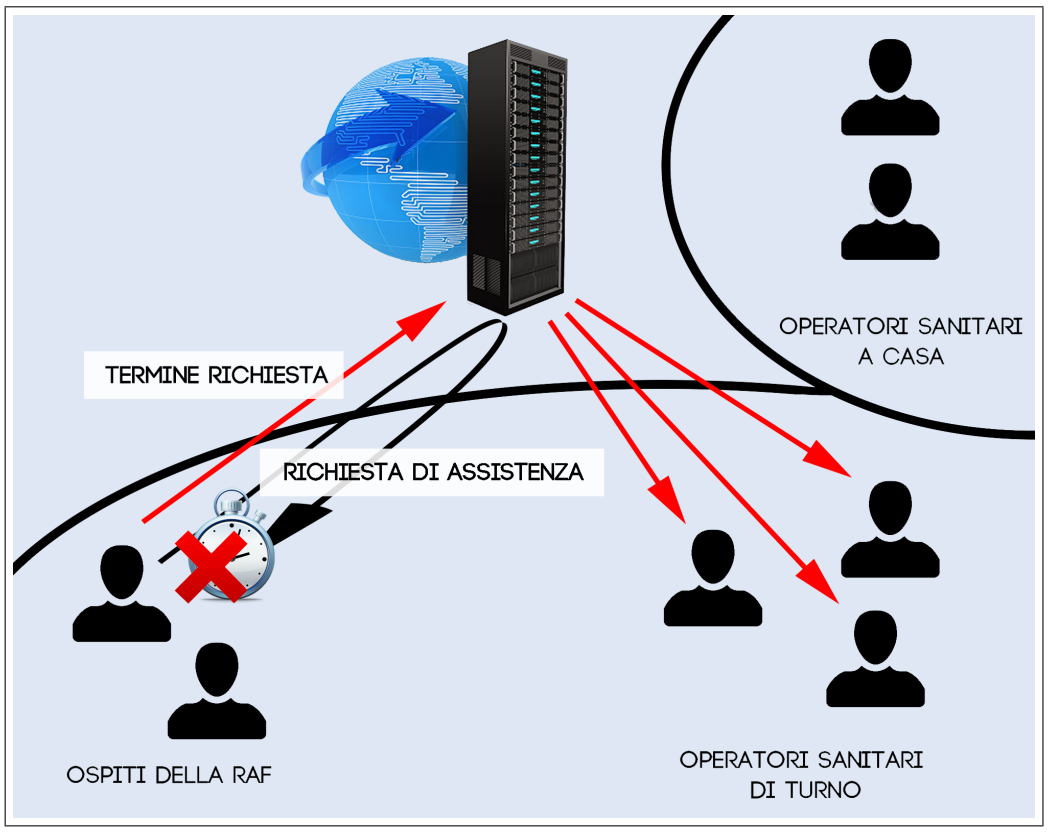


Figura 4.6: La terminazione di una richiesta di assistenza

La richiesta di aiuto degli operatori sanitari

Un operatore sanitario ha egli stesso la possibilità di richiedere aiuto nel caso in cui si trovi in difficoltà. La richiesta raggiunge non solo gli altri assistenti di turno, ma anche coloro che sono a casa, come si può notare dalla figura 4.7.

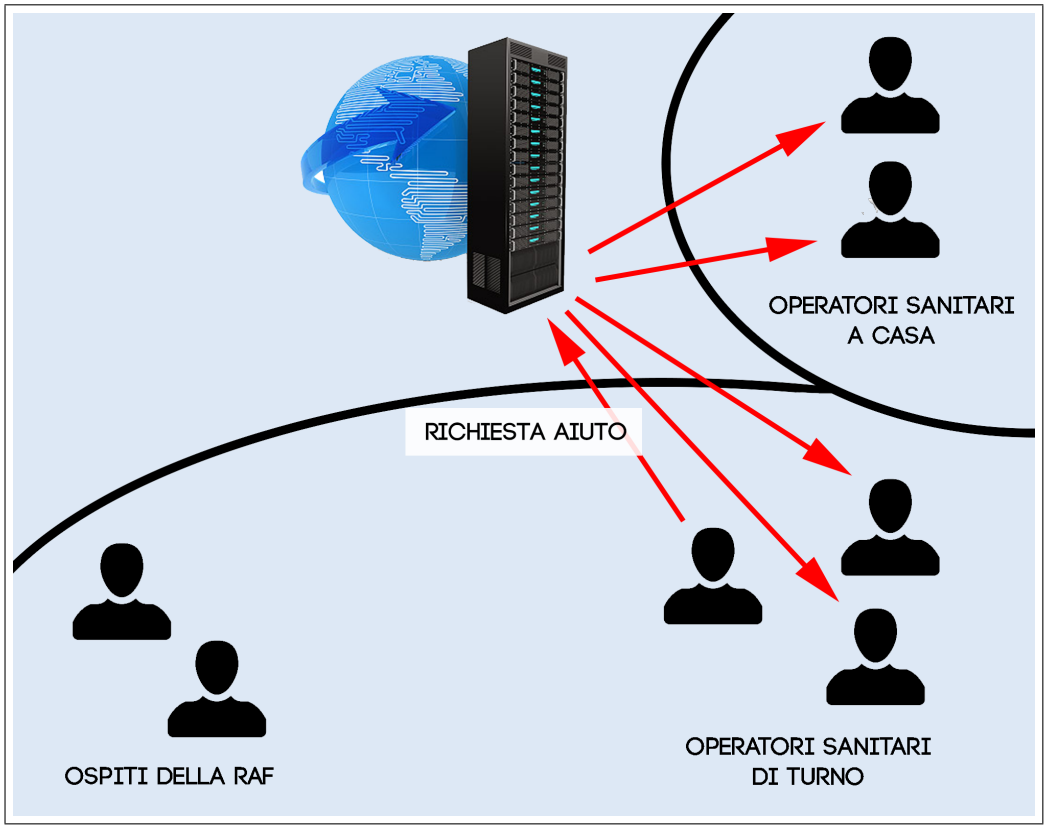


Figura 4.7: La richiesta di aiuto tra operatori sanitari

4.2 Le applicazioni per gli smartwatch Pebble

Le applicazioni presenti sugli smartwatch Pebble sono differenziate in base al ruolo dell'utente, ma entrambe si interfacciano tramite Bluetooth con lo smartphone o il tablet associato per la comunicazione in rete. Queste applicazioni agiscono al bordo del sistema: il loro compito è quello di intercettare eventi, tradurli in messaggi con un opportuno formato condiviso e inoltrarli al dispositivo mobile, che si occuperà di propagarli al resto del sistema. Gli eventi più comuni sono una nuova richiesta di assistenza, la terminazione di una stessa, la presa in carico o la richiesta di aiuto da parte di un operatore.

4.2.1 L'applicazione per l'operatore sanitario

L'applicazione presente sullo smartwatch dell'operatore sanitario deve permettere di:

- Visualizzare le richieste di assistenza in corso, comprendenti il tipo e l'ospite richiedente, e le richieste di assistenza prese in carico dall'operatore, in modo tale che egli non possa dimenticarsene (LG 2.5).
- Ricevere le notifiche del sistema riguardanti lo stato degli allarmi e aggiornarne la visualizzazione.
- Prendere in carico una o più richieste di assistenza in corso.
- Inviare una richiesta di aiuto a tutti gli operatori sanitari, siano essi di turno o a casa.

La progettazione parte con la necessità di mantenere allineati gli operatori sanitari con i dati riguardati gli allarmi degli ospiti della RAF. Posto che lo stato corrente dei dati è mantenuto dal server, il quale si interfaccia al database, occorre studiare come le due liste di informazioni che l'applicazione visualizza (richieste in corso e richieste prese in carico) si aggiornino. Il procedimento adottato è semplice e intuitivo:

- Quando l'applicazione viene avviata si contatta lo smartphone o il tablet di supporto affinché scarichi dal server due diverse liste di allarmi da visualizzare, quelle attualmente in corso e quelle non ancora terminate ma prese in carico dall'operatore.
- L'applicazione rimane in ascolto di nuovi messaggi. Ad ogni nuova notifica ricevuta in merito alle richieste di assistenza (una nuova richiesta, la notifica della terminazione di una esistente ecc.) le due liste vengono aggiornate.

Essendo gli allarmi in corso di fondamentale importanza occorre che se presenti siano continuamente notificati all'operatore, in modo tale che egli non possa ignorarli. Questo deve accadere anche quando l'applicazione viene chiusa: nel caso siano presenti delle richieste di assistenza è necessario prevedere delle notifiche che "risveglino" uno smartwatch in qualunque stato esso si trovi (LG 3.2).

4.2.2 L'applicazione per l'ospite della RAF

L'applicazione presente sullo smartwatch dell'ospite della RAF deve permettere di:

- Rilevare automaticamente, tramite l'accelerometro, situazioni di pericolo per gli ospiti quali cadute e attacchi epilettici.

- Rilevare richieste di assistenza manuali, scatenate con l'interazione consapevole dell'utente con lo smartwatch.
- Inviare le richieste di assistenza a tutti gli operatori sanitari di turno.
- Notificare a tutti gli operatori sanitari di turno che un eventuale allarme in corso è terminato.

Considerando la tipologia di utenti a cui questa applicazione è rivolta è necessario che essa funzioni in modo per lo più automatico, poiché un disabile con gravi problemi fisici potrebbe non essere in grado di interagire volontariamente con lo smartwatch (LG 4.2). Questo non è un grosso problema tenendo conto che le caratteristiche di questa applicazione differiscono molto rispetto a quelle per gli operatori sanitari e si adattano bene a questo scenario: il compito principale è quello di rilevare situazioni di pericolo per un ospite e di comunicarlo al resto del sistema, cosa che può essere fatta senza alcuna interazione volontaria. Le uniche due azioni che impongono la collaborazione di un utente sono le richieste manuali e la segnalazione del termine di un allarme, che tuttavia dovrà essere svolta da un operatore. Proprio quest'ultimo punto impone un vincolo in più da rispettare: non dovrà essere possibile che per sbaglio il paziente stesso termini un allarme. Come modalità di controllo, nel caso si rilevi la tentata terminazione di una richiesta di assistenza, il sistema prevede una scansione Bluetooth da parte dello smartphone dell'ospite alla ricerca nelle vicinanze del dispositivo dell'operatore sanitario che ha preso in carico l'allarme (LG 3.2).

L'uso dell'accelerometro

L'accelerometro è utilizzato per la rilevazione automatica di attacchi epilettici e di cadute degli ospiti della RAF. Come primo compito dunque l'applicazione deve raccogliere i dati provenienti dal sensore. Una scelta si è presentata nel definire dove, ovvero su quale dispositivo, effettuare l'analisi dei dati provenienti dall'accelerometro. Con una potenza di elaborazione elevata sono possibili soluzioni che offrono una affidabilità molto elevata nel riconoscimento dei movimenti: reti neurali, Support Vector Machine, classificatori bayesiani, ecc. Questa strada è stata scartata principalmente per tre motivi. Per prima cosa questi algoritmi di classificazione hanno bisogno di un insieme di dati detto "training set" che questo lavoro non ha a disposizione. Con esso gli algoritmi si istruiscono e costruiscono un modello sul quale basare le successive classificazioni. In secondo luogo si sono considerate le linee guida definite in precedenza: il sistema deve essere a basso costo e non deve introdurre dispositivi superflui. In particolare, volendo utilizzare gli smartphone già in dotazione di operatori sanitari e ospiti delle RAF (LG 2.7), un fattore importante è quello di salvaguardarne la durata della batteria, evitando di creare applicazioni eccessivamente pesanti. Per finire non bisogna dimenticare che la rilevazione di situazioni di pericolo deve essere la più veloce possibile, e come espressamente detto

nei focus group l'affidabilità non è un fattore chiave, ma è ben accetta una soluzione lasca che rilevi qualche richiesta superflua (LG 3.1). Per tutti questi motivi si è deciso che lo smartwatch dell'ospite della RAF, oltre a raccogliere i dati accelerativi, ne compia direttamente l'analisi. Procedere in questo modo porta a dei sostanziali vantaggi. Innanzitutto il riconoscimento è rapido, e non si sovraccaricano altri dispositivi, ma a fare l'elaborazione è il dispositivo con la durata della batteria più lunga. Inoltre si limitano le informazioni trasferite, perché i dati non vengono inviati ad altri dispositivi per l'elaborazione. Chiaramente svolgendosi su un dispositivo con poca potenza di elaborazione l'analisi dei dati si deve basare su un algoritmo semplice, basato su soglie. Esso è così riassumibile:

- Fase 1 - *"Preprocessing dei dati"*: i singoli campioni accelerativi vengono puliti dalla componente gravitazionale e messi insieme in un unica risultante che comprende i tre assi x, y e z.
- Fase 2 - *"Collezionamento dei dati"*: si suddivide la raccolta dati in finestre temporali, andando a salvare le risultanti accelerative dei vari istanti compresi in una opportuna struttura dati.
- Fase 3 - *"Analisi dei dati"*: si esegue l'analisi di ogni finestra temporale, considerando varie caratteristiche del segnale, come l'accelerazione media e il picco.

4.3 Le applicazioni per i dispositivi mobili

Anche le applicazioni per smartphone o tablet si differenziano in base al ruolo dell'utente, seppur in modo minore rispetto a quelle per gli smartwatch Pebble. Esse si possono definire come una "porta" tramite cui gli smartwatch possono comunicare in rete tra di loro. Fungono in sostanza da passacarte tra il server e il dispositivo wearable associato e sono strettamente collegate con l'utente che le sta utilizzando. L'idea è che alla prima accensione l'utente acceda al sistema con uno username e una password che verranno controllati sul server. Da questo momento in poi l'applicazione conosce l'identità dell'utente e la associa ai messaggi provenienti dallo smartwatch e che propagherà in rete.
L'applicazione per un operatore sanitario offre le seguenti caratteristiche:

- Permette la registrazione di un nuovo utente sul server.
- Permette l'accesso al sistema tramite username e password.
- Riceve i messaggi provenienti dallo smartwatch collegato, associa ad essi l'identità dell'operatore e li propaga al server. Dallo smartwatch l'applicazione può ricevere notifiche di presa in carico per determinati allarmi in corso o richieste di aiuto da destinare agli altri operatori sanitari.

- Riceve i messaggi provenienti dal server e li propaga se necessario allo smartwatch associato. Dal server l'applicazione può ricevere messaggi di altri operatori sanitari o di ospiti della RAF. Dei primi riceverà le notifiche di presa in carico di allarmi in corso o richieste di aiuto, dei secondi le richieste di assistenza.

L'applicazione per un ospite della RAF offre le seguenti caratteristiche:

- Permette la registrazione di un nuovo utente sul server.
- Permette l'accesso al sistema tramite username e password.
- Riceve i messaggi provenienti dallo smartwatch collegato, associa ad essi l'identità dell'ospite della RAF e li propaga al server. Dallo smartwatch l'applicazione può ricevere nuove richieste di assistenza oppure notifiche di terminazione di un allarme in corso. Nel caso di richieste di assistenza l'applicazione manda la notifica in continuazione verso il server fino a quando l'allarme non è terminato, in modo tale che l'informazione non possa essere ignorata.
- Permette di eseguire una scansione Bluetooth alla ricerca di un dispositivo di un operatore sanitario quando riceve la notifica di tentata terminazione di un allarme dallo smartwatch associato.

Si è stabilito che l'applicazione in esecuzione su smartphone o tablet, sia essa relativa a un operatore sanitario o a un ospite, debba funzionare per lo più in background (ad esclusione delle schermate per il login e la registrazione), e che le funzionalità non debbano essere interrotte anche se l'applicazione viene chiusa. In questo modo l'utente, dopo aver eseguito l'accesso al primo avvio, non avrà più da preoccuparsi dell'esistenza di tale programma, ma tutto continuerà a funzionare in modo automatico.

4.4 L'applicazione server-side

Il server è uno dei cardini del sistema e i suoi compiti sono numerosi. Ad esso spetta innanzi tutto la gestione della persistenza dei dati in collaborazione con il database, dunque è proprio il server che conosce istante per istante lo stato del sistema. Per questo motivo uno dei suoi punti principali è quello di coordinare la comunicazione tra i vari dispositivi presenti, tenendo traccia degli utenti, aggiornando le loro informazioni e implementando la logica applicativa, ovvero i comportamenti da tenere al verificarsi di determinati eventi. Il server web è costituito da due parti: un sito web amministrativo e un'interfaccia RESTful.
La prima, molto semplice, permette agli operatori sanitari di gestire il sistema. Essi potranno registrare direttamente nuovi utenti e visualizzare la "storia" del sistema,

con gli allarmi terminati, in corso e presi in carico.
La seconda parte, molto più significativa, è quella che permette l'evoluzione del sistema e che coordina tutte le attività. Un dispositivo mobile che voglia inviare in rete un messaggio si rivolge sempre al server, che effettuate le dovute elaborazioni si occuperà di propagarlo ai giusti utenti in rete e nelle opportune modalità. In particolare il servizio RESTful svolge le seguenti funzioni principali:

- Riceve dagli ospiti della RAF messaggi rappresentanti richieste di assistenza. Questi possono essere relativi a nuovi allarmi (dunque il server si dovrà occupare di salvare l'informazione nel database) oppure possono essere relativi ad allarmi già in corso, visto che lo smartphone di un ospite segnala periodicamente le richieste fin quando esse non terminano. In tutti due i casi il server propaga l'informazione a tutti gli operatori sanitari registrati sul sistema e in ascolto.

- Riceve dagli ospiti della RAF messaggi indicanti la terminazione di un allarme in corso o preso in carico, aggiorna la richiesta nel database e inoltra la notifica a tutti gli operatori sanitari registrati sul sistema affinché l'allarme non sia più visualizzato.

- Riceve dagli operatori sanitari messaggi rappresentanti la presa in carico di un allarme in corso, aggiorna la rispettiva richiesta nel database associandogli l'identità dell'assistente e propaga la notifica a tutti gli operatori sanitari registrati sul sistema affinché la visualizzazione dell'allarme sia temporaneamente disabilitata.

- Può far tornare un allarme dallo stato "preso in carica" a quello "in corso" se passato un certo periodo di tempo dalla presa in carico l'allarme non è terminato.

- Riceve dagli operatori sanitari messaggi rappresentanti richieste di aiuto e inoltra la notifica a tutti gli altri operatori sanitari registrati sul sistema.

- Permette di aggiungere un ospite o un operatore sanitario nel database affinché anche dai dispositivi mobili sia possibile effettuare la registrazione al sistema.

Il servizio RESTful mappa su opportune URL le operazioni specifiche da svolgersi sulle entità del sistema. Ricevuta una richiesta il server si occupa di chiamare gli opportuni metodi di servizio, che applicheranno la logica applicativa e agiranno sul database per implementare la persistenza delle informazioni. Data la complessità dei meccanismi coinvolti è stato necessario definire a priori un piano di indirizzamento per le risorse esposte. Nelle tabelle seguenti è tralasciato l'indirizzo base del server, comune a tutte le URL.

Per l'entità che modella gli ospiti della RAF le operazioni esposte sono presentate in tabella 4.1:

Tabella 4.1: Piano di indirizzi per l'ospite della RAF

URL	**Metodo**	**Descrizione**
patients	GET	Ritorna la lista di tutti gli ospiti registrati nel sistema.
patients/{id}	GET	Ritorna un oggetto modellante l'ospite della RAF caratterizzato dall'id passato come parametro dell'URL.
patients/login	POST	Riceve nel corpo della richiesta HTTP un oggetto contente le credenziali di accesso (username e password) di un ospite della RAF e procede a verificarne la correttezza, ritornando in caso di successo l'oggetto che modella l'ospite nel sistema.
patients	POST	Aggiunge un nuovo ospite della RAF nel sistema modellato dall'oggetto presente nel corpo della richiesta HTTP.
patients/{id}	PUT	Modifica l'ospite caratterizzato dall'id passato come parametro nell'URL. L'oggetto modificato rappresentante l'ospite deve essere presente nel corpo della richiesta HTTP.
patients/{id}	DELETE	Cancella dal sistema l'ospite caratterizzato dall'id passato come parametro nell'URL.

Passando agli operatori sanitari, i metodi esposti per l'entità che ne modella il concetto sono presentati in tabella 4.2:

Tabella 4.2: Piano di indirizzi per l'operatore sanitario

URL	**Metodo**	**Descrizione**
caregivers	GET	Ritorna la lista di tutti gli operatori sanitari registrati nel sistema.
caregivers/{id}	GET	Ritorna un oggetto modellante l'operatore sanitario caratterizzato dall'id passato come parametro dell'URL.
caregivers/login	POST	Riceve nel corpo della richiesta HTTP un oggetto contente le credenziali di accesso (username e password) di un operatore e procede a verificarne la correttezza, ritornando in caso di successo l'oggetto che modella l'operatore nel sistema.
caregivers	POST	Aggiunge un nuovo operatore sanitario nel sistema modellato dall'oggetto presente nel corpo della richiesta HTTP.
caregivers/{id}	PUT	Modifica l'operatore sanitario caratterizzato dall'id passato come parametro nell'URL. L'oggetto modificato rappresentante l'operatore deve essere presente nel corpo della richiesta HTTP.
caregivers/{id}	DELETE	Cancella dal sistema l'operatore sanitario caratterizzato dall'id passato come parametro nell'URL.

Nel caso delle richieste di assistenza degli ospiti della RAF i dispositivi mobili possono contattare il server avendo a disposizione i metodi esposti in tabella 4.3:

Tabella 4.3: Piano di indirizzi per le richieste di assistenza degli ospiti

URL	**Metodo**	**Descrizione**
alarms	GET	Ritorna la lista di tutte le richieste di assistenza, in corso e non, presenti sul sistema.
alarms/{id}	GET	Ritorna un oggetto modellante la singola richiesta di assistenza caratterizzata dall'id passato come parametro dell'URL.
alarms/pending	POST	Ritorna tutte le richieste di assistenza attualmente in corso nel sistema.
alarms/taken	POST	Ritorna la lista di tutte le richieste di assistenza attualmente prese in carico nel sistema dall'operatore sanitario il cui oggetto modellante è passato come parametro nel corpo della richiesta HTTP.
alarms	POST	Il metodo riceve nel corpo della richiesta HTTP un oggetto che modella una nuova richiesta di assistenza, contenente anche un riferimento all'ospite che lo ha generato. Se l'ospite della RAF non ha attualmente richieste di assistenza in corso l'oggetto è reso persistente, il nuovo allarme entra a far parte del sistema e una notifica è inviata a tutti gli operatori sanitari di turno.

alarms/{id}	PUT	Utilizzato per l'invio periodico o per la modifica un allarme già presente nel database con i dati ricevuti nel corpo della richiesta HTTP. Nel caso si tratti di una modifica importante per il sistema (termine di una richiesta o presa in carico) la notifica è inviata a tutti gli operatori sanitari di turno, così come se si tratta di una segnalazione di uno stesso allarme non ancora terminato.

Quando il sistema necessita il trattamento di richieste di aiuto da parte degli operatori sanitari i metodi offerti dall'interfaccia REST sono presentati in tabella 4.4:

Tabella 4.4: Piano di indirizzi per le richieste di aiuto degli operatori sanitari

URL	**Metodo**	**Descrizione**
caregiveralarms	GET	Ritorna la lista di tutti le richieste di aiuto presenti sul sistema.
caregiveralarms/{id}	GET	Ritorna un oggetto modellante la singola richiesta di aiuto caratterizzata dall'id passato come parametro dell'URL.
caregiveralarms	POST	Aggiunge nel sistema la nuova richiesta di aiuto ricevuta nel corpo della richiesta HTTP e invia una notifica a tutti gli altri operatori sanitari.

4.5 Il sistema di comunicazione

Tralasciando la semplice comunicazione Bluetooth tra smartwatch e dispositivo mobile, gestita in modo automatico dalle librerie offerte da Pebble, parte della progettazione è stata spesa a definire le modalità con cui smartphone o tablet e server interagiscono. Come si è potuto intuire esistono due situazioni diverse: la comunicazione da dispositivo mobile a server e la comunicazione da server a dispositivo mobile.

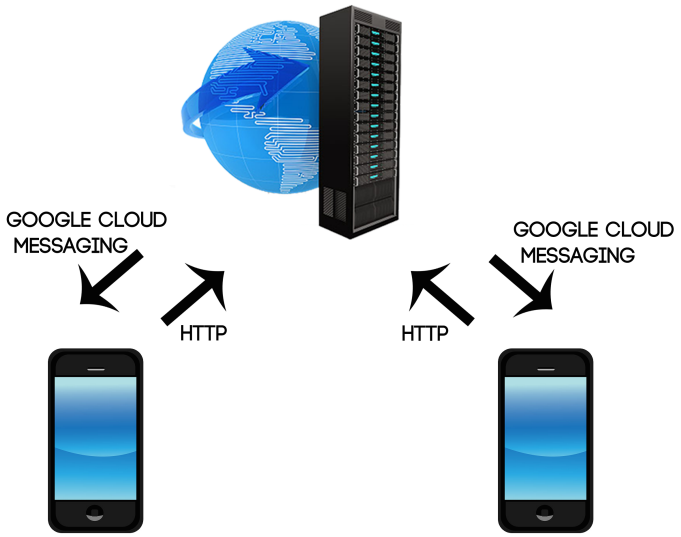


Figura 4.8: Il sistema di comunicazione

La prima modalità è stata la più semplice da progettare, poiché è sempre di tipo "uno a uno" (sono coinvolti un dispositivo e il server) e si adatta perfettamente a uno schema "richiesta - risposta": quando ne ha bisogno, ad esempio al verificarsi di un evento, un dispositivo contatta il server, che svolte le opportune elaborazioni risponderà. Questo ha portato in modo naturale alla scelta del protocollo HTTP, permettendo uno sviluppo ad alto livello e garantendo la massima affidabilità, essendo il protocollo basato su TCP (LG 3.3).
La seconda modalità ha presentato più problemi. Si tratta infatti molto spesso di una comunicazione "uno a molti" (sono coinvolti il server e un insieme di altri dispositivi) che avviene in modalità asincrona, ovvero senza che i client facciano alcuna esplicita richiesta. Si pensi ad esempio al caso di una nuova richiesta di assistenza: il messaggio, arrivato al server, dovrà essere propagato a tutti gli operatori sanitari senza che questi compiano alcuna azione. Per sua natura il protocollo HTTP non offre queste caratteristiche di asincronicità ed è stato necessario appoggiarsi a un servizio di terza parte per implementare queste notifiche, il servizio di Google Cloud

Messaging. Esso gestisce tutti gli aspetti di accodamento dei messaggi e di consegna alle applicazioni client in esecuzione sui dispositivi, ed è completamente gratuito. Con tale servizio si possono spedire messaggi a dispositivi singoli o a gruppi di utenti. Il sistema di comunicazione può essere schematizzato come in figura 4.8, dove si vedono bene i due rami utilizzati, uno per la comunicazione verso il server e uno per le notifiche verso i client.

Capitolo 5

Implementazione

Dopo la progettazione del sistema la prosecuzione naturale è la sua implementazione. Avendo svolto la fase precedente con cura, attraverso linee guida ben definite, questa non ha incontrato particolari difficoltà. Nella prima parte del capitolo verranno ripresi alcuni aspetti generali del sistema, andando a evidenziarne l'implementazione, mentre nelle sezioni successive l'attenzione verrà posta sui singoli componenti del progetto.

5.1 Aspetti generali

Nel capitolo precedente è stata esposta l'architettura del sistema con i vari componenti utilizzati. Si è visto che all'ospite e all'operatore sanitario sono affidati due dispositivi, uno smartwatch Pebble e uno smartphone o un tablet di supporto, mentre al centro del sistema c'è la presenza di un server che coordina le operazioni e gestisce la persistenza delle informazioni in collaborazione con un database MySql. Per ogni tipologia di utente sono state sviluppate due applicazioni: una per lo smartwatch e una per il dispositivo Android associato. Per l'ospite le applicazioni sono state chiamate entrambe *"Health"*, mentre per l'operatore è stato scelto il nome *"Healthcare"*. Dal punto di vista implementativo, come si è già visto nel capitolo 3, sono state utilizzate tecnologie diverse, poiché diversi sono i dispositivi coinvolti. Il server, oltre agli elementi offerti da Java EE (filtri, servlet, lister e pagine JSP), utilizza il framework Jersey per l'implementazione dei servizi REST, e JPA/Hibernate come ORM. Oltre a questo, un aspetto importante è l'utilizzo della libreria *"HealthcareSupportLib"* in comune con le applicazioni Android degli utenti, i cui dettagli verranno resi noti in seguito: tale componente software, appositamente implementato, gestisce la modellazione delle entità scambiate nei messaggi tra i vari dispositivi.

5.1.1 La modellazione delle entità

La caratteristica principale del sistema è lo scambio di messaggi tra i vari dispositivi coinvolti: è un messaggio una nuova richiesta di assistenza di un ospite della RAF, così come la presa in carico di un allarme da parte di un assistente. Queste informazioni scambiate sono sempre relative a un concetto che modella il contesto in cui la comunicazione si svolge: il sistema tratta l'interazione tra due tipologie di utenti, gli operatori sanitari e gli ospiti della RAF, che si possono scambiare informazioni relative a richieste di assistenza. I concetti sono rappresentati dalle entità del sistema e prima di affrontare i dettagli successivi è bene andarle ad osservare più da vicino. Esse hanno una duplice funzionalità perché descrivono sia l'oggetto utilizzato dal codice in memoria che la rappresentazione persistente nella base dati: sarà compito dello strato ORM effettuare il collegamento.

Il database

Partendo dal database la prima entità che viene analizzata è quella che modella gli ospiti della RAF.

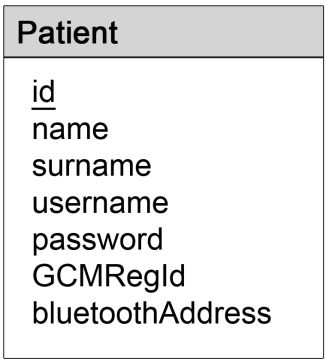


Figura 5.1: L'entità che modella un ospite della RAF

Come si può notare dalla figura 5.1 esiste una chiave primaria che identifica univocamente un ospite nel sistema, il campo *"id"*. La generazione è automatica ed è affidata al DBMS, che ad ogni nuova istanza aggiunta al database associa il corrispondente valore calcolato. I campi *"name"* e *"surname"* rappresentano il nome e il cognome dell'ospite, mentre *"username"* e *"password"* sono i parametri con cui un utente può accedere al sistema. In particolare il campo *"username"* ha un vincolo di univocità: non è plausibile avere due persone che accedano al servizio avendo uno stesso identificativo nominale. Per finire ci sono due campi i cui scopi sono più specifici, ovvero *"GCMRegId"* e *"bluetoothAddress"*. Il primo contiene un codice di registrazione che le applicazioni client utilizzano per poter usufruire del servizio di Google Cloud Messaging, che come già visto è usato per implementare le notifiche

asincrone, mentre il secondo rappresenta l'indirizzo Bluetooth (univoco a livello globale e immutabile) del dispositivo mobile, utile per eseguire ricerche di prossimità tra gli utenti.
La seconda entità analizzata, come si può vedere in figura 5.2, è quella relativa agli operatori sanitari.

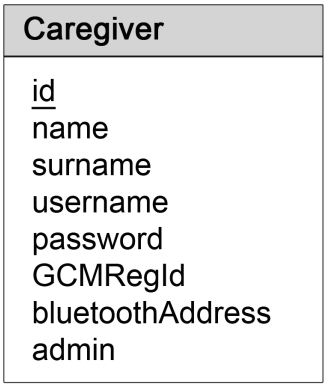


Figura 5.2: L'entità che modella un operatore sanitario

I campi sono esattamente gli stessi di quelli di un ospite della RAF, con l'aggiunta di *"admin"*, che permette di identificare gli operatori sanitari che sono anche amministratori e hanno più privilegi nella gestione del sistema.
La terza entità analizzata, forse quella più coinvolta nello scambio di messaggi, modella le richieste di assistenza da parte degli ospiti della RAF (figura 5.3). Anche in questo caso c'è la presenza di un identificativo univoco generato dal DBMS che viene usato come chiave primaria. Il campo *"type"* identifica invece il tipo di richiesta: un allarme può essere relativo a una caduta, a un attacco epilettico o a una richiesta manuale da parte dell'ospite. *"Ongoing"* è un valore booleano che indica se la richiesta è ancora in corso, e dunque va notificata in continuazione a tutti gli operatori sanitari, oppure se è terminata. Il campo *"patient_id"* è una chiave esterna che collega l'istanza all'utente ospite che l'ha generata. Come visto dal diagramma entità-relazioni la relazione è di tipo uno a molti: un inquilino può generare più richieste, ma una richiesta appartiene a un solo inquilino. Di notevole importanza è l'attributo *"caregiver_id"*, che rappresenta la chiave esterna della relazione uno a molti verso l'operatore sanitario che ha preso in carica la richiesta. La differenza con l'attributo relativo all'ospite è che questa relazione è opzionale e il campo può non essere valorizzato (può dunque contenere il valore NULL): non è detto che l'allarme verrà preso in carica, e di sicuro alla sua creazione il collegamento non può ancora essere esplicitato. Gli ultimi tre campi, *"timestamp"*, *"timestamp_taken"* e *"timestamp_end"* rappresentano rispettivamente l'istante di creazione della richiesta nel database, l'istante in cui eventualmente viene presa in carico da un operatore sanitario e l'istante in cui viene terminata. Avendo a disposizione tutti questi tempi

si potranno monitorare i dati raccolti, calcolando informazioni come il tempo medio di presa in carico e di terminazione degli allarmi, due elementi che potranno tornare utili agli operatori sanitari per valutare l'evolversi del loro lavoro e l'efficienza del sistema adottato. Riassumendo una richiesta di assistenza si trova in stati diversi a seconda della situazione dei suoi attributi:

- L'allarme è *"in corso"* se i campi *"caregiver_id"* e *"timestamp_taken"* non contengono informazioni e se il campo *"ongoing"* ha valore *"true"*.
- L'allarme è *"preso in carico"* se i campi *"caregiver_id"* e *"timestamp_taken"* contengono informazioni valide, ma il campo *"ongoing"* ha ancora valore *"true"*.
- L'allarme è *"terminato* se il campo *"ongoing"* ha valore *"false"*.

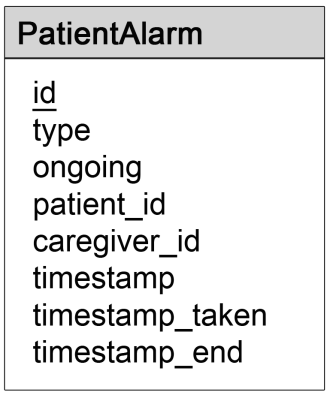


Figura 5.3: L'entità che modella una richiesta di assistenza da parte di un ospite della RAF

L'ultima entità da descrivere è quella relativa alle richieste di aiuto degli operatori sanitari (figura 5.4). Come si può vedere l'entità ha pochi attributi poiché la gestione di questo aspetto del sistema è molto semplice: quando un operatore richiede aiuto viene mandato un unico messaggio a tutti gli assistenti registrati. Come nei casi precedenti il campo *"id"* è la chiave primaria, il campo *"caregiver_id"* rappresenta la chiave esterna verso l'operatore che ha generato la richiesta e *"timestamp"* indica l'istante in cui l'utente ha richiesto aiuto.

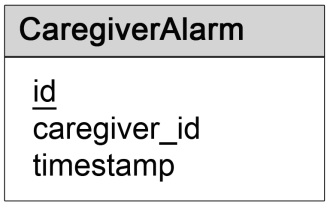


Figura 5.4: L'entità che modella una richiesta di aiuto da parte di un operatore sanitario

Il mondo ad oggetti

Il sistema fa uso di uno strato software intermedio (JPA implementato con Hibernate) che si occupa di mantenere allineata la base dati con la rappresentazione in memoria degli oggetti. Per fare questo occorre che nell'applicazione web del server siano presenti delle opportune classi Java mappate con la corrispondente rappresentazione persistente (la tabella del database in sostanza).

```
21 @Entity
22 public class Patient{
23
24     @Id
25     @GeneratedValue(strategy=GenerationType.AUTO)
26     private Long id;
27     private String name;
28     private String surname;
29     @Column(unique=true)
30     private String username;
31     private String password;
32     private String GCMRegId;
33     private String bluetoothAddress;
34
35     @ManyToOne
36     @JoinColumn(name = "assistedLiving")
37     private AssistedLiving patientLiving;
38
39     @OneToMany(mappedBy = "patient")
40     private List<PatientAlarm> resquests = new LinkedList<PatientAlarm>();
41
42     /**
43      * Method for getting the id of the user
44      *
45      * @return the id of the user
46      */
47     public Long getId() {
48         return id;
49     }
50
51     /**
52      * Method for getting the name of the user
53      *
54      * @return the name of the user
55      */
56     public String getName() {
57         return name;
58     }
59     //[....]
```

Figura 5.5: La classe Java che modella un ospite della RAF

Le istanze di tali classi saranno coinvolte nello scambio di messaggi: non solo il server dovrà conoscerle, ma anche tutti i dispositivi mobili. Per tale ragione è stata creata una libreria chiamata *"HealthcareSupportLib"*, in modo tale che eventuali modifiche siano concentrate sempre in un unico punto. Essa contiene le quattro

entità presenti nella base dati sotto forma di classi Java rappresentanti oggetti Plain Old Java Object (POJO) opportunamente annotati. I POJO utilizzati sono oggetti Java standard che non estendono altre classi, non implementano interfacce, sono privi di costruttore e hanno i metodi *"get"* e *"set"*[1] per tutti i loro attributi. Questi ultimi ricalcano fedelmente le colonne delle tabelle della base dati, e sono mappati in modo automatico dall'ORM in base al nome e al tipo: in presenza di ambiguità opportune annotazioni vengono in aiuto del programmatore. Oltre a questi campi sono presenti anche gli attributi di tipo Collection utilizzati per mappare le relazioni (sempre tramite annotazioni). Ad esempio l'oggetto che modella un ospite della RAF avrà un attributo *"requests"* che conterrà l'elenco di tutte le sue richieste di aiuto. L'importante è l'annotazione *"@Entity"* prima del nome della classe: con essa la classe è mappata con la sua rappresentazione persistente, e le sue istanze corrispondono alle singole righe della tabella. Nella figura 5.5 è mostrata parte della classe *"Patient"*. Si vedono bene i campi corrispondenti alle colonne della tabella nella base dati e gli attributi che mappano le relazione con le richieste di assistenza e con la residenza.

5.1.2 La modellazione dei messaggi

Le entità descritte nella sezione precedente sono il contenuto principale dei messaggi scambiati tra i vari dispositivi che compongono il sistema. La loro rappresentazione va allora tradotta in opportuni formati condivisi affinché l'informazione possa essere propagata nel sistema. Esistono tre diverse modalità:

- Tra smartwatch e dispositivo mobile associato, dove la comunicazione avviene tramite Bluetooth, i messaggi sono modellati tramite coppie chiave-valore.
- Tra dispositivi mobili e server, quando è utilizzato il protocollo HTTP, i messaggi contengono direttamente le istanze della libreria *"HealthcareSupportLib"* tradotte in formato JSON[2].
- Tra server e dispositivi mobili, quando vengono inviate notifiche asincrone tramite GCM, i messaggi sono modellati tramite coppie chiave-valore.

In questa sezione vengono esposte le tecniche adottate per lo scambio dei messaggi, osservando il caso d'uso della richiesta di assistenza.

---

[1] I "getters" e i "setters" in java sono metodi utilizzati per recuperare e manipolare gli attributi privati della classe

[2] JSON, acronimo di JavaScript Object Notation, è un formato adatto all'interscambio di dati fra applicazioni client-server. È basato sul linguaggio JavaScript Standard ECMA-262 3ª edizione dicembre 1999, ma ne è indipendente. Modella i messaggi come un insieme di coppie nome/valore.

La richiesta di assistenza

Il processo di richiesta assistenziale è iniziato da uno smartwatch di un ospite della RAF che invia un opportuno messaggio allo smartphone associato tramite una coppia chiave-valore dove la chiave è una costante che identifica la richiesta di aiuto e il valore identifica il tipo di allarme (figura 5.6).

Figura 5.6: La richiesta di assistenza tra smartwatch dell'ospite e dispositivo mobile associato

Il dispositivo mobile incapsula l'informazione ricevuta (il tipo di allarme) in un nuovo oggetto *"PatientAlarm"*, a cui aggiunge anche l'identità dell'utente (figura 5.7). Il tutto è tradotto automaticamente in formato JSON tramite un'opportuna libreria e inviato al server, che risponderà con lo stesso oggetto a cui avrà aggiunto l'identificativo univoco: il dispositivo potrà salvarsi l'allarme pendente e riferirvisi successivamente. Il server, dopo aver reso persistente l'allarme e prima di aver risposto al client, invia la notifica a tutti i dispositivi mobili degli operatori sanitari tramite GCM attraverso una coppia chiave-valore il cui dato è sempre l'oggetto *"PatientAlarm"* tradotto in JSON (figura 5.8). Il messaggio transiterà prima dal server di terza parte di Google, per essere poi spedito ai vari componenti interessati.

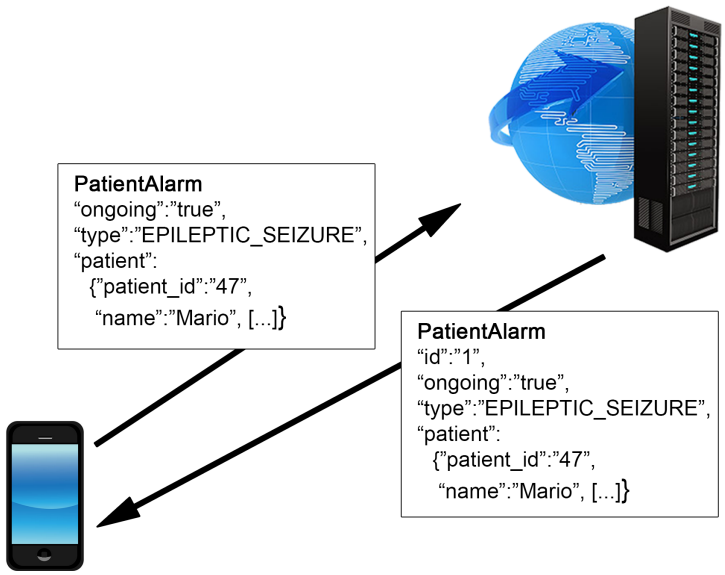


Figura 5.7: La richiesta di assistenza tra dispositivo mobile dell’ospite e server

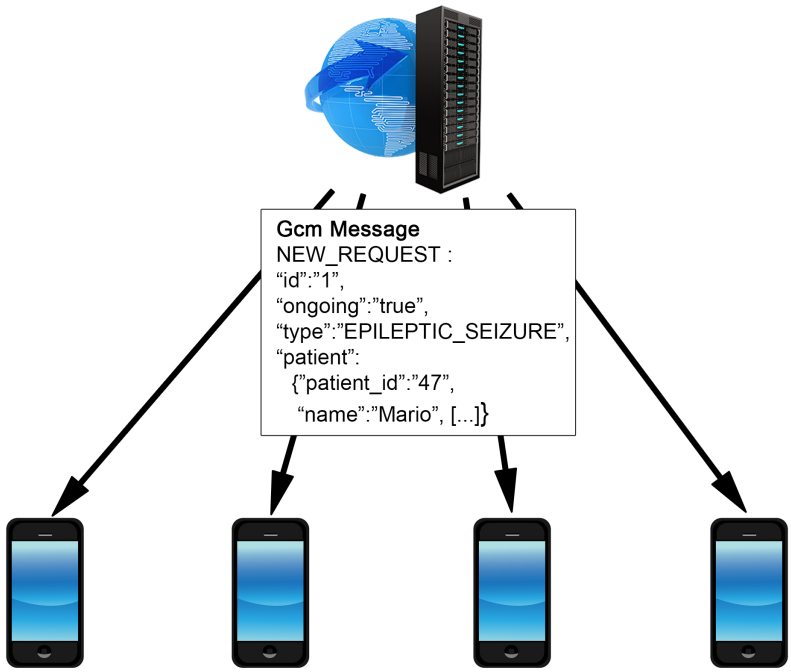


Figura 5.8: La richiesta di assistenza tra server e dispositivi mobili degli operatori

Il dispositivo mobile di un operatore sanitario, ricevuta la notifica da GCM, traduce le informazioni ricevute nelle coppie chiave-valore conformi alla comunicazione con lo smartwatch Pebble associato (figura 5.9). In particolare all'orologio vengono inviati l'id dell'allarme, il tipo, il nome dell'ospite e l'id dell'ospite. Attraverso l'identificativo dell'allarme l'operatore sanitario avrà la possibilità di effettuare la presa in carico, riferendosi direttamente alla richiesta esistente nel database, mentre con il nome dell'ospite e la tipologia della richiesta di aiuto l'applicazione sullo smartwatch potrà visualizzare la notifica e permetterà all'assistente di capire dove la sua presenza è necessaria.

Figura 5.9: La richiesta di assistenza tra dispositivi mobili degli operatori e smartwatch associati

5.2 Il web server Java

Il server è stato implementato interamente come una applicazione web Java Enterprise Edition ed è suddiviso in due parti. La prima parte, la più importante, è quella che permette al sistema di funzionare correttamente, coordina tutte le comunicazioni e applica le regole di comportamento adeguate a seconda delle azioni degli utenti: è ad essa che tutti i dispositivi mobili si rivolgono quando hanno la necessità di comunicare in rete. Questa parte, descritta nella prossima sezione, è stata pensata come un insieme di servizi RESTful, uno per ogni entità modellata dal sistema. La seconda parte del server ospita un semplice sito web di amministrazione che gli operatori sanitari possono utilizzare per gestire il sistema.
Tutte e due le sezioni offerte sfruttano i benefici che si hanno nello sviluppo tramite il paradigma *"Model-View-Controller"* e si appoggiano su un unico strato di servizio che offre i metodi che governano il sistema e applicano la logica applicativa sulle varie entità coinvolte.

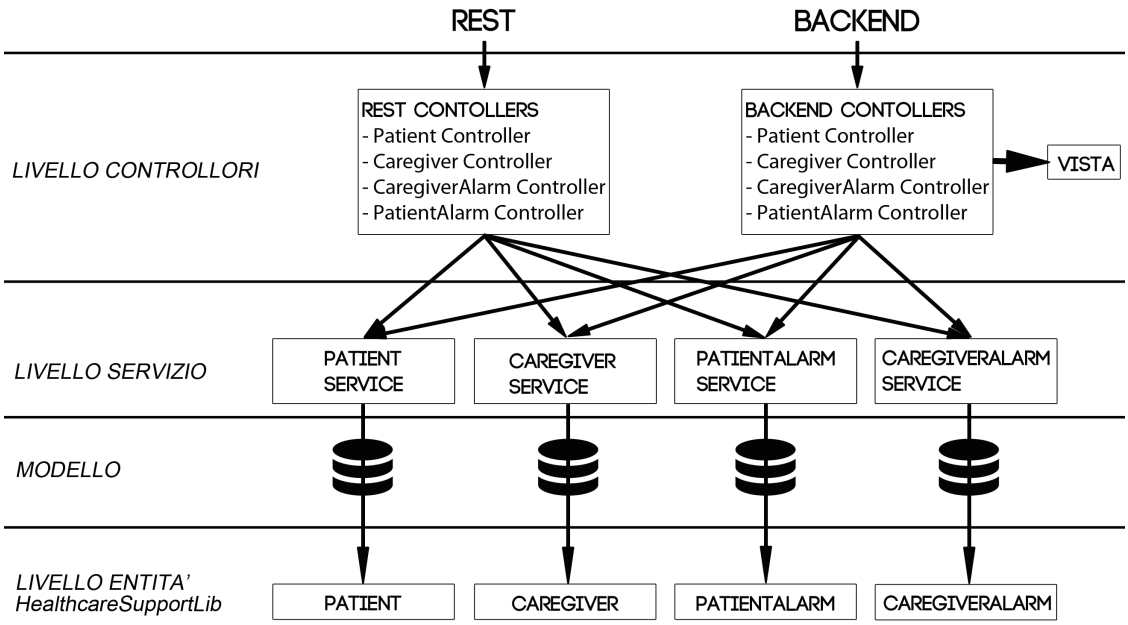


Figura 5.10: La struttura implementativa del server

La figura 5.10 mostra l'organizzazione finale del server: i controllori dell'interfaccia REST e del sito web amministrativo si basano su una stessa struttura comune, formata dal livello dei servizi e da quello delle entità. Questi livelli inferiori, uniti a quello relativo all'accesso alla base dati che in questo progetto è rappresentato da JPA e Hibernate, formano il modello dell'applicazione. I controllori sono diversi perché offrono caratteristiche diverse: quelli relativi al sito web amministrativo

devono ad esempio selezionare le viste da mostrare all'utente dopo aver agito sul servizio, mentre quelli REST non hanno questa esigenza.

5.2.1 L'interfaccia REST

A questa parte del server i dispositivi degli utenti possono rivolgersi per richiedere informazioni sullo stato del sistema o per comunicarne variazioni agli altri componenti. Le richieste vengono fatte a uno specifico URL che come visto nel capitolo precedente è mappato su un ben definito sottoinsieme di elementi trattati dal sistema. Quando viene contattato, il controllore che intercetta la richiesta agisce sui dati utilizzando i metodi di servizio, risponde in modo sincrono al client richiedente con il risultato delle operazioni tramite il protocollo HTTP e ha la possibilità di contattare gli altri utenti in modo asincrono, ovvero senza ricevere alcuna loro richiesta, tramite il servizio di Google Cloud Messaging. Adottare il modello REST presuppone che lo sviluppo sia privo di stato, dunque il server non mantiene informazioni tra una richiesta e l'altra. Questo implica ad esempio che affinché una richiesta di assistenza in corso sia segnalata in continuazione a tutti gli operatori sanitari in ascolto serve il contributo dei dispositivi dell'ospite della RAF, che devono inviare in successione e per più volte la stessa informazione. Il procedimento che per la maggior parte delle operazioni esposte dai servizi REST viene adottato (soprattutto per i metodi POST e PUT) è il seguente:

- Il server riceve una richiesta HTTP contente nel corpo un oggetto rappresentante una entità da creare o da modificare. Spesso questa possiede solo un sottoinsieme dei suoi attributi valorizzati. Si pensi ad esempio a una nuova richiesta di assistenza: la generazione dell'identificativo univoco è lasciata al DBMS e il client quando ne richiede la creazione non può ancora conoscerlo.

- Effettuate le opportune elaborazioni, che possono comprendere il salvataggio nel database, la modifica, l'invio di notifiche tramite GCM e altro, il server risponde al client con lo stesso oggetto ricevuto eventualmente valorizzando altri suoi attributi, in modo tale che il client richiedente possa adottare comportamenti adeguati e capire quale sia l'evolversi dello stato del sistema.

Analizzando che cosa succede in presenza di richieste di assistenza il procedimento risulterà più chiaro.

La gestione delle richieste di assistenza

Il server espone due indirizzi principali per la gestione di richieste di assistenza.
Il primo (*"http://serverName:[port]/alarms*), mappato su metodo POST, viene utilizzato da un dispositivo mobile di un ospite della RAF che voglia immettere in rete un nuovo allarme. Questo viene fatto inserendo un oggetto di tipo *"PatientAlarm"*

nel corpo della richiesta HTTP. La nuova istanza non può avere tutti gli attributi valorizzati, ma sicuramente conterrà il tipo di allarme e il riferimento alla persona richiedente. Il server per prima cosa controlla nel database se l'utente ha attualmente altre richieste di assistenza in corso. Se questo non è vero viene eseguito il salvataggio, inviando l'informazione a tutti gli operatori sanitari di turno tramite GCM. Al client invece il server risponde con lo stesso oggetto valorizzato con l'ulteriore informazione dell'identificativo univoco: il dispositivo lo potrà salvare e utilizzare per riferirsi successivamente allo stesso allarme, ad esempio per la segnalazione periodica. In presenza di una nuova richiesta infatti lo smartphone o il client istanzia un timer deputato al reinvio periodico dell'allarme.
Il secondo URL (*"http://serverName:[port]/alarms/*{*id*}), mappato su metodo PUT, è utilizzabile sia dall'operatore sanitario che dall'ospite e permette di:

- Prendere in carico un allarme in corso: l'operatore sanitario, avvertito della presenza di richieste di assistenza, può decidere di prenderle in carico e avviarsi a prestare aiuto.

- Terminare un allarme: lo stesso operatore sanitario, dallo smartwatch dell'ospite, può terminare una richiesta. Prima di contattare il server il dispositivo mobile associato dell'ospite deve assicurarsi che a compiere l'operazione sia però effettivamente l'operatore.

- Inviare una segnalazione periodica di un allarme in corso: come visto una richiesta di assistenza deve attirare in modo esaustivo l'attenzione di un operatore sanitario, che non deve poter ignorare la notifica. Il dispositivo mobile del paziente reimmette allora periodicamente in rete la stessa richiesta fintanto che essa non termini.

Anche in questo caso il server riceve un oggetto modellante una richiesta di assistenza nel corpo della richiesta HTTP. Questa volta però ci sarà la presenza dell'identificativo univoco associato: il client si sta riferendo a un allarme già esistente. I primi due casi sono abbastanza intuitivi. Se la richiesta è ricevuta da un operatore sanitario e l'oggetto ricevuto contiene gli attributi della presa in carico (*"caregiver_id"* e *"timestamp_taken"*) valorizzati il rispettivo allarme nel database è modificato e una notifica viene inviata a tutti gli altri operatori di turno. Un comportamento simile è ottenuto se l'oggetto contiene il campo *"ongoing"* valorizzato con il valore *"false"* (richiesta che proviene dal dispositivo di un ospite): l'allarme è terminato e l'informazione è propagata nel sistema tramite GCM agli operatori sanitari. Più complicata è la gestione dell'ultimo caso. Il server interpreta come segnalazione periodica una richiesta se l'oggetto ricevuto possiede l'identificativo univoco, ma non contiene gli attributi visti in precedenza. Per prima cosa viene controllata lo stato della corrispondente versione persistente, ovvero presente nel database. Si possono presentare tre casi principali:

- L'allarme ricevuto in realtà è stato preso in carico: il server non invia la notifica agli operatori sanitari di turno, e rimanda al client lo stesso oggetto. La segnalazione è infatti temporaneamente disabilitata, ma l'allarme infatti non è ancora terminato ed è lecito che l'ospite continui a segnalarlo.

- L'allarme ricevuto in realtà è terminato: il server non invia la notifica agli operatori sanitari di turno, e risponde al client con l'allarme ricevuto modificato con il campo che ne indica la fine. Il client a questo punto disabiliterà il timer di reinvio e non segnalerà più quella richiesta.

- L'allarme ricevuto è effettivamente in corso: la notifica è inviata a tutti gli operatori sanitari di turno.

Nel caso in cui l'allarme sia preso in carico, il server esegue ancora un'operazione importante: dopo aver ricevuto un numero elevato di segnalazioni, se la richiesta non è terminata viene riportata nello stato *"in corso"* e alla successiva notifica periodica verranno ripresi i contatti con tutti gli altri operatori sanitari di turno.
Analizzando il codice si possono meglio capire i procedimenti seguiti. Per quanto riguarda i nuovi allarmi il metodo esposto per gli ospiti della RAF ottiene dapprima l'istanza del servizio che offre i metodi relativi alle richieste di assistenza tramite una classe factory, invoca il metodo per l'aggiunta di un nuovo allarme nel sistema (l'oggetto *"PatientAlarm"* ricevuto come parametro e inserito nel corpo della richiesta dal client) e risponde con codice 201 CREATED inserendo nel corpo della risposta lo stesso oggetto valorizzato con gli attributi derivanti dall'interazione con il database (in questo caso ad esempio la richiesta ritornata conterrà sicuramente l'identificativo univoco). In caso di errore il metodo ritorna al client il codice 500 INTERNAL SERVER ERROR, che sta ad indicare un errore avvenuto sul server.

REST: il metodo esposto per l'aggiunta di nuove richieste di assistenza

```
@POST
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
public Response addAlarm(PatientAlarm alarm){
   try{
      PatientAlarm databaseAlarm =
          JPAUtil.getPatientAlarmService().addAlarm(alarm);
      return Response.status(201).entity(databaseAlarm).build();
   }
   catch(Throwable t){
      return Response.status(500).build();
   }
}
```

Il metodo del servizio deputato ad aggiungere un nuovo allarme nel servizio è abbastanza intuitivo. Per prima cosa vengono ottenuti gli elementi necessari a interagire con la base dati: l'oggetto *"EntityManager"*, che rappresenta la singola sessione di lavoro con il database, e l'oggetto *"EntityTransaction"*, che rappresenta una transazione (da utilizzarsi in questo caso poiché le operazioni non sono di sola lettura, ma comprendono un inserimento). Viene poi controllato che l'ospite non abbia già un allarme pendente: in questo caso è inutile inserirne un'altro, essendoci già una segnalazione in corso, ma per ragioni di robustezza la richiesta di assistenza già presente è di nuovo segnalata a tutti gli operatori sanitari tramite GCM (`GCM.sendNewAlarmNotification(pending)`). Se invece l'utente era in uno stato "sicuro" l'oggetto ricevuto è reso persistente attraverso JPA (`em.persist(alarm)`). In questo modo all'allarme viene automaticamente associato un id univoco calcolato dal DBMS, con le modifiche che verranno rese effettive al commit della transazione (`et.commit()`). Dopo aver inviato la notifica a tutti gli operatori sanitari tramite GCM il metodo ritorna lo stesso oggetto aggiunto al database.

Il metodo di servizio per l'aggiunta di nuove richieste di assistenza

```
public PatientAlarm addAlarm(PatientAlarm alarm) throws Exception{
   EntityManagerFactory emf = JPAUtil.getSessionFactory();
   EntityManager em = emf.createEntityManager();
   try{
      EntityTransaction et = em.getTransaction();
      try{
         et.begin();
         PatientAlarm pending = null;
         try{
            pending = (PatientAlarm) em.createQuery("from PatientAlarm a
                where ongoing = true and patient=:p",PatientAlarm.class)
                  .setParameter("p", alarm.getPatient()).getSingleResult();
         }
         catch(Exception notFound){}
         if(pending == null){
            //L'ospite non ha richieste in corso -> salvataggio
            em.persist(alarm);
            et.commit();
            //Notifica agli operatori sanitari
            GCM.sendNewAlarmNotification(alarm);
            return alarm;
         }
         else{
            //L'ospite ha già una richiesta in corso
            et.commit();
            //Notifica agli operatori sanitari
```

```
            GCM.sendNewAlarmNotification(pending);
            return pending;
         }
      }
      catch(Exception e){
         et.rollback();
         throw e;
      }
   }
   finally{
      em.close();
   }
}
```

Il metodo esposto da utilizzarsi quando un client (ospite o operatore sanitario) voglia riferirsi ad un allarme esistente è mappato come già detto su metodo PUT. Anche in questo caso viene recuperata per prima cosa l'istanza del servizio che offre i metodi relativi alle richieste di assistenza tramite una classe factory. Successivamente è invocato il metodo per la modifica di un allarme esistente nel sistema (l'oggetto *"PatientAlarm"* ricevuto come parametro e inserito nel corpo della richiesta dal client, contenente in questo caso un identificativo univoco). Il server risponde con il codice 200 OK se l'elaborazione va a buon fine, altrimenti sono possibili due scenari: se l'allarme a cui il client si riferiva non esiste viene ritornato il codice 404 NOT FOUND, che sta a significare che l'oggetto non è stato trovato, altrimenti è ritornato il codice 500 INTERNAL SERVER ERROR, relativo ad un errore sul server.
Il metodo di servizio che si riferisce a un allarme esistete è articolato perché deve distinguere diversi casi. Anche qui vengono ricavati gli oggetti *"EntityManager"* e *"EntityTransaction"* per operare sul database. Dopo questo si va a cercare la corrispondente versione persistente della richiesta a cui il client si riferisce attraverso il suo id (`PatientAlarm dbAlarm = em.find(PatientAlarm.class, alarmId)`). Se esiste si aprono molteplici casi. Per prima cosa si controlla se l'allarme non sia già terminato a insaputa del client (`dbAlarm.getOngoing()== false` dove *"dbAlarm"* è l'oggetto recuperato dal database). Se è così è ritornato l'allarme stesso senza ulteriori modifiche: il client si accorgerà della fine tramite l'opportuno attributo dell'oggetto. Alternativamente sono poi controllate le altre possibilità. Un ospite può aver richiesto la terminazione (`alarm.getOngoing()== false` dove *"alarm"* è l'oggetto ricevuto), oppure un operatore può aver richiesto la presa in carico (`alarm.getCaregiver()!= null`): lo stato della richiesta di assistenza viene aggiornato e una notifica è inviata agli operatori sanitari tramite GCM, tranne nel caso in cui si richieda la presa in carico di un allarme già sotto il controllo di un operatore (`dbAlarm.getCaregiver()!= null`). L'ultima possibilità è che un ospite invii una segnalazione periodica di una

richiesta di assistenza, caso che si verifica quando nessuna delle precedenti condizioni ha il sopravvento. Sono possibili due situazioni. Se l'allarme nel database non è associato a nessun operatore sanitario (`dbAlarm.getCaregiver()== null`) il metodo invia semplicemente la notifica a tutti gli operatori sanitari tramite GCM: la richiesta non è presa in carico e va continuamente segnalata. Altrimenti se la segnalazione si riferisce a un allarme non terminato, ma preso in carica, il server non propaga la notifica agli operatori sanitari perché la richiesta è sotto il controllo di un operatore, ma esegue il controllo che assicura che se l'allarme non termina entro un certo tempo ritorni pendente (`diff > SettingsUtil.getSettings().getInterval()`). Il tempo di timeout è definibile dal sito web amministrativo, in modo tale da offrire agli operatori sanitari la possibilità di regolare la procedura di evasione delle richieste secondo le loro esigenze.

REST: il metodo di servizio per operare con richieste di assistenza esistenti

```
public PatientAlarm updateAlarm(Long alarmId, PatientAlarm alarm) throws
    Exception{
   EntityManagerFactory emf = JPAUtil.getSessionFactory();
   EntityManager em = emf.createEntityManager();
   try{
      EntityTransaction et = em.getTransaction();
      try{
         et.begin();
         PatientAlarm dbAlarm = em.find(PatientAlarm.class, alarmId);
         if(dbAlarm != null){
            if(dbAlarm.getOngoing() == false){
               et.commit();
               return dbAlarm;
            }
            else if(alarm.getOngoing() == false){
               //Richiesta di terminazione
               dbAlarm.setOngoing(false);
               et.commit();
               GCM.sendEndAlarm(dbAlarm);
               return dbAlarm;
            }
            else if(alarm.getCaregiver() != null){
               //Richiesta per presa in carico
               if(dbAlarm.getCaregiver() != null){
                  //Allarme già preso in carico
                  et.commit();
                  return dbAlarm;
               }
               dbAlarm.setCaregiver(alarm.getCaregiver());
```

```
                dbAlarm.setTimestampTaken(new java.sql.Timestamp(new
                    Date().getTime()));
                et.commit();
                GCM.sendTakenAlarm(dbAlarm);
                return dbAlarm;
            }
            else if(dbAlarm.getCaregiver() == null){
                //Segnalazione periodica di allarme in corso
                et.commit();
                GCM.sendNewAlarmNotification(dbAlarm);
                return dbAlarm;
            }
            else {
                //Segnalazione periodica di allarme preso in carica
                java.util.Date currentTimestamp = new Date();
                java.util.Date takenTimestamp = dbAlarm.getTimestampTaken();
                long diff = currentTimestamp.getTime() -
                    takenTimestamp.getTime();
                if(diff > SettingsUtil.getSettings().getInterval()){
                    //Timeout, l'allarme torna in corso
                    dbAlarm.setCaregiver(null);
                    dbAlarm.setTimestampTaken(null);
                }
                et.commit();
                return dbAlarm;
            }

        }
        else{
            //Allarme non presente nel database
            et.commit();
            throw new Exception();
        }
    }
    catch(Exception e){
        et.rollback();
        throw e;
    }
  }
  finally{
    em.close();
  }
}
```

5.2.2 Il backend amministrativo

Il backend amministrativo è rappresentato da un sito web a cui gli operatori sanitari possono accedere per controllare lo stato del sistema o gestirne i contenuti. In particolare vengono offerte le seguenti caratteristiche:

- Visione dello storico delle richieste di assistenza degli ospiti della RAF. La visualizzazione in forma tabulare può essere totale, degli allarmi attualmente in corso o degli allarmi terminati. Le tabelle includono i tempi di inizio, presa in carico e termine degli allarmi, con cui è possibile effettuare statistiche sui dati raccolti.
- Possibilità di forzare la terminazione di allarmi in corso.
- Visione dello storico delle richieste di aiuto degli operatori sanitari in forma tabulare.
- Registrazione di nuovi ospiti nel sistema.
- Registrazione di nuovi operatori sanitari nel sistema.
- Gestione degli utenti registrati al sistema, con la possibilità di modificarne gli attributi.
- Gestione delle impostazioni del sistema. In particolare è possibile personalizzare il periodo di tempo in cui un allarme preso in carico debba essere servito.

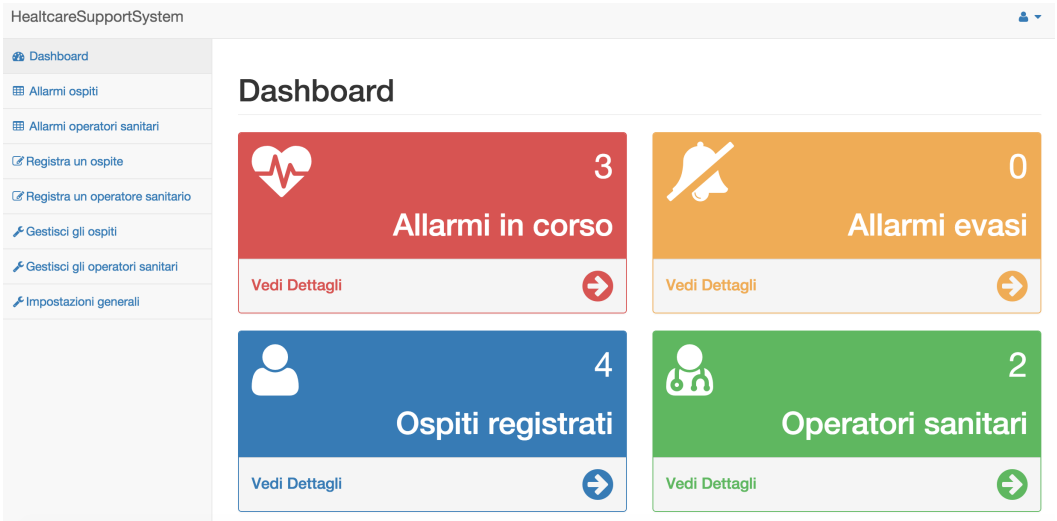


Figura 5.11: L'home page del sito di amministrazione del sistema

La visione degli storici è permessa a tutti gli operatori sanitari registrati sul sistema, mentre la parte di gestione degli utenti è possibile solo per quegli assistenti con privilegi di amministratore.
L'implementazione è molto semplice: seguendo il paradigma *"Model-View-Controller"* su opportune URL sono mappati dei controllori che intercettano le richieste degli utenti, operano sui dati tramite i metodi di servizio per recuperare i contenuti richiesti e selezionano la vista, sotto forma di pagina JSP, da mostrare all'utente. Per l'autenticazione è presente un filtro che ad ogni richiesta controlla se l'utente è autenticato, reindirizzandolo a una pagina di login in caso negativo.
Nelle figure sono riportate due schermate del sito web. In figura 5.11 si può vedere l'home page a cui si può accedere dopo aver eseguito l'autenticazione con le credenziali di un operatore sanitario. Nel menu a sinistra si notano tutte le operazioni che si possono eseguire, mentre nella parte centrale sono messe in evidenza le sezioni più importanti, che rimandano alla visualizzazione degli allarmi e alla gestione degli utenti nel sistema.

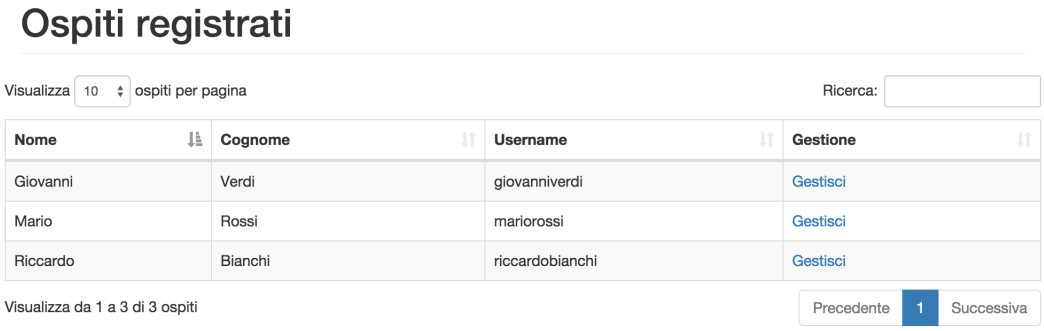


Figura 5.12: La lista degli ospiti registrati sul sistema

In figura 5.12 è presentata invece la parte del sito che permette di visualizzare gli ospiti registrati nel sistema. Cliccando sul bottone *"Gestisci"* di ogni riga della tabella si accederà ai dettagli di ogni ospite, con la possibilità di modifica e di eliminazione dello stesso dal sistema. Per concludere, in figura 5.13, è presentata la schermata che visualizza l'elenco delle richieste di assistenza in corso degli ospiti. Da questa tabella un operatore sanitario ha anche la possibilità di forzare la terminazione di un allarme, affinché in caso la comunicazione tra i dispositivi sia interrotta a causa di malfunzionamenti un operatore possa comunque agire sullo stato del sistema.

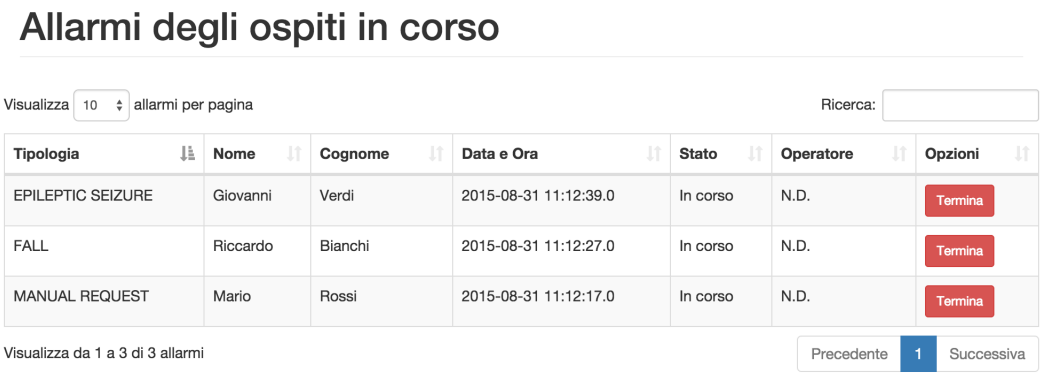


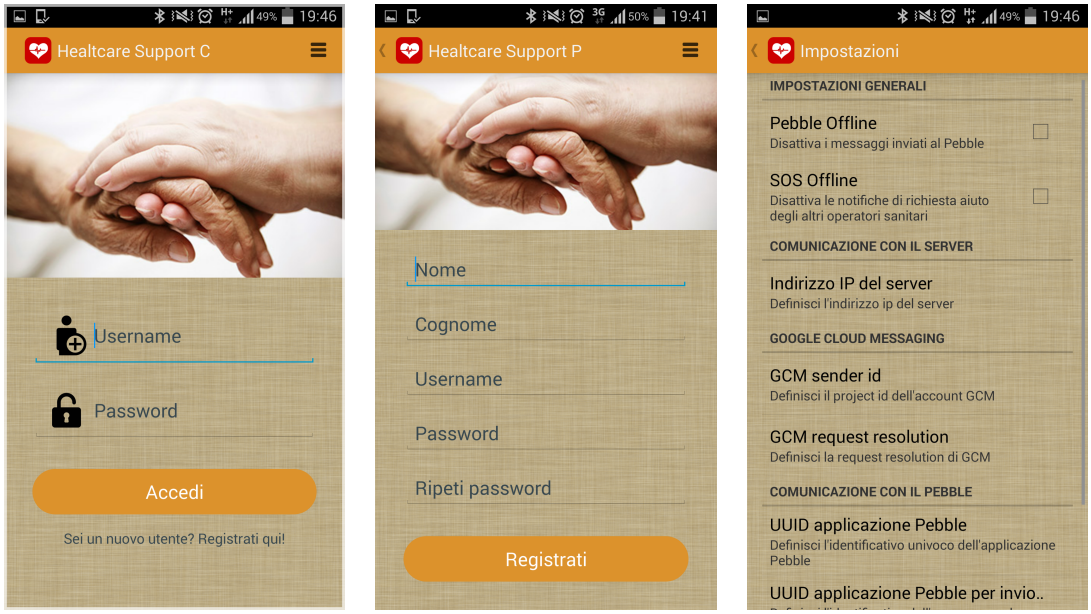
Figura 5.13: La visualizzazione delle richieste di assistenza nel sito web amministrativo

5.3 Le applicazioni Android

Le applicazioni Android sono costruite secondo una logica precisa: il funzionamento dev'essere quanto più possibile automatico, prevalentemente in background, e non deve essere interrotto anche se le applicazioni vengono chiuse. Per fare questo sia la versione per l'operatore sanitario che quella per l'ospite della RAF fanno largo uso di elementi come *broadcast receiver* e *service*. I primi componenti sono utilizzati per registrarsi come ascoltatori degli eventi a cui l'applicazione è interessata, come la ricezione di messaggi dallo smartwatch o di notifiche del servizio di Google Cloud Messaging. Catturato un evento viene istanziato un *service* per effettuare le opportune operazioni in background. Vengono utilizzati gli *intent service* affinché l'elaborazione venga svolta in un thread separato da quello principale; un componente di questo tipo tratta un messaggio per volta, e concluse le operazioni il thread secondario viene terminato. Da notare che anche queste applicazioni, così come il server, utilizzano la libreria *"HealthcareSupportLib"*: possono dunque operare con i vari oggetti modellati dal sistema (*"Caregiver"*, *"Patient"*, *"PatientAlarm"* e *"CaregiverAlarm"*).
Tutte e due le applicazioni hanno una parte in comune formata da tre *activity*, finestre con cui l'utente può interagire. Nell'ordine sono presenti una schermata per il login, una per la registrazione e una per le impostazioni. Di particolare interesse è quella per l'autenticazione, che va svolta al primo avvio dell'applicazione affinché tutto il sistema possa funzionare. Inseriti username e password l'applicazione esegue una richiesta HTTP verso il server inserendo nel corpo le credenziali. Il server in caso positivo risponde con un oggetto di tipo *"Caregiver"* o *"Patient"* rappresentante l'utente stesso: in questo modo il dispositivo può conoscere l'identificativo univoco dell'utilizzatore e salvarselo nelle proprie *Shared Preferences*[3] per successivi utilizzi. Il processo di autenticazione non termina qui, ma il client compie ancora alcuni passaggi. Per prima cosa tenta di registrarsi al servizio di Google Cloud Messaging ottenendo un codice per l'utilizzo, poi recupera il proprio inidirizzo Bluetooth. Queste due informazioni sono inserite nell'oggetto rappresentate l'utente ricevuto in precedenza, il quale viene reinviato al server. Tramite questa UPDATE il nodo centrale viene a conoscenza tutti i parametri che gli servono per interagire con l'utente. Nella schermata di registrazione invece l'applicazione raccoglie i campi inseriti in un oggetto *"Caregiver"* o *"Patient"* e lo invia al server affinché il nuovo utilizzatore sia inserito nel database. I campi relativi al Bluetooth e a GCM verranno riempiti come visto al primo login.
L'ultima finestra, quella delle impostazioni, permette la modifica di alcune caratteristiche come l'indirizzo del web server o alcuni parametri relativi a GCM.

[3]Un oggetto SharedPreferences punta a un file contenente coppie chave-valore e fornisce semplici metodi per la loro lettura e scrittura.

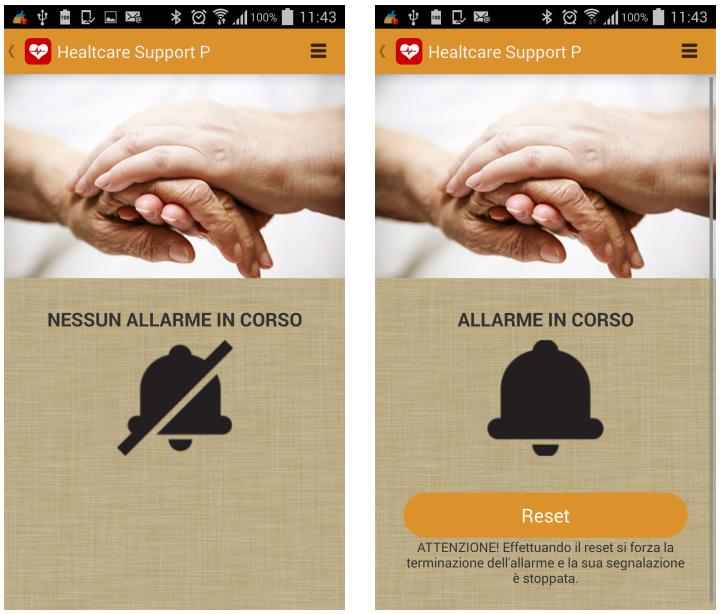
Le tre schermate, comuni a tutti e due i tipi delle applicazioni, sono riportate nelle figure seguenti. In figura 5.14 (a) si può vedere la pagina di autenticazione con cui è necessario interagire al primo avvio. Nella figura 5.14 (b) l'applicazione è ferma sulla schermata di registrazione: inseriti gli opportuni campi è possibile tentare di registrare un nuovo utente (il cui tipo, ospite o operatore, dipende dal tipo dell'applicazione) nel sistema. Per finire, nella figura 5.14 (c) viene mostrata la schermata delle impostazioni. Oltre ai campi relativi alla comunicazione con il server, per la versione dell'operatore sanitario c'è la possibilità di disabilitare le notifiche sia per lo smartwatch che per lo smartphone stesso (in caso non si vogliano ricevere richieste di aiuto di altri operatori).



(a) *La schermata di login.* (b) *La schermata di registrazione.* (c) *La schermata delle impostazioni.*

Figura 5.14: Le applicazioni Android

Esclusiva dell'applicazione per l'ospite della RAF è la schermata mostrata in figura 5.15. Viene offerta la possibilità di eseguire il reset del sistema in caso di problemi o di malfunzionamenti: con questa procedura il sistema tenta prima di forzare la terminazione di un eventuale allarme in corso, per poi disabilitare ogni timer di invio automatico, riportando l'applicazione allo stato iniziale di funzionamento.



(a) *La schermata che visualizza lo stato dell'applicazione in assenza di richieste.*

(b) *La schermata che visualizza lo stato dell'applicazione in presenza di richieste.*

Figura 5.15: La schermata di visualizzazione dello stato per l'ospite della RAF

5.3.1 L'applicazione Android per l'operatore sanitario

La struttura dell'applicazione Android per l'operatore sanitario è rappresentata in figura 5.16.

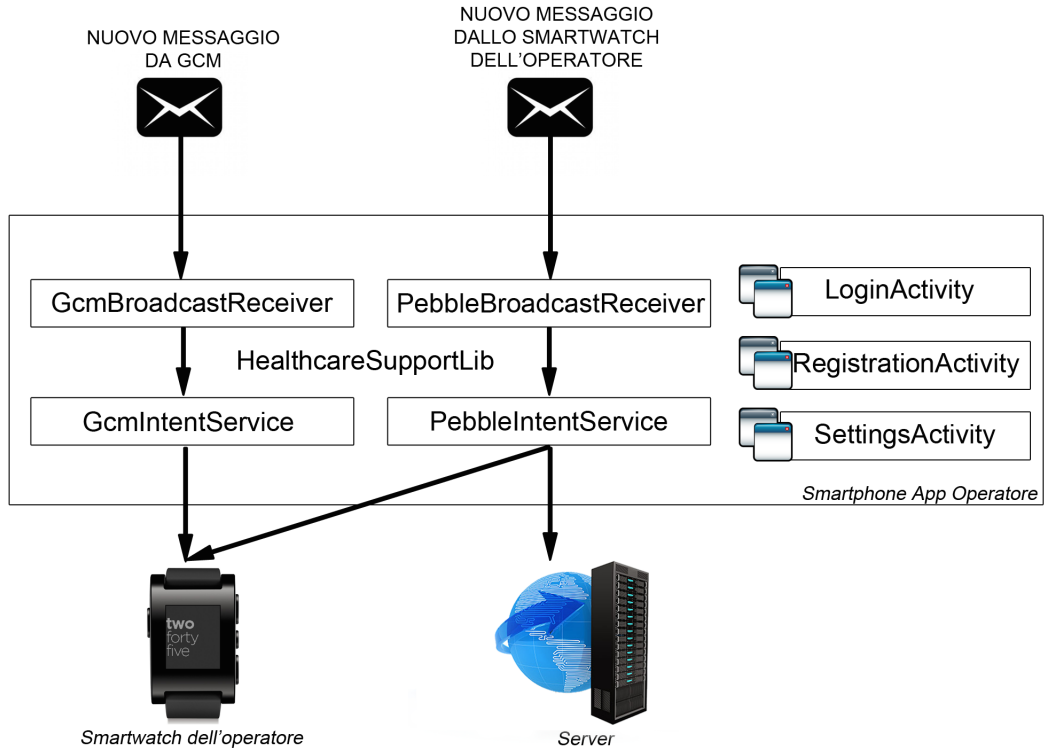


Figura 5.16: La struttura dell'applicazione Android per l'operatore sanitario

Oltre alle tre *activity* descritte in precedenza si vede che esistono altri due flussi principali di esecuzione, ognuno formato da una coppia di componenti, un *broadcast receiver* e un *intent service*, che utilizzano la libreria *"HealthcareSupportLib"* per la modellazione delle entità. Le due parti sono relative ai due eventi principali a cui l'applicazione è interessata, ovvero la ricezione di messaggi dal server (tramite notifiche asincrone implementate con GCM) e la ricezione di messaggi dallo smartwatch associato via Bluetooth. Questi eventi sono catturati grazie ai due *broadcast receiver*, che sono registrati all'avvio tramite il file di manifest[4] e funzionano indipendentemente dal ciclo di vita dell'applicazione, permettendo dunque il funzionamento anche quando quest'ultima viene chiusa. Ricevuto un messaggio i due componenti

[4]Il file di manifest di una applicazione android contiene le informazioni essenziali che il sistema Android deve conoscere prima che sia eseguito il codice, come la descrizione dei componenti, i permessi di esecuzione ecc.

lanciano un servizio deputato all'elaborazione in background. Nel caso di un nuova richiesta dallo smartwatch l'applicazione si comporta come segue:

- Se l'utente non è autenticato viene inviato un messaggio di errore allo smartwatch.

- Se il messaggio ricevuto è una richiesta di inizializzazione, ovvero la richiesta che lo smartwatch invia tutte le volte che la sua applicazione viene avviata, il servizio scarica dal server tramite due richieste HTTP la liste delle richieste di assistenza in corso e quella della richieste prese in carico dall'operatore e le invia allo smartwatch.

- Se il messaggio ricevuto è relativo a una presa in carico per un determinato allarme il servizio redirige la richiesta verso il server, che risponderà con l'esito dell'operazione da ritornare allo smartwatch e invierà eventualmente una notifica a tutti gli altri operatori sanitari in modo tale che essi possano aggiornare la loro visualizzazione degli allarmi.

- Se il messaggio ricevuto è relativo a una richiesta di aiuto da parte dell'operatore stesso il servizio invia al server tramite una richiesta HTTP un oggetto *"CaregiverAlarm"* collegato all'identità del richiedente. Il server si occuperò di inviare la notifica a tutti gli altri operatori sanitari.

Nel caso di un nuovo messaggio arrivato dal lato di Google Cloud Messaging l'applicazione si comporta come segue:

- Se si riceve un nuovo allarme di un ospite il servizio apre forzatamente l'applicazione dello smartwatch associato (grazie a una funzionalità offerta dalle librerie Pebble) e gli invia la richiesta di assistenza.

- Se il messaggio ricevuto è relativo a una presa in carico significa che un operatore nel sistema ha preso in carico una richiesta di assistenza. L'informazione è propagata allo smartwatch, che potrà visualizzare la notifica e aggiornare le proprie liste di allarmi.

- Se il messaggio ricevuto è relativo alla terminazione di una richiesta di assistenza l'informazione è propagata allo smartwatch, che potrà visualizzare la notifica e aggiornare le proprie liste di allarmi.

- Se il messaggio è relativo a una richiesta di aiuto di un altro operatore l'applicazione mostra una notifica di sistema direttamente sul dispositivo.

5.3.2 L'applicazione Android per l'ospite della RAF

Come si può notare dalla figura 5.17, che riproduce la struttura base dell'applicazione, ci sono anche qui le *activity* di login, registrazione e per le impostazioni, unite alla schermata che permette di effettuare il reset dell'applicazione. Anche in questo caso, per modellare le entità, l'applicazione utilizza la libreria *"HealthcareSupportLib"*.

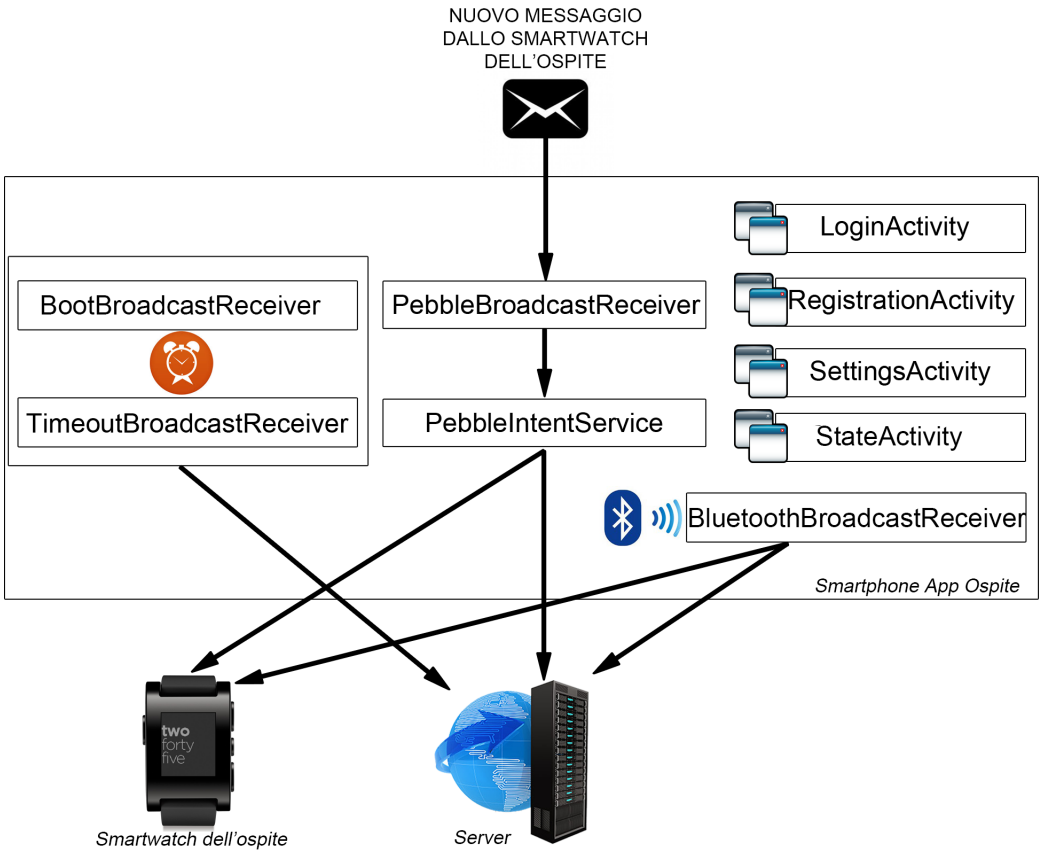


Figura 5.17: La struttura dell'applicazione Android per l'ospite della RAF

Oltre a questo si può notare che rispetto all'applicazione per l'operatore sanitario questa viene contattata dallo smartwatch associato, ma non ha a che fare con le notifiche asincrone del server implementate tramite GCM. Quando viene ricevuto un messaggio dall'orologio, catturato tramite l'apposito *broadcast receiver*, l'applicazione lancia un *intent service* che si comporta nel seguente modo:

- Se l'utente non è autenticato viene inviato un messaggio di errore allo smartwatch.
- Se il messaggio ricevuto è una richiesta di inizializzazione, ovvero la richiesta che lo smartwatch invia tutte le volte che la sua applicazione viene avviata, il

servizio richiede al server se l'utente ha attualmente una richiesta in corso e comunica il risultato allo smartwatch, in modo tale che esso possa visualizzare lo stato corrente dell'ospite.

- Se il messaggio ricevuto è relativo a una nuova richiesta di assistenza e se attualmente l'utente non ha richieste in corso il servizio crea un oggetto *"PatientAlarm"* aggiungendo alle informazioni ricevute l'identità del richiedente, lo salva come allarme pendente nelle *SharedPreferences* e invia una richiesta HTTP con metodo POST al server in modo tale che l'allarme sia aggiunto al database e la notifica sia propagata a tutti gli operatori sanitari di turno. Se tutto va a buon fine il servizio istanzia un timer che ogni 30 secondi sarà deputato a reinviare lo stesso allarme al server fintanto che questo non sia terminato.

- Se il messaggio ricevuto è relativo alla richiesta di terminazione di un allarme per prima cosa l'applicazione controlla se effettivamente l'utente ha una richiesta pendente salvata, in caso contrario le operazioni terminano. Se il servizio deve procedere la richiesta di terminazione va validata, perché solamente un operatore sanitario ha il privilegio di eseguire tale operazione. Questo viene fatto iniziando una scansione Bluetooth e creando un *broadcast receiver* (nella figura chiamato *"BluetoothBroadcastReceiver"*) con il compito di ricevere gli eventi legati a questa attività. Il sistema modella i dispositivi trovati durante la scansione sotto forma di opportuni oggetti contenenti tra le altre cose l'indirizzo Bluetooth: è compito del *broadcast receiver* salvare queste informazioni ricevute in una opportuna struttura dati. Al termine della scansione lo stesso componente scarica dal server la lista di indirizzi Bluetooth appartenenti agli operatori sanitari registrati sul sistema, cercando corrispondenza con quelli dei dispositivi trovati. Se la validazione ha esito positivo l'applicazione richiede al server di terminare la richiesta, altrimenti invia un messaggio di errore allo smartwatch.

Per concludere la spiegazione delle parti principali rimangono due *broadcast receiver*, *"TimeoutBroadcastReceiver"* e *"BootBroadcastReceiver"*.
Il primo si occupa di catturare l'evento scatenato ogni 30 secondi dall'eventuale timer impostato alla presenza di una nuova richiesta di assistenza. Il suo compito sarà quello di recuperare dalle *SharedPreferences* l'allarme pendente e reinviarlo al server. Come risposta il server risponde con lo stesso oggetto *"PatientAlarm"* eventualmente modificato (a seconda dello stato in cui si viene a trovare l'allarme nella base dati). Se l'allarme ricevuto in risposta contiene l'indicazione che è terminato il timer viene disabilitato e l'allarme pendente salvato nelle preferenze di sistema è cancellato.
L'ultimo *broadcast receiver* cattura l'evento che ha origine quando il dispositivo viene acceso. Il suo compito è quello di richiedere al server se l'utente ha attualmente un allarme in corso, affinché se presente venga riattivato il timer per il reinvio

automatico. Non si vuole infatti perdere per strada nessuna richiesta, anche nel caso che il dispositivo si riavvii inaspettatamente.

5.4 Le applicazioni per gli smartwatch Pebble

Le applicazioni per gli smartwatch Pebble agiscono al bordo del sistema e sono quelle più a contatto con l'utente. Le due versioni, per l'operatore sanitario e per l'ospite della RAF, si differenziano molto. La prima ha come caratteristica principale la visualizzazione di richieste di assistenza, mentre la seconda è implementata affinché vengano rilevate cadute e attacchi epilettici per chi utilizza lo smartwatch. Anche gli aspetti implementativi differiscono: in una applicazione il focus è sull'usabilità e sull'interfaccia grafica, nell'altra viene utilizzato intensamente l'accelerometro e il funzionamento è per lo più automatico.

5.4.1 L'applicazione Pebble per l'ospite della RAF

L'applicazione per l'ospite della RAF si può concettualmente suddividere in due parti principali. Alla base vi è un *Background Worker* che in background, ovvero indipendentemente dal ciclo di vita dell'applicazione principale, raccoglie i dati provenienti dall'accelerometro e ne esegue l'analisi alla ricerca di situazioni di pericolo per l'ospite. Vengono raccolti 10 campioni accelerativi alla volta con una frequenza di 100 Hz. Ognuno di essi è composto da tre componenti, rispettivamente relative agli assi x, y e z. Per prima cosa i dati vengono ripuliti dalla loro componente gravitazionale, anch'essa composta di tre dimensioni spaziali. Essa agisce alle basse frequenze: è inizializzata con il primo campione di accelerazione rilevato, isolata tramite un filtro passa basso e successivamente sottratta ai campioni originali. I campioni vengono poi uniti in un unica risultante attraverso la radice quadrata dei quadrati delle singole componenti. Le risultanti sono infine raggruppate in finestre temporali della durata di 5 secondi e salvate in una opportuna struttura dati che servirà per l'analisi della finestra temporale corrente. Per rendere l'idea sono riportati alcuni frammenti di codice.

Collezionamento dei dati

```
//isolamento gravità
g.x = ALFA*g.x + (1-ALFA)*data[i].x;
g.y = ALFA*g.y + (1-ALFA)*data[i].y;
g.z = ALFA*g.z + (1-ALFA)*data[i].z;

//calcolo accelerazione lineare
data[i].x = data[i].x - g.x;
data[i].y = data[i].y - g.y;
```

```
data[i].z = data[i].z - g.z;

//calcolo accelerazione risultante
float accel_res = fsqrt(data[i].x*data[i].x +
   data[i].y*data[i].y + data[i].z*data[i].z);

//salvataggio risultante
accel_data[accel_data_index++] = accel_res;
```

Quando è raggiunta la fine di una finestra temporale viene eseguita l'analisi dei dati raccolti per determinare la presenza di situazioni critiche. In particolare vengono calcolati la media di accelerazione, il picco principale, e il numero di picchi secondari che superano una certa soglia.

- Un attacco epilettico è verificato se la media accelerativa, il picco principale e il numero di picchi secondari superano una certa soglia.
- Una caduta è verificata se il picco principale supera una certa soglia e se la media accelerativa e il numero di picchi secondari cadono in un certo intervallo di valori.

L'idea è che un attacco epilettico presenti media accelerativa e picco principale elevati, e che il numero di picchi secondari sia numeroso, mentre una caduta dovrebbe presentare media e numeri di picchi secondari più bassi, ma picco principale più alto. Terminata l'analisi il procedimento di collezionamento riparte con una nuova finestra temporale. Se si verifica una potenziale richiesta di assistenza il *Background Worker* contatta la seconda parte dell'applicazione, quella principale, forzandone l'avvio. Questa parte offre una semplice interfaccia grafica e si preoccupa di gestire la comunicazione con il dispositivo mobile associato visto che gli elementi in background non hanno la possibilità di farlo. L'applicazione è formata da un'unica *Window*[5] che visualizza se è in corso una richiesta e registra le callback necessarie alla comunicazione. La struttura è la seguente:

- All'avvio l'applicazione contatta il dispositivo mobile associato per sapere se attualmente l'utente ha una richiesta di assistenza in corso. In questo modo sullo schermo è possibile visualizzare lo stato corrente.
- Quando l'applicazione viene contattata dal *Background Worker* oppure quando l'applicazione è aperta e l'utente preme il tasto centrale dello smartwatch viene inviato automaticamente al dispositivo mobile associato la notifica di una nuova richiesta di assistenza.

[5]Nelle API fornite da Pebble una Window è il blocco base della user-interface

- Quando un utente preme il pulsante superiore dello smartwatch l'applicazione invia una richiesta di tentativo di terminazione per l'allarme in corso.

La comunicazione è implementata interamente tramite le API di *"AppMessage"* che permettono una scambio bidirezionale affidabile di un numero arbitrario di coppie chiave-valore tra il dispositivo mobile e lo smartwatch. I singoli messaggi sono incapsulati in strutture dati chiamate *Tuple* e raggruppati in *Dictionary*. Visto che le API sono bidirezionali sia il dispositivo mobile che l'orologio possono iniziare per primi la comunicazione. Ogni messaggio deve essere confermato attraverso un acknowledge positivo (ACK) oppure non confermato (ad esempio per problemi di formato) attraverso un acknowledge negativo (NACK), in caso contrario ci sarà un errore di timeout. Le applicazioni utilizzano l'UUID (numero univoco associato all'applicazione per lo smartwatch) come identificativo dei messaggi scambiati. La figura 5.18, presa dalla documentazione ufficiale Pebble, illustra bene come il protocollo delle API *"AppMessage"* funziona: si vede bene che tutti e due i lati della comunicazione (lo smartwatch Pebble e il dispositivo mobile) devono confermare in modo positivo o negativo la ricezione di un messaggio.

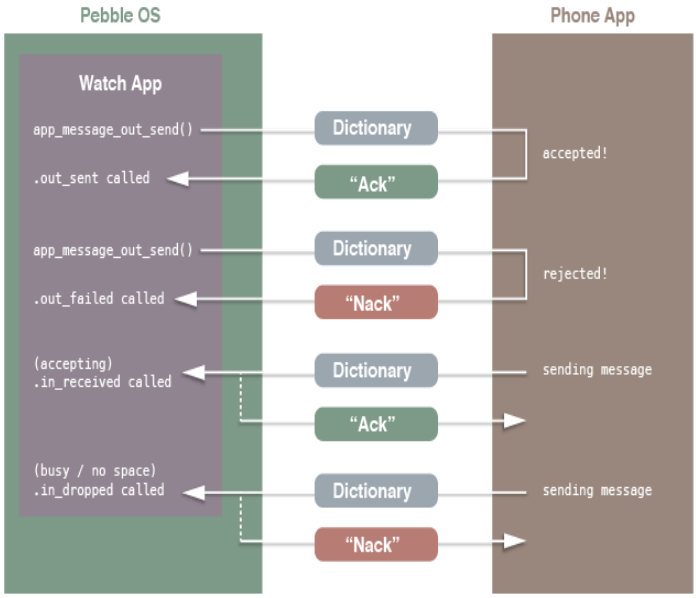


Figura 5.18: La comunicazione implementata con AppMessage

Il funzionamento dell'applicazione è per lo più automatico affinché anche chi non abbia la possibilità di interagire con l'orologio possa lo stesso usufruirne: le richieste di assistenza rilevate grazie all'accelerometro sono inviate senza il bisogno di alcuna interazione dell'utente.

In figura 5.19 è mostrata l'implementazione grafica dell'applicazione. Quando l'utente non ha richieste in corso viene mostrata la schermata di figura 5.19 (a). L'ospite ha la possibilità di richiedere aiuto premendo il pulsante centrale dell'orologio, come indicato dal menu ad immagini presente sul lato destro dello schermo, oppure una richiesta è automaticamente segnalata in caso di movimenti innaturali. Se questo avviene l'applicazione passa alla schermata di figura 5.19 (b). In questo caso un operatore sanitario che raggiunga l'ospite in difficoltà ha la possibilità di richiedere la terminazione dell'allarme premendo il pulsante superiore dell'orologio. In figura 5.19 (c), per finire, è mostrato come l'applicazione si comporta in caso riceva un messaggio di errore dallo smartwatch associato.

(a) *Applicazione in assenza di richieste in corso.*

(b) *Applicazione durante una richiesta in corso.*

(c) *Un messaggio di errore ricevuto dallo smartphone.*

Figura 5.19: L'applicazione Pebble per l'ospite della RAF

5.4.2 L'applicazione Pebble per l'operatore sanitario

L'applicazione per l'operatore sanitario è strutturalmente diversa da quella per l'ospite poiché assolve a compiti differenti. La caratteristica principale in questo caso è la presenza di un'interfaccia grafica semplice e intuitiva che permette all'utente di orientarsi bene tra le varie richieste di assistenza presenti nel sistema e che tiene sempre presente il vincolo di robustezza dato dal fatto che la segnalazione degli allarmi in corso deve essere esaustiva. L'applicazione si compone di tre finestre:

- La finestra principale contiene un menu che permette di accedere all'elenco delle richieste di assistenza in corso, a quello delle richieste prese in carico dall'utente, e di inviare una richiesta di aiuto a tutti gli altri operatori sanitari.

- La seconda finestra visualizza l'elenco delle richieste di assistenza in corso, comprendenti il tipo di allarme e il paziente richiedente. Essa permette all'utente di scorrere la lista e di selezionarne i singoli componenti per la presa in carico.

- La terza finestra visualizza l'elenco delle richieste di assistenza prese in carico dall'utente, comprendenti il tipo di allarme e il paziente richiedente, in modo tale che l'operatore non possa dimenticarsi dei suoi compiti.

Anche in questo caso l'applicazione gestisce la comunicazione con il dispositivo mobile associato tramite le API di *"AppMessage"*, registrando una callback per la ricezione di messaggi. In particolare vengono mantenute in memoria due liste contenenti gli allarmi in corso e gli allarmi presi in carico, il cui contenuto iniziale è richiesto all'avvio al dispositivo associato, che propagherà la richiesta al server.
Un allarme è modellato da un file separato che contiene una opportuna struttura dati e fornisce i metodi che servono per implementare il concetto di *Abstract Data Type*: si cerca di far avvicinare la programmazione in linguaggio C alla programmazione ad oggetti.

La struttura dati modellate una richiesta di assistenza

```
struct alarm_s{
  //Il tipo dell'allarme: caduta, attacco epilettico
  //o richiesta manuale
  int type;
  //L'id della richiesta, assegnato dal server
  int request_id;
  //L'id dell'ospite richiedente
  int patient_id;
  //Il nome dell'ospite richiedente
  char *patient_name;
};
```

Quando dallo smartwatch si riceve una nuova richiesta di assistenza essa viene aggiunta alla lista di allarmi in corso e automaticamente viene messa in evidenza la corrispondente finestra di visualizzazione. Se la lista di allarmi in corso era precedentemente vuota viene attivata una vibrazione continua dell'orologio, affinché l'attenzione dell'operatore sia catturata in modo efficace. Quando si ricevono invece notifiche di presa in carico o di terminazione il corrispondente allarme è spostato nella seconda lista o eliminato e se la lista delle richieste in corso rimane vuota la vibrazione è stoppata. In particolare, se una richiesta di assistenza è presa in carico dall'utente, essa è spostata nella lista delle prese in carico, mentre se è presa in carico da un'altro operatore essa è eliminata dalla memoria. In tutti e due i casi intanto, se la richiesta non sarà terminata entro un certo tempo, il server la farà

ritornare nello stato pendente ed essa arriverà nuovamente allo smartwatch sotto forma di nuovo allarme.
Si ricordi che se l'applicazione è chiusa, alla presenza di una situazione critica il suo avvio è forzato dal dispositivo mobile associato. Di seguito è possibile vedere lo pseudo-codice della callback di ricezione messaggi:

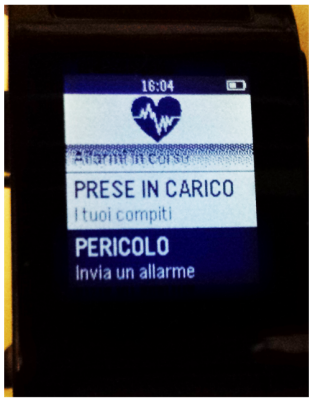
La comunicazione con lo smartphone

```
inbox_callback(){
 //[...]
 if(alarm){
  /*Ricevuta una richiesta di assistenza.
  Se non è già presente viene aggiunta
  alla lista*/
  if(!pending_request.contains(alarm))
   insert(alarm,pending_requests);
  /*Se è la prima richiesta viene
  fatta partire la vibrazione*/
  if(size(pending_requests) == 1)
   start_vibration();
  /*Viene visualizzata la schermata
  che mostra gli allarmi in corso*/
  showPendingScreen();
 }
 if(alarm_taken_id){
  /*Ricevuto l'id di un allarme preso in carica dall'utente.
  L'allarme è rimosso da quelli pendenti
  e inserito in quelli presi in carico dall'utente*/
  Alarm alarm_taken = remove(pending_requests,alarm_taken_id);
  insert(alarm_taken,taken_requests);
  /*Se non ci sono richieste pendenti
  la vibrazione è stoppata*/
  if(size(pending_requests) == 0)
   stop_vibration();
  refreshCurrentWindow();
 }
 if(alarm_end_id){
  /*Ricevuto l'id di un allarme terminato o preso
  in carica da un altro utente.
  L'allarme è rimosso dalla memoria*/
  remove(pending_requests,alarm_end_id);
  remove(taken_requests,alarm_end_id);
  /*Se non ci sono richieste pendenti
  la vibrazione è stoppata*/
  if(size(pending_requests) == 0)
```

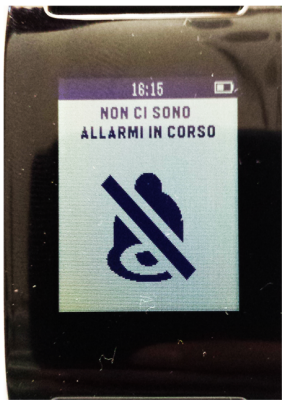
```
   stop_vibration();
  refreshCurrentWindow();
 }
}
```

In figura 5.20 si possono vedere le schermate principali dell'applicazione.

(a) *Il menu principale (1).*

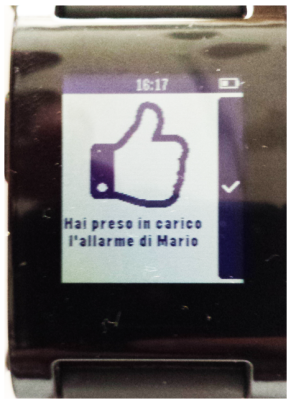


(b) *Il menu principale (2).*



(c) *Visualizzazione allarmi in corso (1).*

(d) *Visualizzazione allarmi in corso (2).*



(e) *Presa in carico di una richiesta.*

(f) *Termine di una richiesta.*

Figura 5.20: L'applicazione Pebble per l'operatore sanitario

Le prime due immagini mostrano il menu principale: all'utente è offerta la possibilità di vedere eventuali allarmi in corso o le sue richieste prese in carico, mentre con l'ultima voce può inviare lui stesso una richiesta di aiuto. Nelle figure 5.20 (c) e 5.20 (d) si può vedere la parte dell'applicazione per la visualizzazione degli allarmi

in corso, dove nella prima immagine il sistema non presenta segnalazioni, mentre nella seconda è notificata una richiesta all'operatore. Per concludere, nelle figure 5.20 (e) e 5.20 (f) viene mostrata la visualizzazione di due messaggi provenienti dal dispositivo mobile associato: uno è la conferma per la presa in carico di un allarme, l'altro è la notifica della terminazione di una richiesta.

La richiesta di aiuto per l'operatore sanitario

Come visto in precedenza l'applicazione per l'operatore sanitario possiede una schermata principale che permette l'invio di richieste di aiuto nel caso l'assistente stesso si trovi in difficoltà. Ovviamente questa funzionalità è disponibile solamente se l'applicazione è aperta. Anche se la chiusura dell'applicazione non è un problema per gli allarmi degli ospiti, in presenza dei quali l'avvio sarebbe forzato, affinché un operatore sanitario possa richiedere velocemente aiuto si è scelto di implementare un'altra applicazione da utilizzarsi unicamente per richiedere assistenza quando l'altro applicativo è chiuso. Essa è registrata sulla funzionalità Pebble di *Quick Launch* in modo tale che per il suo avvio basti tenere premuto un pulsante dell'orologio. L'implementazione è molto semplice ed è costituita da una funzione che tramite le API di *AppMessage* invia automaticamente all'avvio dell'applicazione un messaggio di richiesta aiuto allo smartphone o tablet associato, come si può notare dalla figura 5.21.

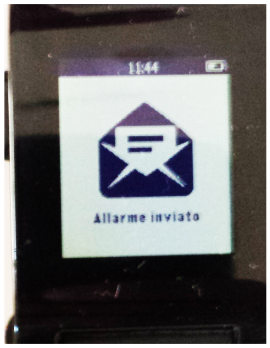


Figura 5.21: L'applicazione specializzata per inviare richieste di aiuto tra operatori sanitari

Capitolo 6

Test con utenti reali

Il sistema è stato testato in collaborazione con una comunità alloggio gestita dal comune di Torino. Con la sperimentazione si è voluto verificare l'usabilità del sistema, la sua robustezza e la sua efficacia, in modo tale da rilevare eventuali situazioni di inefficienza, errori e possibili miglioramenti. In questo capitolo vengono presentate le modalità con cui le prove sono state eseguite, ponendo particolare attenzione al protocollo adottato e agli aspetti più importanti su cui ci si è focalizzati.

6.1 La comunità alloggio "Officina delle idee"

La sperimentazione del sistema con utenti reali si è svolta nella comunità alloggio di tipo A "Officina delle idee" gestita dal comune di Torino. Una comunità alloggio è un servizio residenziale per persone con disabilità che si caratterizza come servizio funzionale alla soddisfazione dei bisogni della persona, al mantenimento ed al potenziamento delle capacità e dei livelli di autonomia acquisiti, al percorso di crescita personale e all'inserimento sociale, prestando particolare riguardo alla rete dei rapporti già esistenti e alla strutturazione di reti di sostegno al singolo e al gruppo. In particolare, le residenze di tipo A accolgono persone con disabilità fisiche e sensoriali con un grado di disabilità non incidente sulle facoltà intellettive e relazionali, con necessità di sostegno nella conduzione della vita quotidiana. Nell'"Officina delle idee" vengono ospitati al massimo sei disabili motori adulti, sia di sesso maschile che femminile. Gli ospiti soffrono spesso di disabilità gravi, si muovono per lo più grazie all'ausilio di carrozzine e alcuni di loro non hanno nemmeno la possibilità di utilizzare le mani per effettuare le operazioni più semplici come afferrare gli oggetti. La residenza è di piccole dimensioni e offre come unico spazio in comune la cucina con annesso un piccolo salotto. Da questo spazio è possibile accedere al bagno e al corridoio, su cui si affacciano le camere da letto. Durante il test si è constatato che la quotidianità degli operatori sanitari di questa residenza è simile a quella che si

sperimenta nelle RAF della cooperativa P.G. Frassati, mentre diversa è la tipologia di persone ospitate: nella comunità alloggio gli ospiti che soffrono di disabilità motorie molto gravi hanno come unico strumento per attirare l'attenzione la voce, mentre nelle RAF interessate dai focus group vengono ospitati prevalentemente dei disabili cognitivi, che hanno una vita molto più indipendente.

6.2 Il protocollo adottato per le prove del sistema

In questa sezione è riportata la descrizione della sperimentazione svolta, che ha seguito un particolare procedimento. Prima di recarsi nella residenza "Officina delle idee" infatti, si è definito uno specifico protocollo per l'effettuazione del test (visibile in Appendice A). Stabilendo a priori le modalità e gli aspetti più importanti su cui incentrare la sperimentazione si è avuta la sicurezza di non tralasciare alcun aspetto, e soprattutto sarà possibile replicare gli stessi test in futuro con altre realtà, in modo tale da poter avere un confronto attendibile tra i dati raccolti.

6.2.1 Gli aspetti principali della sperimentazione

La definizione di un protocollo da utilizzare per la sperimentazione è iniziata con il decidere gli aspetti più importanti su cui basare i test. In questo modo le operazioni da far svolgere ai partecipanti hanno seguito una linea ben precisa e sono state mirate proprio allo scopo di recuperare le informazioni predefinite, volte a stabilire un risultato complessivo sulla riuscita del progetto. Le informazioni qualitative da ricavare dalla sperimentazione sono state raccolte tramite un questionario consegnato agli operatori sanitari alla fine del test e basato sulle seguenti domande:

- Quali sono le prime impressioni che ti vengono in mente alla luce del test svolto?
- Qual è il tuo giudizio sull'usabilità del sistema? Il sistema è facile e intuitivo da utilizzare oppure hai incontrato qualche difficoltà?
- Come giudichi le interfacce grafiche con cui hai avuto a che fare (smartwatch, smartphone e sito web)?
- Come giudichi la modalità con cui gli ospiti possono richiedere volontariamente o involontariamente aiuto? Le richieste generabili dal sistema per gli ospiti coinvolti sono sufficienti secondo te?
- Come giudichi la modalità con cui puoi prendere in carico e terminare una richiesta? Le operazioni che devi svolgere in presenza di una richiesta di assistenza rispondono alle tue esigenze oppure hai qualche miglioramento da suggerire?

- Il sistema ti sembra utile? Secondo te può migliorare la tua condizione lavorativa e può garantire più sicurezza per gli ospiti?

- Alla luce del test, è un peso/preoccupazione il dover portarsi dietro un orologio (e il telefono)?

6.2.2 Descrizione del test

Durante il test si sono istruiti gli operatori sanitari all'utilizzo del sistema, cercando di valutarne soprattutto l'usabilità e l'intuitività di utilizzo. Alla presenza di un moderatore, che ha guidato i partecipati a compiere le varie operazioni previste, il test si è svolto velocemente e senza particolari intoppi. Durante i test un altro componente del gruppo di ricerca ha preso alcuni appunti, affinché potessero essere registrate anche le semplici reazioni dei partecipanti nell'utilizzo del sistema.

La preparazione al test

Le operazioni si sono svolte all'interno di un unica stanza, il salotto della residenza. La sperimentazione è stata effettuata con la collaborazione di due operatori sanitari della struttura, che si sono dimostrati subito molto interessati al funzionamento del sistema. Per poter mostrare tutte le caratteristiche del progetto, a uno degli operatori è stato fatto impersonare il ruolo di ospite della residenza. Sono stati infatti registrati preventivamente nel sistema un operatore e un ospite come utenti di prova. A ognuno dei partecipanti è stato affidato uno smartwatch Pebble e un smartphone Android associato su cui sono state installate le versioni corrette delle applicazioni (per ospite e per operatore). Come prima cosa è stato fatto compilare agli operatori sanitari un breve questionario iniziale per meglio classificare i profili dei partecipanti al test: sono stati richiesti i nominativi, la professione svolta e l'esperienza nell'utilizzo di dispositivi elettronici quali smartphone, smartwatch e computer.

I profili dei partecipanti

Analizzando i questionari iniziali è possibile ricavare il profilo dei partecipanti al test. Si tratta di due operatori socio-sanitari di sesso femminile con più di 55 anni, aventi tutti e due più di 10 anni di esperienza in questo campo.
I due operatori hanno esperienze diverse nell'utilizzo quotidiano della tecnologia: un operatore dice di avere una buona dimestichezza nell'uso di computer, smartphone e tablet, mentre l'altro confessa di averne poca. Tuttavia, entrambi dichiarano di non avere nessuna esperienza nell'utilizzo di dispositivi wearable come gli smartwatch che sono stati loro consegnati.

Lo svolgimento del test

La sperimentazione si è articolata in tre fasi. In un primo momento, sotto la guida di un moderatore, si è fatto vedere agli operatori come avviene la fase di inizializzazione del sistema, affinché esso possa essere utilizzato. Ai partecipanti si sono riferite le loro credenziali di accesso ed è stato detto di aprire le applicazioni sugli smartphone sviluppate per il sistema ("Health" per l'ospite e "Healthcare" per l'operatore), dicendo loro di autenticarsi. Il moderatore ha subito spiegato che il processo di autenticazione, a meno di utilizzare uno smartphone in comune con altri utenti, va svolto una sola volta e che dopo di questo non bisogna prestare particolari attenzioni al telefono, avendo però cura di portarselo sempre dietro.
In un secondo momento si sono simulati i casi d'uso comuni del sistema. Per prima cosa si è provato il caso di una richiesta di assistenza di un operatore. Il moderatore ha mostrato al partecipante che impersonava l'operatore le due modalità con cui è possibile richiedere aiuto ai colleghi, tramite l'applicazione "Healthcare" oppure tramite la pressione continuata del tasto superiore dell'orologio. Successivamente l'attenzione è stata posta sulla richiesta di assistenza manuale di un ospite. Si è per prima cosa guidato il partecipante che impersonava l'ospite ad effettuare una richiesta con l'applicazione "Health" per lo smartwartch. Dopo aver constatato l'inizio della segnalazione sullo smartwatch dell'operatore grazie all'applicazione "Healthcare" e aver fatto notare che anche se l'applicazione viene chiusa la segnalazione riprende, si è detto all'operatore di prendere in carico la richiesta e successivamente di terminarla. Il moderatore in questi passaggi ha volutamente fornito poche indicazioni, affinché si potesse verificare il grado di facilità e di intuitività dell'utilizzo del sistema. Questa parte della sperimentazione si è conclusa cercando di far vedere agli operatori la parte del monitoraggio automatico: al partecipante "ospite" si è detto di simulare un attacco epilettico mediante l'agitamento del braccio. L'altro partecipante ha così potuto ripetere il procedimento di presa in carico e terminazione, questa volta senza alcuna indicazione da parte del moderatore.
Per concludere sono stati esposti alcuni possibili casi di malfunzionamento, per far vedere agli operatori le funzionalità offerte in questi casi dal sistema, quali la possibilità di resettare l'applicazione per il dispositivo mobile dell'ospite e la terminazione di un allarme dal sito web amministrativo.

Capitolo 7

Risultati dei test e obiettivi raggiunti

Il capitolo conclusivo di questo lavoro è uno dei più importanti. All'interno viene definito il grado di successo che questo sistema ha ottenuto mediante la descrizione degli obiettivi raggiunti con la sperimentazione in laboratorio e con utenti reali. Aver potuto provare il sistema con gli stessi utenti a cui è rivolto aumenta l'attendibilità dei risultati ed è un aiuto per individuare gli sviluppi futuri e le correzioni che questo lavoro dovrà avere. Per meglio collegare i test con le aspettative che si avevano prima della progettazione e dell'implementazione è riportata, nella sezione 7.2, la stessa tabella del capitolo 2 (tabella 2.1), con un occhio di riguardo al grado di soddisfazione di ogni linea guida.

7.1 La raccolta dei risultati

Durante il test un componente del gruppo di ricerca ha appuntato le reazioni dei partecipanti. Questo, unito al fatto che spontaneamente gli stessi operatori hanno espresso alcuni loro pareri sul funzionamento del sistema e hanno proposto dei miglioramenti da adottare, soprattutto nell'ottica della loro residenza, ha permesso di tracciare un primo bilancio sulla sperimentazione. Le impressioni sono state confermate dall'analisi dei questionari finali fatti compilare agli operatori sanitari al termine della sperimentazione, con cui è stato possibile ricavare le informazioni qualitative prefissate.

7.1.1 Lo svolgimento del test e il confronto con gli operatori

Il dibattito informale con gli operatori creatosi durante e dopo il test è stato molto interessante e soprattutto molto utile. A un primo impatto con gli smartwatch Pebble gli operatori, che avevano confessato di non avere alcuna esperienza nell'uso

di tali dispositivi, si sono trovati un po' a disagio, mostrando alcune difficoltà nell'utilizzo comune degli orologi. Inoltre un operatore ha fatto notare che, soprattuto per gli utenti non più giovani, uno schermo di così piccole dimensioni potrebbe dar luogo a problemi di lettura.
Per quanto riguarda lo svolgimento della sperimentazione le modalità con cui effettuare una richiesta di assistenza manuale sono state individuate facilmente da un operatore su due, mentre la presa in carico e la terminazione non hanno presentato problemi: tramite indicazioni basilari del moderatore le operazioni sono andate subito a buon fine. La seconda prova di richiesta di assistenza, dove si è simulato un attacco epilettico, ha avuto ottimi risultati, poiché il partecipante che impersonava l'operatore ha ripetuto in autonomia la presa in carico e la terminazione, confermando che l'apprendimento del funzionamento del sistema è un processo sufficientemente veloce, anche per chi non ha mai avuto a che fare con uno smartwatch.
I commenti finali degli operatori sono stati positivi: essi hanno affermato che l'adozione di tale sistema sarebbe utile per l'assistenza ai disabili, soprattutto in strutture grandi. I partecipanti hanno anche suggerito alcuni miglioramenti secondo le loro esigenze. Visto che gli ospiti della loro residenza spesso non possono utilizzare le mani è stata suggerita l'introduzione della chiamata vocale, che unita ai metodi di richiesta assistenziale già implementati consentirebbe un'espansione e una generalizzazione notevole del sistema.

7.1.2 I dati raccolti dai questionari

In questa sezione è riportato un quadro complessivo dei dati raccolti dai questionari finali. Di seguito viene riportato il parere dei partecipanti al test, che potevano rispondere ad ogni domanda secondo una scala Linkert da 1 a 5, dove 1 indicava il risultato peggiore e 5 quello migliore. Il questionario comprendeva anche due domande di carattere più generale, affinché si potesse valutare se il sistema rispondesse o meno alle esigenze di ospiti e operatori.

Alla luce del test, come valuti l'utilità del sistema?

Dall'analisi delle risposte date a questa domanda si evince che entrambi gli operatori sanitari valutano l'utilità del sistema buona, avendo selezionato come punteggio 4.

Secondo te il sistema potrebbe migliorare la condizione lavorativa all'interno delle RAF?

Le risposte dei due partecipanti in questo caso sono diverse: un operatore, assegnando 5 punti, dice che che il sistema migliorerebbe molto la condizione lavorativa all'interno di una RAF, mentre l'altro valuta quest'aspetto con meno entusiasmo, dicendo che il sistema migliorerebbe solo abbastanza il suo lavoro quotidiano (punteggio 3).

Alla luce del test, è un peso o una preoccupazione il dover portarsi dietro un orologio e un telefono?

Entrambi gli operatori sanitari sono concordi nello specificare di non essere preoccupati da questo aspetto. Le risposte date infatti indicano che per un operatore non è per niente una preoccupazione (punteggio 5), mentre per l'altro è solamente una preoccupazione relativa (punteggio 3).

Come valuti le interfacce grafiche con cui hai avuto a che fare (applicazioni per gli smartwatch, applicazioni per gli smartphone e sito web)?

A questa prima domanda riguardate l'usabilità del sistema un operatore sanitario afferma che le interfacce grafiche con cui ha avuto a che fare durante il test sono molto intuitive, assegnando 5 punti con la sua risposta. L'altro operatore, con una valutazione di 3 punti, asserisce invece che la grafica del sistema è abbastanza intuitiva.

Come definiresti l'usabilità del sistema?

Anche la seconda domanda relativa all'usabilità del sistema ha avuto risposte diverse dagli operatori. Un partecipante valuta il sistema facile da utilizzare e assegna 4 punti alla risposta, mentre l'altro si dimostra un po' meno a suo agio con l'utilizzo, ma non totalmente insoddisfatto: valuta infatti il sistema abbastanza facile da utilizzare (punteggio 3).

Le modalità con cui puoi prendere in carico una richiesta di assistenza e terminarla rispondono alle tue esigenze?

Entrambi gli operatori sanitari asseriscono con le loro risposte a questa domanda che le modalità con cui sono avvertiti da una richiesta di assistenza, la possibilità della presa in carico e le operazioni da compiere per la terminazione rispondono alle loro esigenze.

Le modalità con cui si può richiedere assistenza rispondono alle esigenze degli ospiti?

Secondo i partecipanti al test le modalità con cui un ospite può richiedere assistenza non sono sufficienti. Gli operatori hanno voluto sottolineare che questa loro risposta è dovuta principalmente alla tipologia di ospiti (disabili motori con gravi limitazioni) della loro residenza.

7.2 Gli obiettivi raggiunti

Grazie ai risultati estratti dalla sperimentazione si vuole concludere il lavoro andando a riprendere le linee guida seguite e gli obiettivi prefissati. Adesso infatti è possibile capire quanti di essi sono stati soddisfatti e in quali ambiti sarà invece necessario introdurre miglioramenti. La tabella 7.1 ricalca quella del capitolo 2 (tabella 2.1) con la differenza di essere riscritta nell'ottica dei test svolti e dei risultati raccolti con la sperimentazione con utenti reali e in laboratorio.

Tabella 7.1: I risultati raggiunti

Tipo	Numero	Descrizione	Priorità	Raggiungimento
Ubiquità	LG 1.1	Sistema funzionante in movimento.	1	Parziale
	LG 1.2	Sistema funzionante nell'intera struttura.	1	Parziale
	LG 1.3	Dispositivi portabili e resistenti.	2	Totale
Usabilità	LG 2.1	Mani libere per gli operatori sanitari.	1	Totale
	LG 2.2	Richieste di aiuto non intrusive.	1	Totale
	LG 2.3	Assenza di dispositivi inutili e d'intralcio.	2	Parziale
	LG 2.4	Facilità e intuitività di utilizzo.	2	Totale
	LG 2.5	Protocollo di richiesta assistenza rispondente alle esigenze degli operatori.	2	Totale
	LG 2.6	Assenza di dispositivi difficilmente utilizzabili dagli ospiti.	2	Parziale
	LG 2.7	Mantenimento basso dei costi.	2	Totale
Robustezza	LG 3.1	Robustezza nell'individuazione di richieste di aiuto.	1	Totale
	LG 3.2	Robustezza nella segnalazione di allarmi.	2	Totale
	LG 3.3	Affidabilità della comunicazione tra i dispositivi.	2	Totale

Supporto per gli operatori	LG 4.1	Avvertimento automatico degli operatori.	1	Totale
	LG 4.2	Rilevazione automatica di richieste di assistenza.	1	Parziale
	LG 4.3	Possibilità di richiesta aiuto tra operatori.	3	Totale

I risultati sull'ubiquità del sistema

Il primo obiettivo (LG 1.1), che presupponeva la possibilità di utilizzo del sistema a partire da qualunque condizione degli utenti, è stato parzialmente raggiunto grazie all'adozione dei dispositivi wearable. In questo modo sia l'operatore sanitario che l'ospite possono utilizzare il sistema anche in movimento. Inoltre richiedere assistenza non implica più il dover raggiungere un punto preciso della struttura, ma può avvenire subito, proprio mentre un ospite o un operatore si trova in difficoltà. Un problema nasce dalla necessità di dover utilizzare sempre dispositivi mobili di supporto agli smartwatch. Se per l'operatore, come confermato dalle loro risposte al termine della sperimentazione, non è un problema, diverso è per gli ospiti con gravi problemi psico-fisici. Ad essi è infatti sconsigliato dare in dotazione questi dispositivi, in quanto possono essere facilmente rotti o non essere utilizzati correttamente: nascondere gli smartphone o i tablet in posizione fissa limiterebbe l'utilizzo in mobilità del sistema.
Come per l'obiettivo precedente anche la linea guida 1.2, riguardante la copertura del sistema a tutta la residenza, è stata solamente parzialmente raggiunta a causa dell'impossibilità di alcuni ospiti della RAF a tenere con sé smartphone o tablet di supporto. Con il dispositivo mobile di supporto in posizione fissa lo smartwatch associato rimane collegato nel raggio di copertura della tecnologia Bluetooth, che è circa di 100 metri.
La linea guida 1.3 è stata invece totalmente raggiunta poiché gli smartwatch Pebble utilizzati sono resistenti agli urti, essendo per lo più in plastica, e resistono anche all'acqua. Con gli smartphone il problema è meno rilevante perché essi possono essere tenuti in tasca, senza una necessità particolare di interazione.

I risultati sull'usabilità del sistema

Dal punto di vista dell'usabilità del sistema, in particolare per quanto riguarda la richiesta degli operatori sanitari di avere sempre le mani libere (LG 2.1), si può dire

che l'obiettivo sia stato completamente raggiunto: l'operatore deve solamente indossare un orologio, mentre lo smartphone di supporto può benissimo essere tenuto in tasca.
Anche la linea guida 2.2, riguardante la non intrusività delle richieste di assistenza, è stata pienamente rispettata. Gli operatori sanitari hanno dato la loro approvazione all'idea che le richieste arrivino direttamente sui loro smartwatch, a differenza di prima dove un ospite che richiedeva aiuto con il sistema esistente o con la voce rischiava di disturbare la quiete della residenza.
La linea guida 2.3 e la linea guida 2.6, che ponevano come obiettivo il non introdurre dispositivi inutili, facilmente dimenticabili o non utilizzabili da ospiti con gravi problemi, sono state raggiunte solo parzialmente a causa della necessità già emersa di avere i dispositivi mobili a supporto degli smartwatch Pebble.
Alla luce dei risultati raccolti grazie alla sperimentazione e grazie alle risposte al questionario finale è emerso che gli operatori sanitari trovano le interfacce grafiche con cui si sono scontrati sufficientemente intuitive e il sistema in generale rispondente alle esigenze richieste nei focus groups. In particolare essi hanno notato come non possano ignorare facilmente una richiesta di assistenza in quanto per terminarla occorre interagire con l'ospite richiedente aiuto. Per tutti questi motivi si può affermare che anche le linee guida 2.4 e 2.5 siano state rispettate.
Concludendo possiamo anche asserire che il costo di sviluppo e di mantenimento del sistema risulta abbastanza basso (LG 2.7). Gli smartwatch Pebble sono tra i più economici sul mercato, soprattuto se acquistati nella loro versione base, e la possibilità di utilizzare i propri smartphone contribuisce a non far incrementare le spese. Inoltre, se non ci si volesse sobbarcare l'onere di avere un server pubblico, sotto una buona copertura Wi-Fi il sistema è utilizzabile anche all'interno di una stessa rete locale.

I risultati sulla robustezza del sistema

Durante i test, sia in laboratorio che nella comunità alloggio, il sistema si è rilevato sufficientemente stabile e robusto, non presentando problemi di funzionamento. Per quanto riguarda la rilevazione automatica delle richieste (LG 3.1) il sistema si è comportato proprio come previsto durante la simulazione di un attacco epilettico.
Le altre due linee guida (LG 3.2 e 3.3) riguardanti la robustezza della segnalazione e l'affidabilità della comunicazione tra i dispositivi sono state pienamente rispettate. Un allarme infatti è continuamente segnalato agli operatori fin quando si trova nello stato *"in corso"*, e la comunicazione tra smartphone e server avviene tramite il protocollo HTTP: a meno di un malfunzionamento della rete sottostante l'affidabilità della comunicazione è assicurata.

I risultati sull'aiuto agli operatori sanitari

L'avvertimento automatico degli operatori sanitari (LG 4.1) è sicuramente un obiettivo totalmente raggiunto dal sistema, perché strettamente legato alla rilevazione automatica di richieste di assistenza (LG 4.2). Un possibile appunto va fatto in previsione di una espansione futura del sistema: per il momento infatti vengono monitorati quei problemi che sono più comuni per gli ospiti delle RAF ospitanti prevalentemente disabili cognitivi e comunque con disabilità motorie lievi, ovvero cadute e attacchi epilettici. Come già detto in precedenza gli operatori sanitari della residenza "Officina delle idee" hanno riportato che per la tipologia degli ospiti con cui hanno a che fare (disabili motori gravi) occorrerebbe inserire almeno una modalità di richiesta di assistenza vocale. Affinché le linee guida 4.1 e 4.2 siano rispettate totalmente anche per altre realtà occorre dunque prevedere l'adozione di dispositivi che permettano di analizzare altri segnali del corpo oltre al movimento. Per concludere, la linea guida 4.3, che prevedeva l'introduzione di una modalità di richiesta di assistenza anche per gli operatori sanitari, è stata rispettata pienamente.

Capitolo 8

Conclusioni

La progettazione, lo sviluppo e la sperimentazione di un sistema di supporto per gli operatori sanitari che lavorano in Residenze Assistenziali Flessibli si è rivelata una sfida motivante che ha regalato molte soddisfazioni. Le motivazioni sono nate soprattutto dalla possibilità di avere un riscontro pratico sull'utilità del lavoro svolto: aiutare lo svolgersi del lavoro quotidiano degli assistenti che lavorarono nelle RAF migliorerà di conseguenza anche la qualità della vita delle persone meno fortunate che sono ospitate, fattore che per l'autore ha avuto notevole importanza e ha dato una spinta in più all'impegno profuso. Il fatto poi di aver potuto provare il sistema, una volta completato, con la collaborazione della comunità alloggio "Officina delle idee" ha portato a benefici ulteriori: testare il sistema in laboratorio è utile fino a un certo punto, ma le funzionalità e soprattutto gli errori e gli aspetti da migliorare possono venir fuori solamente con l'utilizzo "sul campo". Da questo punto di vista dunque questa tesi si può dire privilegiata poiché ha potuto articolarsi su tutti gli stadi che l'evolversi di un sistema presuppone.
Il lavoro è incominciato con l'estrapolazione dei requisiti e delle linee guida dallo studio svolto in precedenza attraverso i focus group con la cooperativa P.G. Frassati, fase che si è rivelata di notevole importanza in quanto ha permesso l'evolversi lineare e senza particolari intoppi dei passi successivi.
Dai dati estratti si sono stabiliti gli obiettivi da raggiungere, affinché al termine dei lavori il confronto con le aspettative prefissate fosse semplice da eseguire.
La parte di progettazione ha implicato il focalizzarsi sugli aspetti più importanti ricavati nelle fasi precedenti e ha definito con precisione le caratteristiche principali che il sistema avrebbe adottato e i componenti utilizzati.
La fase di implementazione è venuta di conseguenza: la difficoltà principale è stata integrare le diverse tecnologie adottate, affinché dispositivi di natura diversa potessero comunicare tra di loro.
Per ultima cosa, ma non meno importante, c'è stata la possibilità di un test con utenti reali, che come visto nei capitoli precedenti ha fatto rilevare riuscita del progetto con il raggiungimento totale di molti degli obiettivi prefissati. In particolare

è stata riscontrata l'utilità di quanto fatto: la qualità del lavoro in una Residenza Assistenziale Flessibile viene sicuramente migliorata, e questo era forse l'obiettivo principale che ha fatto nascere tutti gli altri. Adesso infatti un operatore sanitario non ha più la necessità di rivolgere in modo estenuante la sua attenzione al controllo continuo degli ospiti, ma è avvertito in modo automatico in caso si verifichino problemi. Questa caratteristica principale del sistema conferisce all'assistente più tranquillità e lo porta ad avere un lavoro meno stressante.

8.1 Sviluppi futuri

8.1.1 Lo sviluppo tecnologico

Generalizzazione della soluzione

Il sistema implementato si presta molto bene ad essere ampliato e migliorato. Innanzi tutto le sue capacità dipendono in parte dai dispositivi coinvolti, soprattutto quelli wearable, e l'evoluzione della tecnologia non potrà che essere un vantaggio. Attualmente infatti il sistema rileva automaticamente situazioni di pericolo per gli ospiti della RAF solamente attraverso l'accelerometro a bordo degli smartwatch Pebble. Questo implica che l'unico fattore su cui ci si basa è il movimento: sebbene per gli individui coinvolti attualmente dal progetto questo è l'elemento principale a cui le crititcità più comuni (cadute e attacchi epilettici) sono relative, avendo a disposizione più sensori adatti al monitoraggio delle persone (viene da pensare ad esempio a un sensore per il rilevamento dei battiti cardiaci o all'utilizzo di un microfono per le chiamate vocali) il sistema potrebbe rilevare più tipologie di rischio e potrebbe essere adattato a più tipi di utenti, come i disabili motori gravi le cui esigenze sono venute fuori durante la sperimentazione nella comunità alloggio.

Impatto energetico

Con lo sviluppo della tecnologia, dei dispositivi e dei sistemi operativi sarà possibile apportare miglioramenti anche al sistema, soprattuto per quanto riguarda il consumo energetico, fattore fondamentale se si pensa che la maggior parte dei dispositivi coinvolti è di tipo mobile. Un esempio può essere la modalità con cui, quando viene richiesta la terminazione di una richiesta di assistenza da un dispositivo appartenente a un ospite della RAF, quest'ultimo ricerchi tramite una scansione Bluetooth la presenza di dispositivi appartenenti ad operatori sanitari. Attualmente per motivi di compatibilità con smartphone o tablet più vecchi non viene utilizzata la nuova tecnologia del Bluetooth Low Energy: la sua adozione renderebbe l'impatto del procedimento sulle batterie dei dispositivi meno importante.

Portabilità

Altro fattore che incide sulle prestazioni del sistema è la necessità di avere sempre i dispositivi mobili di supporto per gli smartwatch Pebble: la loro presenza limita allo stato attuale l'utilizzo del sistema in movimento per alcune tipologie di ospiti. Attraverso dispositivi wearable con possibilità di connessione diretta alla rete Internet questo problema verrebbe meno, e la portabilità del sistema aumenterebbe in modo sostanziale.

8.1.2 La sicurezza informatica

Come visto parte dell'affidabiltà e della robustezza del sistema è data dall'utilizzare per la comunicazione il protocollo HTTP e la rete Internet: in questo modo, se al server si assegna un indirizzo IP pubblico i dispositivi comunicare tra di loro anche attraverso la rete cellulare, e non occorre prevedere una copertura Wi-Fi totale. In sostanza non si ha la necessità di creare reti locali che spesso potrebbero dare problemi di connettività, essendo il sistema per sua natura distribuito e utilizzabile in movimento. Se da un lato questo è un innegabile vantaggio, dall'altro presuppone l'interessarsi a un problema, quello della sicurezza, che questa tesi non ha trattato in profondità e che ha lasciato come eventuale sviluppo futuro per il sistema. Con il termine sicurezza informatica si intende quel ramo dell'informatica che si occupa dell'analisi delle vulnerabilità, del rischio, delle minacce o attacchi e quindi della protezione dell'integrità fisica (hardware) e logico-funzionale (software) di un sistema informatico e dei dati in esso contenuti o scambiati in una comunicazione con un utente. Tale protezione è ottenuta attraverso misure di carattere tecnico-organizzativo e funzionali tese ad assicurare principalmente:

- La correttezza dei dati (integrità).
- La confidenzialità dei dati (cifratura).
- L'accesso fisico e/o logico solo ad utenti autorizzati (autenticazione).
- La fruizione di tutti e soli i servizi previsti per quell'utente nei tempi e nelle modalità previste dal sistema (disponibilità).

Sebbene per usufruire del sistema, sia nella sua parte principale che nel sito web amministrativo, un utente debba autenticarsi, nessuna attenzione è stata posta sulla confidenzialità e integrità dei dati. Anche se non vengono trattati dati particolarmente sensibili essi al momento viaggiano in chiaro tra i componenti del sistema. Questo problema si può limitare attraverso il protocollo HTTPS, che consiste in una comunicazione tramite protocollo HTTP all'interno di una connessione sicura tra i dispositivi, cifrata asimmetricamente dal protocollo TLS. L'uso di questa funzionalità presuppone però, almeno per il server, il possesso di un certificato digitale che

attesti l'associazione univoca tra una chiave pubblica e l'identità di un soggetto che dichiara di utilizzarla nell'ambito delle procedure di cifratura asimmetrica e/o autenticazione tramite firma digitale. Tale certificato deve essere fornito da un'entità di terza parte fidata, chiamata autorità di certificazione, ma al momento non è a disposizione del progetto.

Appendice A

Protocollo per la sperimentazione del sistema

A.1 Obiettivi del test

Verificare l'usabilità del sistema, la sua robustezza e la sua efficacia. Rilevare situazioni di inefficienza, errori e i miglioramenti possibili.

A.2 Domande di ricerca

A.2.1 Domande da fare agli operatori sanitari al termine del test

- Quali sono le prime impressioni che ti vengono in mente alla luce del test svolto?
- Qual è il tuo giudizio sulla usabilità del sistema? Il sistema è facile e intuitivo da utilizzare oppure hai incontrato qualche difficoltà?
- Come giudichi le interfacce grafiche con cui hai avuto a che fare (smartwatch, smartphone, e sito web)?
- Come giudichi la modalità con cui gli ospiti possono richiedere volontariamente o involontariamente aiuto? Le richieste di assistenza generabili dal sistema sono sufficienti secondo te?
- Come giudichi la modalità con cui puoi prendere in carico e terminare una richiesta? Le operazioni che devi svolgere in caso di una richiesta di assistenza rispondono alle tue esigenze oppure hai qualche miglioramento da suggerire?

- Il sistema ti sembra utile? Secondo te può migliorare la tua condizione lavorativa e può garantire più sicurezza per gli ospiti?
- Nell'ottica del test svolto, è stato un peso/preoccupazione il dover portarsi dietro un orologio (e il telefono)?

A.3 Caratteristiche del test

A.3.1 Tipologia del test e caratteristiche dei partecipanti

Nel test, della durata di circa trenta minuti, si istruiscono gli operatori sanitari all'utilizzo del sistema. Vengono scelti due operatori sanitari, di cui uno avrà il compito di impersonare un ospite della residenza. A ognuno di essi si affida uno smartwatch Pebble e un dispositivo Android associato.

A.3.2 Descrizione del luogo del test

Il test si può svolgere all'interno di un unica stanza, meglio se con la disponibilità di una connessione Wi-Fi e di un computer.

A.4 Descrizione del test

Il test si svolge in un unica stanza e coinvolge due operatori sanitari e un moderatore, che indicherà i compiti da eseguire. Un operatore sanitario simulerà un ospite della residenza, mentre un altro simulerà un operatore sanitario di turno. A ognuno viene affidato uno smartwatch Pebble e uno dispositivo Android (eventualmente il proprio smartphone) su cui vengono installate le versioni corrette delle applicazioni (per ospite e per operatore). All'inizio del test ogni partecipante riceverà un breve questionario a risposta chiusa dove gli verrà chiesto:

- Sesso.
- Età.
- Professione.
- Esperienza da operatore sanitario.
- Esperienza nell'utilizzo di un computer.
- Esperienza nell'utilizzo di uno smartphone.
- Esperienza nell'utilizzo di dispositivi wearable.

Il test proseguirà con il moderatore che farà eseguire dei task ai partecipanti. All'inizio di ognuno il moderatore spiega in breve le operazioni da compiere e le caratteristiche del sistema coinvolte. I task sono divisi in tre gruppi principali: setup (per inizializzare i dispositivi in modo da poter utilizzare il sistema), operatività (per testare il sistema nei casi d'uso tradizionali) e eccezioni (per testare il caso in cui si verifichino problemi nell'utilizzo del sistema). Le fasi di setup e di eccezioni vengono fatta svolgere tramite le spiegazioni del moderatore, mentre i task di operatività vengono fatti svolgere più in autonomia ai partecipanti, intervenendo in caso di difficoltà.

Tabella A.1: Elenco dei task da eseguire per il training degli operatori sanitari

Task	**Categoria**	**titolo**	**Descrizione**
T1	**Setup**	Autenticazione al sistema	Processo di autenticazione al sistema, da svolgersi una volta soltanto.
T2	**Operatività**	Simulazione di una richiesta di aiuto tra operatori sanitari	Invio di richieste di aiuto tra operatori sanitari.
T3		Simulazione di una richiesta di assistenza manuale	Simulazione di una richiesta manuale, constatazione della notifica continua all'operatore anche in caso di chiusura delle applicazioni per smartwatch.
T4		Simulazione della presa in carico	Presa in carico dell'allarme generato.
T5		Simulazione della terminazione	Terminazione dell'allarme preso in carica con ricerca di prossimità.
T6		Simulazione di un attacco epilettico	Simulazione di una richiesta di assistenza automatica scatenata con l'agitarsi del braccio.
T7		Simulazione di una presa in carico fallita	Simulazione di una presa in carico che non porta alla terminazione dell'allarme, con conseguente ritorno alla segnalazione della stessa.

T8	**Eccezioni**	Simulazione di una terminazione non riuscita	Simulazione di una terminazione non riuscita a causa di un guasto con lo spegnimento del Bluetooth dei dispositivi.
T9		Simulazione di una terminazione forzata	Forzatura della terminazione di un allarme tramite reset dell'applicazione dell'ospite o terminazione da sito web amministrativo.

A.5 Svolgimento del test

Grazie per aver accettato di prendere parte a questo test. Lo scopo è di provare le funzionalità di un sistema di supporto per il lavoro nelle RAF, verificarne le potenzialità e capirne i problemi. Vi spiego come si svolgerà il test. Nella prima fase proveremo il sistema solo tra di noi in questa stanza, con uno di voi che simulerà un ospite della residenza. Ognuno avrà con se due dispositivi, uno smartphone (o tablet) e un orologio Pebble, che come vedete è fatto in plastica ed è resistente agli urti e all'acqua. Verrà richiesto ad ognuno di compiere delle azioni che simulino il comportamento del sistema. Io spiegherò che cosa si deve fare per ogni situazione, una per volta, andando a evidenziare gli aspetti più importanti che potranno fornirvi le giuste indicazione per utilizzare il sistema. Fate attenzione che in alcuni casi dovrete provare a cavarvela da soli, senza le mie spiegazioni. In questo modo potremo valutare la facilità (o difficoltà) di utilizzo del sistema. Le spiegazioni che vi darò sono scritte in modo che io possa dare le stesse informazioni a tutti i partecipanti anche in test futuri. Se non avete domande, possiamo iniziare con il compilare un questionario iniziale.
[Compilazione questionario iniziale]
Ecco, possiamo incominciare il test con la prima situazione.

- *"Task 1"*: Vi chiedo allora di aprire sul vostro orologio l'applicazione del sistema, "Health" per gli ospiti e "Healthcare" per gli operatori. Come vedete viene visualizzato un errore. Affinché possiate utilizzare il sistema infatti dovete autenticarvi tramite il vostro smartphone (solo una volta, poi quest'ultimo si ricorderà di voi). Inserite dunque le credenziali che vi ho fornito nella schermata principale e premete il pulsante "Accedi".

- *"Task 2"*: Ottimo, per il momento adesso potete scordarvi del vostro smartphone, potete anche chiudere l'applicazione, l'importante è avercelo sempre dietro. Dopo la fase di setup incomincia la vera prova del sistema. Poniamo il caso che tu, operatore sanitario, sia l'unico di turno. Sei in pericolo oppure hai bisogno di aiuto e vuoi avvertire i tuoi colleghi. Prova a inviare una richiesta di aiuto al tuo collega a casa. Facciamo un'altra prova. Supponiamo che tu, operatore sanitario, sia di nuovo in difficoltà, ma non hai più guardato l'orologio e non sai se l'applicazione è aperta. Tieni premuto il tasto superiore, invierai immediatamente una richiesta.

- *"Task 3"*: Un ospite può scatenare una richiesta di assistenza in tre modi. Il primo modo è quello manuale, che ora andremo a vedere. Chiedo gentilmente all'ospite di aprire l'applicazione "Health" dell'orologio e chiedere aiuto. Come vedi la schermata è cambiata: il campanello è attivo, significa che stai richiedendo aiuto. Mi rivolgo all'operatore sanitario adesso: come vedi il tuo orologio ha iniziato a vibrare, e ti viene mostrata una richiesta di assistenza dell'ospite tramite l'applicazione "Healthcare". Tutto è successo in automatico, senza una tua interazione. Per confermare questo ti chiedo di chiudere l'applicazione e di attendere alcuni secondi. Ecco, anche se hai chiuso l'applicazione, l'allarme in corso continua a essere segnalato automaticamente.

- *"Task 4"*: Quando si riceve una richiesta di assistenza un operatore può prenderla in carico: in questo modo solamente uno si occupa di servire una richiesta, e tutti sanno chi ha quel compito. Chiedo all'operatore sanitario di prendere in carico la richiesta. Come hai visto sono successe diverse cose. Innanzi tutto il tuo dispostivo ha smesso di vibrare: ti stai infatti preoccupando di servire la richiesta. Se ci fossero altri operatori anche i loro dispositivi avrebbero smesso di vibrare e avrebbero visualizzato la notifica della presa in carico. Inoltre per te, operatore sanitario, è possibile accedere dal menu principale dell'applicazione all'elenco degli allarmi che hai preso in carica: non puoi dimenticartene facilmente!

- *"Task 5"*: Ok operatore sanitario, hai preso in carica una richiesta, ti chiedo dunque di avvicinarti all'ospite, di sentire di cosa ha bisogno e di terminare l'allarme. Devi attendere un attimo: si sta verificando che tu sia effettivamente un operatore sanitario e che non sia stato l'ospite a chiedere la terminazione per sbaglio. Ok, ben fatto! L'allarme è terminato.

- *"Task 6"*: Come detto si possono verificare allarmi per tre motivi. I restanti due non richiedono alcuna interazione degli ospiti, ma rilevano automaticamente delle situazioni di pericolo, in particolare attacchi epilettici e cadute. Ora, ospite, ti chiedo di non badare al tuo orologio (l'applicazione "Health"

può anche essere chiusa), ma di simulare un attacco epilettico agitando il braccio per alcuni secondi. Come potete vedere si è ritornati nella situazione di un allarme in corso.

- *"Task 7"*: Ora operatore, prendi di nuovo in carico la richiesta. Questa volta però non andare a soccorre l'ospite, facciamo finta che tu te ne sia dimenticato, e attendiamo circa un minuto. Scaduto questo tempo il sistema riconosce che nessuno è andato a soccorrere l'ospite e l'allarme cessa di essere preso in carico: il sistema lo notifica di nuovo. La durata di questo periodo di tempo è regolabile dal sito web amministrativo, dalla sezione "Impostazioni Generali".

- *"Task 8"*: Ok operatore, riprendi in carico l'allarme e avvicinati all'ospite. Prima di chiedere la terminazione dell'allarme ti chiedo però di staccare il Bluetooth (se non sai come fare lo faccio io, tranquillo, è solo per simulare un guasto). Ok operatore, prova pure a terminare l'allarme. Ecco che il sistema sta facendo una ricerca di prossimità, ma questa volta non va a buon fine, perché non vengono rilevati dispositivi vicini: l'allarme non è terminato. Ora puoi riaccendere il Bluetooth.

- *"Task 9"*: In situazioni del genere, quando cioè non vi è possibile terminare un allarme per qualche motivo, ci sono due metodi. Il primo, che garantisce sicuramente la terminazione dell'allarme (a meno che Internet non funzioni!), è farlo dal sito web amministrativo, il secondo è di effettuare il reset dall'applicazione per lo smartphone dell'ospite.

Grazie mille di aver permesso questo test approfondito, il vostro aiuto sarà preziosissimo per migliorare il sistema. Vi chiediamo ancora gentilmente alcuni minuti di tempo per il questionario finale.

A.6 Questionario iniziale

Sesso:

1. Maschio
2. Femmina

Età:

1. 16-25
2. 26-35
3. 36-45
4. 46-55
5. 55+

Età:

1. 16-25
2. 26-35
3. 36-45
4. 46-55
5. 55+

Professione:

.....

Esperienza:

1. Meno di un anno
2. 1-2 anni
3. 2-5 anni
4. 5-10 anni
5. 10+ anni

Esperienza nell'uso di un computer:

1. Nessuna
2. Poca
3. Sufficiente
4. Buona
5. Eccellente

Esperienza nell'uso di smartphone o tablet:

1. Nessuna
2. Poca
3. Sufficiente
4. Buona
5. Eccellente

Esperienza nell'uso di dispositivi wearable come gli smartwatch:

1. Nessuna
2. Poca
3. Sufficiente
4. Buona
5. Eccellente

A.7 Questionario finale

Alla luce del test, come valuti l'utilità del sistema?

1. Nessuna
2. Poca
3. Sufficiente
4. Buona
5. Eccellente

Secondo te il sistema potrebbe migliorare la condizione lavorativa all'interno delle RAF?

1. Per niente
2. Poco
3. Abbastanza
4. Sì
5. Molto

Alla luce del test, è un peso o una preoccupazione il dover portarsi dietro un orologio e un telefono?

1. Molto
2. Sì
3. Abbastanza
4. Relativamente
5. Per niente

Come valuti le interfacce grafiche con cui hai avuto a che fare (applicazioni per gli smartwatch, applicazioni per gli smartphone e sito web)?

1. Per niente intuitive
2. Scarsamente intuitive
3. Abbastanza intuitive
4. Intuitive
5. Molto intuitive

Come definiresti l'usabilità del sistema?

1. Sistema molto difficile da utilizzare
2. Sistema difficile da utilizzare
3. Sistema abbastanza facile da utilizzare
4. Sistema facile da utilizzare

5. Sistema molto facile da utilizzare

Le modalità con cui puoi prendere in carico una richiesta di assistenza e terminarla rispondono alle tue esigenze?

1. Per niente
2. Non a sufficienza
3. Sì

Le modalità con cui si può richiedere assistenza rispondono alle esigenze degli ospiti?

1. Per niente
2. Non a sufficienza
3. Sì

Bibliografia

[1] World Health Organization, *"http://www.who.int/topics/disabilities/en/"*, Ultima visita il 2/09/2015

[2] Organizzazione Mondiale della Sanità (OMS), *"Classificazione Internazionale del Funzionamento, della Disabilità e della Salute (ICF)"*, 2001

[3] World Health Organization, *"Disability World Report"*, 2011

[4] Dichiarazione di Madrid, *"http://unipd-centrodirittiumani.it/public/docs/dich_madrid.pdf"*, Ultima visita il 27/08/2015

[5] La Convenzione delle Nazioni Unite sui diritti delle persone con disabilità, *"http://www.governo.it/backoffice/allegati/42085-5202.pdf"*, Ultima visita il 27/08/2015

[6] Aced Lopez S., Corno F., De Russis L., *"Supporting Caregivers in Assisted Living Facilities for Persons with Disabilities: a User Study"*, 2015

[7] Pebble Company, *"https://getpebble.com"*, Ultima visita il 10/09/2015

[8] De Buysere K., Gajda O., Kleverlaan R., Marom D., *"A Framework for European Crowdfunding"*, 2015

[9] Android open source project, *"http://developer.android.com"*, Ultima visita il 2/09/2015