ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA

CAMPUS DI CESENA

SCUOLA DI SCIENZE

CORSO DI LAUREA IN SCIENZE E TECNOLOGIE INFORMATICHE

SVILUPPO DI UN'APPLICAZIONE ANDROID PER L'ACCESSIBILITÀ URBANA DEGLI EDIFICI E DELLA CITTÀ

Relazione finale in:

Sistemi Operativi

Relatore:

Prof.ssa Paola Salomoni

Presentata da:

Eugenio Berardi

Correlatore:

Dott.ssa Catia Prandi

Sessione III

Anno Accademico 2012/2013

Indice

Introduzione

Il progetto di tesi è consistito nello sviluppo di un'applicazione Android dedicata al supporto dell'accessibilità urbana. Questo è un tema fondamentale per la società odierna e di conseguenza per la ricerca, in quanto è necessario adattare le città e i suoi servizi, anche attraverso lo sviluppo di nuove tecnologie, alle esigenze di tutti i tipi di cittadini, compresi coloro che presentano difficoltà nel superamento di quelle che sono considerate barriere architettoniche. Queste ultime si possono definire come tutti gli elementi fisici che in determinate situazioni possono rappresentare un ostacolo alla qualità di vita di alcune categorie di persone, specialmente quelle con limitata capacità motoria o sensoriale. Si potrebbero superare tali impedimenti in modo più efficiente attraverso il sorgere di una Smart City, il cui concetto idealistico consiste in una città che sfruttando al meglio le sue possibilità, sia tecnologiche che economiche, crei delle condizioni ottimali di vita per i suoi abitanti, coinvolgendoli nel processo di evoluzione. Vi sono già esempi del successo di questa ideologia nella società odierna, sia all'estero che in Italia. È in questo contesto che si vuole inserire la nostra applicazione che, attraverso un sistema di raccolta di segnalazioni di utenti, si propone di creare una comunità che tragga vantaggio dalla condivisione di esperienze altrui. Tale attivita è definita come "crowdsourcing", che come si può dedurre dall'appellativo, consiste nel permettere ai cittadini di dare un contributo al miglioramento di un qualsiasi tipo di servizio attraverso la rete. È risultato un metodo efficace, in quanto la possibilità di rimanere anonimo e la facilità di utilizzo portano l'utente a sentirsi libero di esprimere le proprie opinioni in modo disinteressato. Alla raccolta di dati contribuiscono anche l'insieme di sensori presenti oggi nelle nuove tecnologie, come i dispositivi mobile con fotocamere o accelerometri, con i quali le persone interagiscono quotidianamente. Quest'interazione viene chiamata "participatory sensing", la cui struttura teorica è una rete formata da tanti "nodi", che consistono nell'identificazione dell'utente con il suo sensore. Questi ultimi partecipano così alla condivisione di informazioni che serviranno a potenziare i servizi offerti alla società. Si può notare perciò come i dispositivi mobile connessi alla rete siano fondamentali per lo sviluppo delle Smart City e di una qualità della vita migliore.

L'applicazione è stata sviluppata per smartphone e tablet, la cui continua evoluzione li ha portati a diventare uno strumento sostituivo e a volte più efficace di altre tecnologie già esistenti, predisposte al supporto di persone con disabilità. Vengono costantemente sviluppati software per facilitare ogni aspetto della vita quotidiana degli utenti. Nel nostro caso si è scelto di creare l'applicazione per il sistema operativo Android, la cui natura open source e i bassi costi di produzione lo rendono un candidato ideale allo scopo della tesi. Considerando che i dispositivi aventi Android come sistema operativo sono al momento i più diffusi al mondo, si è presentata con il suo utilizzo la possibilità di raggiungere il maggior numero di potenziali utenti possibile. L'obiettivo del progetto di tesi è stato quello di creare un'applicazione la cui caratteristica principale fosse quella di visualizzare su una mappa e inviare segnalazioni sull'accessibilità dei luoghi. Questo può avvenire solo attraverso la creazione di un'interfaccia, che permetta agli utenti di compilare un questionario per esprimere le proprie opinioni sulle barriere architettoniche incontrate. Concretamente all'interno di tale questionario sono state inserite domande specifiche sullo stato di accessibilità di un certo luogo, come ad esempio la presenza di ascensori o parcheggi per disabili. Inoltre è presente un campo di testo, che permette all'utente di lasciare un commento, così che possa indicare particolari specifici non inclusi nel questionario. Per rendere le segnalazioni disponibili pubblicamente l'applicazione necessita di un database, che in questo caso è stato implementato attraverso il servizio Google Fusion Table, dove vengono memorizzate tutte le informazioni di interesse. Fusion Table è stato progettato appositamente per facilitare l'utilizzo e la condivisione di risorse, il che lo rende uno strumento ideale per il progetto. I dati vengono poi visualizzati sul dispositivo attraverso Google Maps, un servizio gratuito per la rappresentazione di mappe. Per consentire agli utenti di consultare delle informazioni di interesse specifiche, è stato pensato un sistema di filtri, che ha lo scopo di mostrare sulla mappa solo a quei luoghi che corrispondono alla categoria o preferenza indicata. L'applicazione sfrutta i servizi offerti da Foursquare per ottenere dati di locali sui quali gli utenti vogliono inviare una segnalazione, un social network divenuto popolare negli ultimi che si dimostra utile per la grande quantità di informazioni già presenti nei suoi database. Foursquare è stato utilizzato nel progetto anche per permettere di integrare l'applicazione con l'eventuale account posseduto dall'utente sul social network, in modo da sfruttare le sue funzionalità più sociali come il check-in e i tip, diffondendo così le segnalazioni inviate in modo più efficace ed esteso, attraverso un mezzo esterno all'app.

Il documento di tesi è composto da tre capitoli, che in successione analizzano gli aspetti teorici del progetto fino ad arrivare all'implementazione vera e propria dell'applicazione. Nel primo capitolo vediamo definito il tema dell'accessibilità, seguito dalle problematiche derivanti dalle barriere architettoniche, accennando alle normative presenti in Italia e all'estero sull'argomento. Successivamente viene spiegato il concetto di Smart City e le tecniche che vengono utilizzate allo scopo di migliorare la qualità di vita nelle città. Vengono introdotti i concetti di crowdsourcing e participatory sensing, ideologie alla base delle Smart City, i quali sfruttano le tecnologie, in particolare quelle mobile, per raccogliere informazioni di interesse sulle città e i suoi abitanti, allo scopo di creare una società sostenibile. Al termine del capitolo viene introdotto uno studio sul grado di accessibilità degli smartphone presenti oggi sul mercato, analizzando anche le tecnologie assistive disponibili per questi dispositivi, che li rendono accessibili a chiunque voglia usufruirne.

Nel secondo capitolo si descrivono le tecnologie e i servizi utilizzati durante la creazione del progetto. In particolare viene presentato Android ed il suo funzionamento, essendo quest'ultimo il sistema operativo per cui è stata sviluppata l'applicazione. In seguito troviamo una breve analisi di tutti i servizi impiegati nel progetto: Foursquare utilizzato per ottenere informazioni sui luoghi di interesse dell'utente; Fusion Table il cui scopo è stato quello di memorizzare i dati raccolti; Google Maps che permette di visualizzare le segnalazioni sulla mappa. Infine vengono descritte le tecnologie intermedie utilizzate per far comunicare fra loro gli strumenti spiegati in precedenza, tra cui il protocollo HTTP sfruttato per le richieste ai servizi in rete e JSON impiegato per analizzare i dati ottenuti in risposta.

Nel terzo capitolo viene presentata in modo dettagliato l'implementazione del progetto. Inizialmente vengono definite le classi principali create per progettare le funzionalità di base del software e per interagire con i servizi scelti. Successivamente viene descritto il funzionamento e l'aspetto dell'applicazione sviluppata. Per una rappresentazione più completa del progetto realizzato si è pensato all'aggiunta di frammenti di codice, che dimostrano il comportamento del software in alcuni particolari situazioni ed a screenshot delle sue schermate, le quali permetteranno al lettore di visualizzare l'interfaccia creata e quindi di avere un riferimento visivo di ciò che è stato esposto nel corso della tesi.

1 Accessibilità e Smart City

In questo capitolo sarà inizialmente introdotto il tema dell'accessibilità e le problematiche legate alle barriere architettoniche, con una breve storia riguardante le leggi e le normative attuate in Italia per risolverlo. Verrà anche fatto un accenno sulle normative che affrontano questi temi all'estero, per osservare come altri paesi hanno deciso di affrontare il problema.

In seguito si parlerà della nascita della concezione di città intelligente, in inglese Smart City, dei suoi scopi e di come utilizza le nuove tecnologie per migliorare la qualità della vita dei cittadini. Verranno definiti i concetti di crowdsourcing e di participatory sensing, tecniche che permettono di sfruttare i dispositivi mobile e le tecnologie fornite dalle Smart City per raccogliere informazioni sull'accessibilità dei punti di interesse.

Infine, si parlerà del grado di accessibilità, in base ai servizi che offrono, degli smartphone e dei dispositivi mobili in generale che hanno avuto una diffusione e uno sviluppo esponenziale negli ultimi anni. Verrà fatto notare come possano anche essere d'aiuto in modo considerevole a persone con disabilità di qualsiasi tipo e con quali mezzi, sia hardware che software, essi possano potenzialmente diventare inclusivi e accessibili a chiunque voglia usufruirne.

1.1 Accessibilità: barriere architettoniche

L'accessibilità è stata definita nel secondo articolo del Decreto Ministeriale n. 236 del 1989 come la possibilità, anche per persone con ridotta o impedita capacità motoria o sensoriale, di raggiungere un edificio e le sue singole unità immobiliari e ambientali, di entrarvi agevolmente e di fruire di spazi e attrezzature in condizioni di adeguata sicurezza e autonomia. In sostanza, in qualsiasi situazione chiunque deve poter accedere ad un edificio o struttura se lo desidera. Nel caso dell'accessibilità urbana, il principale ostacolo sono le barriere architettoniche.

Una barriera architettonica viene definita come un qualunque elemento costruttivo che impedisca, limiti o renda difficoltosi gli spostamenti o la fruizione di servizi (specialmente di persone con limitata capacità motoria o sensoriale). [BA06]

Ne consegue che elementi che non costituiscono barriere architettoniche per un individuo potrebbero essere invece di ostacolo per un altro: si prenda come esempio una qualsiasi scala, essa può rappresentare un ostacolo insuperabile per chi soffre di disabilità motorie o una fonte di affaticamento anche per chi riesce a salirvi, ma in condizioni di svantaggio e fatica, potrebbe quindi diventare una fonte di pericolo in innumerevoli casi. Allo stesso tempo però potrebbe non rappresentare un ostacolo per chi soffre di disabilità visiva ma rappresentare invece un punto di riferimento per orientarsi nella città. Si evince quindi che il concetto di barriera viene percepito in maniera diversa da ogni individuo. Secondo stime elaborate dall'Organizzazione Mondiale della Sanità (OMS) le persone che hanno difficoltà, più o meno sensibili, nello spostarsi nell'ambito del tessuto urbano e del territorio più in generale, rappresentano una percentuale molto elevata, stimata in oltre il 20% della popolazione. Da ciò è emerso il bisogno dei cittadini di spazi in cui le persone sentano di potersi muoversi liberamente, città senza ostacoli, ovvero ambienti privi di barriere architettoniche e facilmente accessibili da chiunque. [VES06]

1.1.1 Le normative in Italia

Si vuole ora presentare sinteticamente, le normative disponibili in Italia sull'argomento delle barriere architettoniche.

Sono presenti numerose leggi, decreti e direttive aventi lo scopo di regolamentare il tema dell'accessibilità. In passato erano presenti due differenti filoni normativi: uno dedicato agli edifici pubblici e l'altro a quelli privati. Con il D.P.R. n. 503 del 24 luglio 1996 questi due filoni sono stati uniti riconducendo tutte le norme ad un unico riferimento (D.M. 236/89) che verranno definite nelle seguenti sotto sezioni del testo di tesi. Gli edifici di proprietà pubblica, come ad esempio università, scuole, ospedali o stazioni ferroviarie dovevano applicare le norme ed i parametri di un obsoleto filone normativo che faceva riferimento al D.P.R. n. 384/78. Quest'ultimo è stato abrogato e in seguito sostituito con il D.P.R. n. 503, con titolo "Regolamento recante norme per l'eliminazione delle barriere architettoniche negli edifici, spazi e servizi pubblici". Il D.M. n.236/89 si occupava invece di regolamentare gli edifici di proprietà privata e quelli di edilizia residenziale. Ad esempio, in questo caso, venivano presi in considerazione, oltre agli edifici residenziali, uffici, teatri, centri commerciali o negozi.

1.1.1.1 D.P.R. n. 384/78

Questo decreto è stato approvato come regolamento per l'esecuzione e attuazione delle disposizioni contenute nell'articolo 27 della legge 30 marzo 1971, n. 118. Aveva come fondamento l'idea che per l'accessibilità potesse essere stabilita una dimensione ottimale degli spazi costruiti in qualsiasi tipo di situazione e questa non può considerarsi una modalità corretta. Ad esempio, si prevedevano ascensori di dimensioni minime eccessive, quindi con necessità di spazi e finanziamenti considerevoli, scoraggiando molte Amministrazioni Comunali, specialmente quelle di piccola entità, a modificare i propri edifici in base a tali parametri. Per questi motivi le leggi e norme che fin dal 1971 mantengono l'obbligo dell'accessibilità per gli edifici pubblici e per quelli privati aperti al pubblico, mediante l'eliminazione delle barriere architettoniche, spesso non hanno avuto alcun effetto. Il D.P.R. 384/78 è rimasto in vigore, con tutte le sue limitazioni, fino alla sua abrogazione nel 1996, ma continuava ad essere un riferimento obbligato per i progetti edili, anche se considerato da tempo un provvedimento antiquato.

1.1.1.2 Legge n.41 del 1986

Il 28 febbraio 1986 è stata emanata la legge n.41 e in particolare l'articolo 32 ha introdotto molte importanti novità. Il testo di legge impone che non possono essere approvati progetti di opere pubbliche che non rispettino le norme sulla eliminazione delle barriere architettoniche e non possono essere finanziati con denaro pubblico progetti o opere edilizie che siano difformi da quanto prescritto dalla normativa in vigore per l'accessibilità. Inoltre l'articolo obbliga tutti gli Enti Pubblici a definire uno specifico piano per l'eliminazione delle barriere architettoniche degli edifici già esistenti e non ancora adeguati alle norme del D.P.R. 384/1978. È interessante notare come questa sia una legge finanziaria dello Stato Italiano, la prova di come una città piena di ostacoli possa causare danni ingenti anche a livello economico.

1.1.1.3 Legge n.13 del 1989

La legge n.13/89 ha come titolo "Disposizioni per favorire il superamento e l'eliminazione delle barriere architettoniche negli edifici privati" e da ciò si rende evidente che le disposizioni per garantire l'accessibilità degli spazi vengono estese anche ad edifici privati, residenziali e non, in sede di nuova costruzione o di ristrutturazione. Al contrario, la legislazione precedente, cioè l'art.27 della legge 118/71, riguardava solo opere, edifici

pubblici o privati aperti al pubblico. Ogni progetto edilizio da questo momento dovrà quindi essere pensato per essere accessibile e visitabile da chiunque.

1.1.1.4 D.M. n.236 del 1989

Dopo poco tempo, è stato emanato il regolamento di esecuzione per la legge n.13, il D.M. 236/89. Questo provvedimento fornisce consistenti innovazioni e non parla solo di norme legate ad ostacoli che limitano il movimento, ma definisce anche quelle aventi lo scopo di migliorare il comfort ambientale e la sicurezza, quindi all’eliminazione delle fonti di disagio e pericolo. Interessante notare come nell’articolo 2 vengano definiti i concetti fondamentali di barriere architettoniche, accessibilità, visitabilità e adattabilità, riportati qui di seguito:

- Barriere architettoniche: ostacoli fisici che sono fonte di disagio per la mobilità di chiunque ed in particolare di coloro che, per qualsiasi causa, hanno una capacità motoria ridotta o impedita in forma permanente o temporanea; ostacoli che limitano o impediscono a chiunque la comoda e sicura utilizzazione di parti, attrezzature o componenti; la mancanza di accorgimenti e segnalazioni che permettono l'orientamento e la riconoscibilità dei luoghi e delle fonti di pericolo per chiunque e in particolare per i non vedenti, per gli ipovedenti e per i sordi.
- Accessibilità: la possibilità, anche per persone con ridotta o impedita capacità motoria o sensoriale, di raggiungere l'edificio e le sue singole unità immobiliari e ambientali, di entrarvi agevolmente e di fruirne spazi e attrezzature in condizioni di adeguata sicurezza e autonomia.
- Visitabilità: la possibilità, anche da parte di persone con ridotta o impedita capacità motoria o sensoriale, di accedere agli spazi di relazione e ad almeno un servizio igienico di ogni unità immobiliare. Sono spazi di relazione gli spazi di soggiorno o pranzo dell'alloggio e quelli dei luoghi di lavoro, servizio ed incontro, nei quali il cittadino entra in rapporto con la funzione ivi svolta.
- Adattabilità: la possibilità di modificare nel tempo lo spazio costruito a costi limitati, allo scopo di renderlo completamente ed agevolmente fruibile anche da parte di persone con ridotta o impedita capacità motoria o sensoriale.

1.1.1.5 Legge n.104 del 1992

Uno degli ultimi importanti provvedimenti legislativi relativi all'eliminazione delle barriere architettoniche in Italia è stata la legge n.104 del 5 febbraio 1992, chiamata "Legge-quadro per l'assistenza, l'integrazione sociale e i diritti delle persone handicappate" che integra le norme contenute nelle precedenti disposizioni, modificandole notevolmente. Detta inoltre che tutte le opere realizzate negli edifici pubblici e privati aperti al pubblico che non rispettino le disposizioni vigenti sull'accessibilità, tali da renderne impossibile l'uso, debbano essere dichiarate inabitabili e inagibili. Questa legge rappresenta un notevole passo avanti sull'agevolazione dell'accessibilità urbana e sull'eliminazione degli ostacoli architettonici.

1.1.1.6 D.P.R. n.503 del 1996

All'interno del D.P.R. n.503 del 24 luglio 1996 è presente il "Regolamento recante norme per l'eliminazione delle barriere architettoniche negli edifici, spazi e servizi pubblici" che sostituisce ed integra il precedente D.P.R. n. 384 del '78 con cui vengono unificati i due precedenti filoni normativi sull'accessibilità urbana e sull'eliminazione delle barriere architettoniche. Viene esteso anche agli spazi ed edifici pubblici quanto descritto nel D.M. 236/89 e sancisce che ogni spazio costruito, aperto o chiuso, pubblico o privato, debba seguire gli obblighi di tipo prestazionale presenti nella normativa, proponendo inoltre, in sede di progettazione, soluzioni alternative per garantire una migliore qualità degli esiti ottenuti per migliorare l'accessibilità della struttura.

Viene anche ribadito lo stesso concetto del D.M. 236/89 che gli ostacoli fisici costituiscono fonte di pericolo e fonte di disagio o affaticamento per chiunque e non servono soltanto a favorire la vita delle persone disabili, ma più in generale di rendere più sicuro e confortevole l'habitat dell'uomo.

1.1.2 Le normative all'estero

Vengono ora definite brevemente delle leggi e delle normative presenti in diversi Paesi Esteri riguardanti l'accessibilità e le barriere architettoniche.

1.1.2.1 In America

In America hanno affrontato l'abbattimento delle barriere architettoniche presenti nelle infrastrutture pubbliche con largo anticipo rispetto ai Paesi d'Europa. In questo contesto, il testo di riferimento è il "American with Disability Act" (ADA), approvato nel 1990 dall'allora presidente George H. W. Bush, che ha posto le fondamenta per i diritti e la non discriminazione dei cittadini americani con disabilità in qualsiasi ambito della loro vita quotidiana. La legge definisce una persona disabile come un individuo con un impedimento che limita alcuni degli aspetti della sua vita, come studio o lavoro. Non viene specificata nessuna categoria di disabilità quindi, ad esempio, anche una persona anziana con delle limitazioni è coperta dal testo di legge.

L'ADA è composto da cinque titoli differenti: i primi due riguardano il lavoro e le attività della Pubblica Amministrazione, mentre il terzo e il quarto spiegano come facilitare gli accessi ai trasporti e ai luoghi pubblici con una serie di linee guida apposite. Il quinto e ultimo titolo invece si occupa dell'accessibilità per i principali mezzi di comunicazione.

1.1.2.2 In Inghilterra

Nel Regno Unito, la legge principale per l'accessibilità nel campo dell'edilizia è la parte M delle Building Regulations, un insieme di norme che devono rispettare i progetti di costruzione. Nel dettaglio, la parte M si occupa le condizioni di accesso, sia esterno che interno, di edifici nuovi e di corpi aggiunti ad edifici già esistenti. Da un altro punto di vista, è stato attuato il Disability Discrimination Act (DDA) del 1995 che impone ai fornitori di beni, servizi e strutture di garantire l'assenza, nei propri locali, di eliminare possibili elementi discriminatori nei confronti di eventuali clienti disabili.

Nel 2010, il DDA è stato sostituito dall'Equality Act, una legge contenente regolazioni per un gran numero di tipi diversi di discriminazione, incluse quelle riguardo alla disabilità. [ZEC12]

1.1.2.3 In Francia

In Francia, è presente la Legge d'orientamento del 30 giugno 1975 che si occupa di definire le norme di responsabilità nei confronti delle persone con disabilità, fondate sulla solidarietà.

Il testo di legge francese definisce come obbligo nazionale il dovere di garantire l'accessibilità per tutti le strutture. Infatti, stabilisce che la disposizione architettonica e la sistemazione degli edifici aperti al pubblico devono garantire l'accesso alle persone con disabilità. Purtroppo, per via del ritardo nell'emanazione dei decreti esecutivi, da oltre trent'anni la legge non viene adeguatamente seguita ed esistono ancora numerose strutture pubbliche non completamente accessibili.

Il 13 luglio del 1991 è stata emanata un'ulteriore legge che ha integrato le norme riguardanti gli edifici, in cui venivano definite diverse misure destinate a favorire l'accessibilità dei locali abitabili, pubblici e dei luoghi di lavoro. Questa legge è entrata effettivamente in vigore nel 1994. [MOR06]

1.2 Smart City

In questa sezione, si vuole introdurre il concetto di Smart City, sebbene non esista ancora una definizione univoca di questo ideale ambiente urbano. Possiamo avvicinarci ad un'idea del concetto di Città Intelligente attraverso il progetto europeo "Smart Cities" [SC06], che stila una classifica delle città valutandone il grado di smartness, attraverso sei parametri principali:

- Economia: spirito innovativo, imprenditorialità, produttività, flessibilità del mercato del lavoro e capacità di trasformarsi.
- Governabilità: partecipazione nelle decisioni, servizi pubblici e sociali, trasparenza.
- Ambiente: attrattiva delle condizioni naturali, inquinamento, protezione dell'ambiente, gestione delle risorse.
- Popolazione: flessibilità, creatività, cosmopolita, livello di qualifica, pluralità sociale ed etnica.
- Qualità di vita: edifici pubblici, sicurezza, attrattiva turistica, coesione sociale, strutture per l'educazione e per la salute.
- Mobilità: accessibilità, presenza di una infrastruttura ICT, sistemi di trasporto innovativi, sicuri e sostenibili.

Le tecnologie dell'informazione e della comunicazione (ICT, acronimo dall'inglese Information and Communication Technologies) che si stanno sempre più

diffondendo ed evolvendo, possono essere sfruttate per far diventare le nostre città intelligenti. Infatti, le ICT sono già il nucleo di molti modelli per lo sviluppo urbano, ad esempio per rinnovare infrastrutture, per controllare il traffico stradale o per monitorare l'inquinamento ambientale. Perciò possiamo anche definirla città *smart* quando gli investimenti nella società, nelle infrastrutture ICT e di trasporto servono ad avere una crescita economica sostenibile ed a migliorare la qualità di vita delle persone, con una intelligente gestione delle risorse. Quindi è necessario un salto tecnologico in campo ICT per potenziare le esistenti infrastrutture per il futuro sviluppo delle Smart City. Questo salto tecnologico può essere fatto considerando le Smart City come strutture all'avanguardia nella visione del Future Internet (FI, dall'inglese Internet del futuro) [JOS14]. Sebbene non ci sia una definizione universale di Future Internet, può essere interpretato come "un sistema socio-tecnologico composto da informazioni e servizi accessibili tramite Internet, associato all'ambiente fisico e al comportamento umano e di sostegno ad applicazioni intelligenti di sociale importanza".

Quindi il Future Internet può trasformare una Smart City in una piattaforma aperta ed innovativa, che supporti lo sviluppo delle infrastrutture basate sulle nuove tecnologie. I più importanti pilastri del FI per un'ambiente Smart City sono: Internet of Things (IoT), Internet of Services (IoS) e Internet of People (IoP).

1.2.1 Internet of Things

Internet of Things è una nuova visione in cui viene definita una rete globale e dinamica con capacità di configurazione automatica, basata su protocolli di comunicazione standard e interoperabili dove le "cose" fisiche e virtuali hanno identità, attributi fisici, personalità virtuali e usano interfacce intelligenti, rimanendo perfettamente integrate nella rete.

Nell'Internet of Things, ci si aspetta che le "cose" diventino partecipanti attivi nei processi economici, sociali e informativi dove sarà permesso loro di interagire e comunicare a vicenda e con l'ambiente circostante. Questo avviene attraverso lo scambio di dati e informazioni, che le "cose" rilevano mentre reagiscono autonomamente agli eventi del mondo reale, influenzandolo con processi che scatenano azioni e creano servizi, con o senza ricevere direttamente un intervento umano. Una Smart City deve essere capace di sfruttare questo servizio, che le permette di essere connessa alla rete e quindi più efficiente e dinamica.

Le interfacce sotto forma di servizi servono a facilitare le interazioni con queste "cose intelligenti" attraverso Internet, richiedendo e cambiando il loro stato e tutte le informazioni a loro associate, tenendo sempre in considerazione problematiche di sicurezza e privacy.

Nella pratica, consisterebbe nel catalogare ogni oggetto tramite identificazione a radio frequenza, codici a barra o codici QR (in inglese Quick Response) in modo da poterli interfacciare con la rete. [IOT14]

Un'applicazione di questa teoria è stata fatta a Varese con il progetto Smart City, ideato e realizzato dal Lab#ID, il laboratorio RFID della LIUC, Università di Cattaneo con lo scopo di rilanciare le imprese artigianali, commerciali e turistiche. Il progetto consiste in un'architettura aperta, grazie alla quale sono stati attivati servizi fruibili da cittadini e visitatori attraverso smartphone dotati di tecnologia NFC (Near Field Communication, in italiano comunicazione in prossimità). Sono stati apposte ad oltre 500 punti di interesse, fra cui vetrine dei negozi, edifici pubblici, fermate di autobus e luoghi significativi della città, delle vetrofanie NFC contrassegnate dal logo dell'iniziativa con cui, avvicinandovi lo smartphone, vengono rese disponibili informazioni riguardo al luogo. Ciò è possibile grazie all'applicazione gratuita, realizzata sempre dal Lab#ID e rilasciata sul Play Store di Android, che consente utilizzare coupon nei punti vendita aderenti e visualizzare sulla mappa le promozioni più vicine. [VSC13]

1.2.2 Internet of Services

L'idea di base del concetto di Internet of Services è quella di facilitare l'armonizzazione di varie applicazioni in servizi interoperabili e nell'uso della semantica per la combinazione e l'uso dei dati e delle informazioni che restituiscono i vari servizi. Consiste quindi nell'abbattere le barriere che esistono nell'accedere e fornire questi servizi e allo stesso tempo nell'aumentarne le possibilità non solo per i consumatori, ma anche per le aziende che ne vorranno fare uso.

1.2.3 Internet of People

Internet of People consiste nel considerare le persone come parte di una rete intelligente di sensori onnipresente con il potenziale di connettersi, interagire e scambiare informazioni su se stessi e sul proprio contesto sociale e dell'ambiente circostante.

1.2.4 Applicazioni Mobile per Smart City

Per raggiungere il pieno potenziale delle Smart City, la città deve diventare una piattaforma in modo da alimentare la creatività, lo scambio delle informazioni e la creazione di applicazioni da parte degli sviluppatori. In questo modo, la città diventa come Internet: un connettore, un mezzo per valorizzare il potenziale dei cittadini.

Le applicazioni per Smart City sono simili a quelle convenzionali, però insieme ai dispositivi mobile che solitamente contengono sensori, come accelerometro, sensore di prossimità o giroscopio, questo software può interfacciarsi con l'hardware che permette di ricavare dati utili riguardo a particolari luoghi della città.

Un' applicazione per Smart City può avere sette caratteristiche:

- Uso dei sensori: li utilizza per ricavare dati dall'ambiente circostante.
- Collegabile: il dispositivo connesso alla rete porta le informazioni rilevate su Internet.
- Accessibile: le informazioni del nostro ambiente vengono pubblicate sul Web e possono essere accessibili all'utente.
- Onnipresente: l'utente può accedere all'informazione che desidera attraverso Internet, in qualsiasi momento e in qualsiasi luogo.
- Sociale: l'utente può condividere dati attraverso un social network.
- Condivisibile: l'oggetto di interesse deve essere accessibile e indirizzabile e quindi condivisibile come in una rete con un'infrastruttura di tipo Peer 2 Peer (dall'inglese, pari a pari).
- Visibile/realtà aumentata: rendere l'informazione nascosta visibile attraverso il dispositivo mobile.

Le possibili implementazioni di applicazioni per Smart City possono riguardare le seguenti tematiche: collaborazione, condivisione di risorse, Open data, tecnologia, cultura, sostenibilità, salute.

Le comunità sono un'importante componente delle applicazioni per Smart City, agendo attraverso mezzi come il crowdsourcing e participatory sensing. Nelle seguenti sezioni verranno definite alcune tecniche predisposte alla raccolta di dati di interesse in una Smart City, associandole ad esempi di città che attualmente sfruttano dispositivi mobile e applicazioni per tali scopi. [ASC14]

1.2.4.1 Crowdsourcing

Il crowdsourcing (dall'inglese crowd, folla e sourcing, fonte) consiste nell'ottenere una serie di servizi, idee o contenuti da un grande gruppo di persone, in particolare sotto forma di una comunità online, invece di utilizzare una classica infrastruttura con dipendenti e fornitori. Questo processo viene spesso adoperato per dividere un lavoro che può essere considerato tedioso, oppure per raccogliere fondi destinati ad opere di beneficienza o aziende appena nate bisognose di finanziamenti e può avvenire anche offline. Vuole combinare gli sforzi di volontari o lavoratori part-time, in quanto è attraverso il contributo di ognuno che possono raggiungere il risultato desiderato. Il crowdsourcing si distingue dall'outsourcing, cioè la scelta di appaltare un certo processo di un'azienda ad un'altra, per il fatto che il lavoro proviene da un pubblico non definito, invece che essere commissionato ad un gruppo specifico.

Nella pratica, il crowdsourcing si svolge principalmente tramite Internet, essendo un'ottima piattaforma in cui gli individui tendono ad essere più aperti e più propensi a condividere dati e informazioni. Il crowdsourcing può avere due tipi di implementazione, implicita o esplicita: nella sua forma esplicita, gli utenti lavorano insieme per valutare e completare compiti condividendo informazioni e modificando il lavoro di altri. Il crowdsourcing implicito può invece avere due casi; standalone (cioè a sé stante) se le persone risolvono problemi come effetto secondario di un'altra mansione, piggyback (cioè a cavalluccio) dove le informazioni degli utenti vengono raccolte da terze parti. [CRW14]

Un interessante esempio di applicazione che usa il crowdsourcing per migliorare l'accessibilità e il superamento degli ostacoli che le persone possono incontrare ogni giorno è Accessible Way, un progetto nato in Brasile dalla collaborazione fra IBM e l'Association for Assistance to Disabled Children (AACD). Consiste in un'applicazione per dispositivi mobile gratuita in cui gli utenti possono compilare un sondaggio per valutare lo stato di accessibilità delle strade delle città brasiliane.

L'obiettivo del progetto è quello di permettere agli utenti di condividere le difficoltà relative agli ostacoli che possono incontrare nella vita di tutti i giorni per aiutare a costruire un database per la mobilità urbana brasiliana. La piattaforma include una serie di categorie con cui possono organizzare i problemi che incontrano, come marciapiedi, semafori o segnaletica per esempio. [PRI13]

1.2.4.2 Participatory Sensing

Più cresce la diffusione di dispositivi mobile e più aumentano le loro capacità catturare, classificare e trasmettere immagini, suoni, locazioni e altre informazioni, interattivamente e automaticamente. Con la giusta architettura possono agire come sensori e strumenti per collezionare dati localizzati. L'idea del Participatory Sensing è di utilizzare la proliferazione di questi dispositivi nella popolazione per formare reti di sensori interattive e creare applicazioni attraverso cui utenti qualsiasi possono raccogliere e analizzare informazioni, ad esempio riguardo all'ambiente, il meteo oppure il traffico.

Una struttura tipica di Participatory Sensing contiene i seguenti componenti:

- Nodi mobili, che sono l'unione del singolo utente e del sensore installato sul suo cellulare o tablet. I nodi sono la base di ogni applicazione del Partecipatory Sensing.
- Le richieste che vengono eseguite per ottenere le informazioni raccolte precedentemente.
- Gli operatori di rete, gestiscono le reti usare per collezionare e restituire i dati rilevati dai sensori.
- I fornitori di servizi che agiscono come intermediari fra i nodi e le richieste, in modo da restituire informazioni di interesse. [CRI12]

Un'implementazione di Participatory Sensing è stato usato nel progetto Smart City della città di Santander, città sulla costa atlantica della Spagna. Alla base di questo progetto c'è un sistema di oltre diecimila sensori che monitorano una serie di elementi: illuminazione, temperatura, umidità, traffico e spostamenti delle persone. Ogni due minuti, i dati rilevati vengono trasmessi al laboratorio IT dell'Università di Cantabria, dove vengono analizzati ponendo particolare attenzione ai livelli di inquinamento e congestione urbana. Il sistema permette quindi di conoscere in tempo reale alcune criticità come incidenti o guasti e porvi rimedio. Oltre a tutto ciò, è stata anche sviluppata un'applicazione per smartphone che permette ai residenti di Santander di ottenere immediatamente i dati rilevati dai sensori. Inoltre gli utenti possono diventare "sensori" a loro volta, avendo la possibilità di poter inviare informazioni al database. [SAS14]

1.3 Accessibilità nella tecnologia: smartphone

Negli ultimi anni la diffusione degli smartphone è cresciuta esponenzialmente, nel 2012 ne sono stati registrati più di un miliardo in uso. Ormai rappresentano quasi la metà del mercato dei dispositivi mobile. Ovviamente l'aumento della diffusione dei cellulari intelligenti è anche caratterizzata da un uso maggiore di questi dispositivi da parte di persone disabili, che hanno la possibilità di trarne numerosi benefici preziosi come facilitare la comunicazione con altre persone e quindi interagire con la società con meno sforzi e più efficacia, ma anche facilitare piccoli aspetti della vita quotidiana come gli acquisti personali, che grazie ad Internet sono veloci e non necessitano spostamenti impegnativi. I comuni smartphone di oggi possono quindi sostituire quei dispositivi molto più costosi su cui alcune persone disabili dipendevano per comunicare, per il wayfinding o anche solo per leggere.

Si nota anche uno slittamento di interesse sia degli sviluppatori che dei consumatori dai cellulari classici verso i veri e propri smartphone. I normali cellulari sono dispositivi chiusi, con software proprietario legato al telefono. Ne consegue che spesso non è possibile sviluppare, installare o lanciare applicazioni create da terze parti e che quindi tutte le funzioni per l'accessibilità dovranno essere state incorporate nel dispositivo dal produttore oppure tramite periferiche aggiuntive da connettere al cellulare. Il numero di cellulari classici nel mercato con incluse delle funzionalità per l'accessibilità è molto ridotto o quasi nullo. Invece, gli smartphone, hanno sistemi operativi più avanzati e più aperti che permettono di sviluppare e installare applicazioni create da altri sviluppatori e non solo dal produttore del dispositivo. Questo vantaggio ha permesso la creazione di nuove opportunità per tutte le persone con disabilità che possono ora avere accesso alle funzionalità dello smartphone e ad applicazioni di ogni genere che possono aiutarli in numerose istanze.

Tuttavia questa apertura degli smartphone può portare ad una frammentazione delle funzionalità disponibili, ad esempio ogni tipo di dispositivo ha un diverso sistema operativo e quindi possibilità di accessibilità diverse. Inoltre non tutte le applicazioni potrebbero essere state progettate e implementate con il pensiero rivolto all'accessibilità di tutte le persone che le useranno. Con centinaia di migliaia di applicazioni disponibili il cui uso può essere dei più disparati, come l'intrattenimento o la comunicazione, diventa fondamentale che gli sviluppatori comprendano la necessità di renderle fruibili a tutti.

Non solo dal punto di vista legale o morale, ma anche da un punto di vista economico dato che, se progettata male, ne viene precluso l'utilizzo ad un numero elevato di persone. [AVI12] Per risolvere questo problema i diversi produttori tra cui Apple, Google e Microsoft, hanno stilato una serie di linee guida per l'accessibilità. Gli sviluppatori possono seguire queste indicazioni per fare in modo che le applicazioni che stanno sviluppando sfruttino il massimo delle funzionalità del dispositivo e possano raggiungere tutta l'utenza.

Al momento, non sono disponibili linee guida definite appositamente per applicazioni mobile dal World Wide Web Consortium (W3C) ma si possono ricavare dalle linee guida già pubblicate: il Web Content Accessibility Guidelines (WCAG) e User Agent Acessibility Guidelines (UAAG), che contengono sezioni apposite riguardanti la visione di contenuto Web su dispositivi mobile, ma proprio per questo non forniscono linee guidi per le applicazioni native. [MA13] Questo ha portato diverse organizzazioni come ad esempio la British Broadcasting Corportation (BBC) [SWA13] o Funka Nu AB, una compagnia svedese che si occupa esclusivamente di accessibilità, a creare e rendere pubbliche all'uso di chiunque delle linee guida apposite per applicazioni mobile, native, ibride o basate sul Web. Queste linee guida sono frutto di numerosi studi e ricerche e tentano di dimostrare come le applicazioni possano essere progettate migliorando l'interfaccia utente e di conseguenza aumentare il loro grado di accessibilità.

1.3.1 Tecnologie assistive

Nel tempo, sono nate molte tecnologie assistive per aiutare le persone ad interfacciarsi con l'informatica, ognuna di tipo diverso a seconda del caso. Verrà ora fatto un breve elenco dei prodotti e mezzi, hardware e software, disponibili in questo momento:

- Display Braille: un dispositivo elettro-meccanico utilizzato per visualizzare i caratteri Braille, solitamente per mezzo di una raccolta di punti attraverso dei buchi su una superficie piana. Solitamente vengono visualizzate 40 o 80 celle Braille e ognuna di esse è composta da 8 punti, permettendo di rappresentare tutti i 256 caratteri della codifica Ascii estesa. Oltre alla riga in Braille, sono presenti anche un certo numero di tasti funzione, alcuni necessari per muoversi all'interno delle informazioni riportate sullo schermo e altri usati per modificare le impostazioni del display braille stesso. [BRA14]

- Screen reader: letteralmente lettore di schermo dall'inglese, è un tipo di software che identifica ed interpreta il testo mostrato sullo schermo del dispositivo, presentandolo all'utente tramite sintesi vocale o attraverso un display braille. [SCR14]
- Teletype: in passato, era un dispositivo elettromeccanico molto diffuso, impiegato per trasmettere messaggi di testo da un punto all'altro attraverso diversi canali di comunicazione. I modelli più recenti sono interamente elettronici e visualizzano il testo su uno schermo invece che stamparli sulla carta come in passato. I teletype sono ormai obsoleti ma vengono ancora utilizzati da utenti con disabilità uditive per comunicare testualmente attraverso le comuni linee telefoniche. [TTY14]
- Mouse trackball: un tipo particolare di mouse la cui componente principale è una sfera, libera di ruotare in una cavità dell'involucro dove si trovano i sensori che ne rilevano il movimento e parzialmente accessibile dall'esterno in modo che possa essere comandata dalla mano. Questo strumento non è necessariamente una tecnologia assistiva ma può essere più facile da utilizzare per un utente con disabilità motorie. Esistono esempi particolari di smartphone che contengono una trackball, ad esempio il Nexus One di Google e diversi modelli di BlackBerry. [TRA14]
- Mouth stick: è una semplice bacchetta che funge da mouse applicato alla bocca. Data la semplicità e il basso costo, è una delle tecnologie assistive più usate. Può essere usato da persone con disabilità di tipo motorie per utilizzare un touch screen o un mouse trackball.
- Head wand: ha lo stesso concetto di base del mouth stick ma la bacchetta è attaccata ad un elmetto. Tramite un head wand l'utente può interagire con il dispositivo muovendo la testa.

1.3.2 Accessibilità Apple

Apple ha rivoluzionato in modo drastico il mercato dei cellulari con l'iPhone e continua ad essere il leader tutt'oggi. L'iPhone è stato uno dei primi prodotti ad usare un touch screen come unico metodo di input e controllo del dispositivo e per questo i primi modelli rappresentavano un grosso ostacolo dal punto di vista dell'accessibilità per molte persone. Apple ha dato ascolto alle richieste della comunità disabile e con il rilascio

dell'iPhone 3GS nel 2007 è stato introdotto nel dispositivo lo screen reader VoiceOver, cioè un'applicazione software che identifica e interpreta il testo mostrato sullo schermo, presentandolo all'utente tramite sintesi vocale, includendo inoltre più di 30 lingue diverse. Apple ha progettato un insieme di gesti, cioè movimenti fatti sul touch screen che corrispondono ad una particolare azione, che anche gli utenti non vedenti posso eseguire per poter controllare lo smartphone. Ad esempio quando l'utente scorre il dito o tocca un elemento sullo schermo, verrà enunciato il nome dell'oggetto, invece quando si tocca due volte viene attivato. Facendo scorrere il dito da sinistra verso destra sullo schermo è possibile navigare attraverso gli elementi senza sapere per forza la posizione esatta dell'elemento che si sta cercando, se si scorre da destra verso sinistra invece si scorreranno gli elementi nell'ordine inverso.

Negli ultimi anni, Apple ha continuato ad introdurre sempre più caratteristiche per migliorare l'accessibilità dell'iPhone, come ad esempio:

- Metodi di input e output alternativi tramite il supporto di tastiere Bluetooth e schermi Braille.
- La tecnologia assistiva Zoom, che può allargare tutto lo schermo fino al 500 percento, è una caratteristica che funziona su tutte le applicazioni e simile al classico gesto del "pizzicare" lo schermo per ingrandire la visualizzazione.
- La funzionalità AssistiveTouch, che permette di creare gesti personalizzati e supporto a dispositivi adattativi, consentendo agli utenti di usare degli head wand o mouth stick per operare lo smartphone. Avendo l'iPhone uno schermo capacitivo, è possibile anche usare penne speciali come metodo alternativo di input.
- Un'altra funzione disponibile è quella di poter invertire i colori sullo schermo, usata spesso da persone con disabilità visive per aumentare il contrasto dei colori che potrebbe risultare più facile per la lettura.
- Il software basato su riconoscimento vocale chiamato Siri, introdotto per la prima vota con l'iPhone 4S, che permette all'utente di controllare il cellulare senza usare lo schermo.
- Completa compatibilità con dispositivi Teletype (TTY) e per gli apparecchi acustici, con la possibilità anche di personalizzare allarmi con la vibrazione.

Il continuo inserimento e aggiornamento di funzioni dimostra l'impegno di Apple per migliorare l'accessibilità dei suoi prodotti, infatti rispetto agli smartphone concorrenti l'iPhone è considerato il preferito da utenti con disabilità, specialmente da persone ipovedenti. L'organizzazione Web Accessibility In Mind (WebAIM) ha dichiarato che fra gli utilizzatori di screen reader, il 58,5 % usa prodotti Apple. [SRS14] Un altro motivo per cui l'iPhone ha acquisito tanta popolarità è che molte applicazioni per iOS sono facilmente accessibili tramite VoiceOver senza modifiche del software particolari da parte degli sviluppatori.

1.3.3 Accessibilità Android

Gli smartphone Android al momento dominano il mercato dei cellulari ma ricevono molto meno uso da parte di utenti disabili rispetto ai dispositivi iOS. Un sondaggio da parte di WebAIM sull'utilizzo di screen reader indica che solo il 7,9% dei partecipanti usa principalmente un cellulare Android.

Sebbene Google fabbrichi degli smartphone, come il Galaxy Nexus, ci sono molti altri produttori di cellulari che usano Android come loro sistema operativo, dando al consumatore la possibilità di scegliere quale prodotto preferiscono. Questo approccio però porta diversi svantaggi, ad esempio alcuni di questi cellulari fatti da altri non supportano le ultime versioni del sistema operativo e non possono essere sempre aggiornati con le ultime funzionalità disponibili. Inizialmente, questa filosofia "open source" di Android aveva portato ad un modello di accessibilità dove erano gli sviluppatori stessi delle applicazioni a dover trovare soluzioni a questa problematica, invece che averne già a disposizione. A partire dalla versione 1.6 di Android, Google ha invece cominciato ad aggiungere funzioni per l'accessibilità:

- TalkBack: servizio che con un sintetizzatore vocale esprime a "voce" il risultato di azioni, eventi o notifiche.
- KickBack: questo servizio restituisce un feedback aptico, cioè la vibrazione dei tasti quando vengono premuti.
- SoundBack: servizio che esegue suoni diversi a seconda dell'azione compiuta.

Visto che i produttori di cellulari hanno il pieno controllo dell'implementazione Android nel loro dispositivo, a volte alcuni di questi servizi potrebbero essere rimossi, ma in ogni caso possono essere riscaricati dall'Android Play Store.

Nel tempo Google ha continuato a migliorare l'accessibilità di Android nelle successive versione del sistema operativo. Ad esempio infatti una volta TalkBack funzionava solo se l'utente usava una tastiera o frecce direzionali, limitandone l'uso solo ai dispositivi che possedevano questi metodi di input. Questo problema è stato risolto creando una tastiera virtuale per i cellulari che avevano a disposizione solo il touch screen. Nella versione 4.2.1 di Android sono stati fatti molti miglioramenti per l'accessibilità, come l'aggiunta della funzione "Esplora al tocco" che permette all'utente di sentire il nome dell'oggetto che sta toccando tramite lo schermo, oltre a molte opzioni per modificare la dimensione del carattere e le impostazioni sonore del cellulare. Nelle ultime versioni è anche possibile attivare TalkBack e Esplora al Tocco con un semplice gesto sullo schermo, mentre precedentemente dovevano essere attivate andando fino al menù delle impostazioni di accessibilità. In aggiunta sono stati introdotti altri gesti come quelli per leggere l'intero schermo, attivare la schermata iniziale o per tornare a quella precedente. Grazie a tutte queste aggiunte, dalla versione 4.2 in poi è diventato veramente utilizzabile anche da persone con disabilità visive, sebbene, Android non possieda ancora un'opzione per invertire i colori.
L'architettura aperta di Android permette facilmente agli sviluppatori di creare alternative per tecnologie assistive e migliorare l'accessibilità, al contrario di iOS dove non è sempre possibile data la "chiusura" del sistema. Ad esempio, esistono molte alternative allo screen reader TalkBack, gratuite e a pagamento, tutte con caratteristiche diverse.

Sebbene ci siano dei problemi di frammentazione e limiti dell'accessibilità su Android, il sistema continua a migliorare e Google inserisce nuove funzioni apposite ad ogni versione, come il supporto nativo di schermi Bluetooth Braille e nuovi gesti per controllo non visuale.

1.3.4 Accessibilità Windows Phone 8

Sebbene la versione precedente, il Windows Phone 7, non consentisse l'uso di tecnologie assistive create da terze parti, il Windows Phone 8 invece ha un supporto migliore e più funzionalità di accessibilità. Al momento è possibile l'ingrandimento dello

schermo, aumentare il contrasto dei colori e la dimensione del carattere. Purtroppo non è presente uno screen reader e nemmeno un'interfaccia che permetta agli sviluppatori di crearlo.

Al contrario, i tablet Windows 8 Surface RT contengono le stesse funzionalità che sono presenti anche su Windows 8, quindi oltre alle funzionalità dette precedentemente è incluso anche uno screen reader. In ogni caso, il supporto non è certamente comparabile rispetto a tutte le funzionalità che offrono Apple e Google con i loro sistemi.

1.3.5 Accessibilità BlackBerry OS

Il BlackBerry è una linea di dispositivi creata dalla società canadese BlackBerry Limited (precedentemente conosciuta come Research in Motion (RIM)). Come Apple e iPhone, i cellulari sono prodotti esclusivamente dall'omonima società, usando il sistema operativo proprietario BlackBerry OS. È stato uno dei primi dispositivi ad offrire la possibilità, ad esempio, di navigare il web e usare le e-mail ed ha avuto un grande successo perché preferito da chi se ne serviva per scopi lavorativi proprio per le sue funzionalità.

Molti BlackBerry contengono funzionalità di accessibilità, sebbene il loro supporto non è stato sempre consistente nelle nuove versioni del telefono. Alcune funzionalità disponibili sono:

- Compatibilità con apparecchi acustici.
- Possibilità di personalizzare gli allarmi con la vibrazione.
- Possibilità di eseguire azioni con una sola mano.
- Supporto per invertire il contrasto dei colori.
- Personalizzazione del carattere.
- Possibilità di ingrandire il browser.
- Compatibilità con terminali Teletype.

In oltre, BlackBerry ha appena rilasciato il proprio screen reader come download gratuito dando la possibilità di accedere non visivamente ad applicazioni principali come mail, messaggi e calendario, ma solo per specifici modelli di cellulare.

2 Tecnologie utilizzate

In questo capitolo verranno descritti i servizi e le tecnologie principali utilizzate durante la realizzazione del progetto. Inizialmente si forniranno diverse informazioni su Android, sistema operativo su cui si è deciso di sviluppare l'applicazione mobile. In seguito verrà brevemente spiegato il funzionamento della sua API (Application Programming Interface), che permette di interagire con i vari servizi offerti da diversi provider che l'app utilizza per ottenere i punti di interesse vicini all'utente (attraverso Foursquare), visualizzare i dati in mappe (utilizzando Google Maps) e memorizzare i dati su tabelle tramite Google Fusion Table. Successivamente, si definiranno le tecnologie intermedie sfruttate per far comunicare fra loro questi servizi.

2.1 Android

Android [AND14] è un sistema operativo basato sul kernel di Linux e progettato principalmente per funzionare con dispositivi mobile aventi touch screen come smartphone o tablet. Il sistema operativo è nato come progetto dell'omonima azienda, Android, Inc., fondata nel 2003 con l'obiettivo di creare dispositivi mobile più intelligenti e a conoscenza della locazione e delle preferenze del proprietario. Nel 2005 Android è stata acquisita da Google, con lo scopo di utilizzarne la tecnologia per entrare nel mercato dei cellulari. Con l'aiuto di Open Handset Alliance, un consorzio di produttori che si occupa di definire degli standard aperti per i cellulari, il 5 novembre del 2007 Google ha annunciato pubblicamente Android come suo primo prodotto nel campo mobile. Il primo smartphone ad utilizzarlo è stato il HTC Dream, uscito nel 2008.

Nel 2012 Android è diventato il sistema operativo mobile più popolare al mondo, avendo il più grande numero di consumatori ed essendo leader del mercato in molti Paesi. Alla fine del 2013 la percentuale di dispositivi Android venduti globalmente è stata del 81,3% sul totale dei prodotti mobile.

2.1.1 Intefaccia

L'interfaccia di Android si basa sulla manipolazione diretta, perciò utilizzando come input dei tocchi che corrispondono approssimativamente ad azioni della vita quotidiana, come "strisciare" o "pizzicare" per interagire con gli oggetti rappresentati

sullo schermo. La risposta all'input è progettata per essere immediata e fornisce un'interfaccia fluida basata sul tocco, spesso impiegando le capacità di vibrazione del dispositivo, restituendo un feedback aptico all'utente. Hardware interno come accelerometri, giroscopi e sensori di prossimità vengono adoperati da alcune applicazioni per rispondere a particolari azioni dell'utente, ad esempio per cambiare la visualizzazione dello schermo a seconda dell'orientamento del dispositivo, oppure di sterzare con un veicolo in un gioco di corsa, simulando il controllo di un volante di un'auto.

All'avvio i dispositivi Android mostrano la schermata iniziale, simile al desktop dei computer, composta dalle icone delle applicazioni installate e da widget: immagini particolari che mostrano informazioni e contenuti che si aggiornano automaticamente, come le previsioni meteo o la casella di posta dell'utente. La schermata iniziale di Android è completamente personalizzabile, permettendo all'utente di cambiarla a suo piacimento e di modificare il contenuto delle pagine da cui è composta. Sul negozio virtuale Google Play è possibile scaricare applicazioni e temi per modificare l'aspetto dell'interfaccia del sistema operativo. Spesso i produttori di smartphone, che installano Android sui loro dispositivi, ne modificano l'aspetto per differenziarsi dai loro concorrenti.

Nella zona superiore dello schermo è presente una barra di stato che mostra informazioni sul dispositivo e la sua connettività, come lo stato di carica della batteria o la connessione ad una rete Internet. Questa barra può essere trascinata verso il basso per rivelare la schermata delle notifiche, dove le applicazioni segnalano informazioni importanti o aggiornamenti come l'arrivo di un nuovo messaggio SMS o email. La barra di stato è progettata per mostrare dati d'interesse all'utente senza disturbarlo e interrompere le azioni che sta eseguendo al momento della ricezione di nuove informazioni.

2.1.2 Applicazioni

Android possiede una libreria in continua crescita di applicazioni create da terze parti, che gli utenti possono acquisire attraverso un negozio virtuale come il Play Store di Google o l'Amazon Appstore. È comunque possibile scaricare e installare direttamente il software ottenendolo attraverso mezzi differenti, come siti ad esempio Web esterni. All'interno del Play Store è possibile sfogliare, scaricare e aggiornare le applicazioni

pubblicate da Google e da altri sviluppatori, inoltre vengono automaticamente filtrati e visualizzati solo i software compatibili con il dispositivo in uso.

Il software viene principalmente sviluppato tramite il linguaggio Java utilizzando l'Android SDK (Software Development Kit, in italiano kit di sviluppo software). Il kit contiene numerosi strumenti per lo sviluppo, come un debugger, documentazione, librerie, esempi di codice con inclusi vari tutorial e un emulatore per poter testare le proprie applicazioni. L'ambiente di sviluppo integrato (IDE, Integrated Development Kit) supportato ufficialmente è Eclipse, accompagnato dal plugin ADT (Android Development Tools). Sono disponibili molti altri strumenti per sviluppare software per Android, come il kit per lo sviluppo nativo attraverso codice C e C++, oppure framework generici per la creazione di applicazioni compatibili con più sistemi operativi mobile.

2.1.2.1 Sicurezza e privacy

Su Android, le applicazioni vengono eseguite in un'area isolata da cui non hanno accesso al resto delle risorse di sistema, a meno che non venga concesso il permesso esplicito da parte dell'utente durante l'installazione del software. Isolando e gestendo si riducono le vulnerabilità e i possibili errori che potrebbero presentare, ma per via della disinformazione degli sviluppatori e della scarsa documentazione, accade spesso che vengano richiesti permessi non necessari all'applicazione, riducendo l'efficacia del sistema.

Diverse aziende come McAfee e AVG Technologies hanno rilasciato degli antivirus per dispositivi Android, sebbene non siano molto efficaci per via dell'isolamento a cui sono costretti, limitando l'abilità di controllare il sistema da eventuali minacce. Attualmente, Google usa un sistema chiamato Google Bouncer per controllare la presenza di malware nelle applicazioni del Play Store, con lo scopo di verificare quali siano sospette e avvertire gli utenti di potenziali pericoli prima del download. Nella versione 4.2 di Android soprannominata Jelly Bean, sono state aumentate le funzioni di sicurezza, includendo un malware scanner per controllare le applicazioni non provenienti dallo Store ufficiale e un sistema di allarme che avverte l'utente se viene rilevato il tentativo di invio di un messaggio ad un servizio telefonico a pagamento, una dei disagi più comuni che vengono provocati dai malware dei sistemi Android.

2.1.3 Hardware

La piattaforma hardware principale di Android è un'architettura a 32 bit ARMv7, anche se sono già in atto progetti per il supporto di architetture x86. I dispositivi Android possono incorporare molti componenti hardware come fotocamere, GPS, sensori di orientamento, controller per giocare, accelerometri, giroscopi, barometri, magnetometri, sensori di prossimità, sensori di pressione e termometri. Alcuni di questi componenti non sono richiesti ma sono diventati standard per alcune classi di dispositivi come gli smartphone. Altri tipi di hardware invece erano inizialmente obbligatori all'interno dei dispositivi, ma nel tempo certi requisiti sono stati rilassati e completamente eliminati.

2.1.4 Open Source e licenze

Il codice sorgente di Android è disponibile con licenze gratis e open-source. Google pubblica la maggior parte del codice sotto la Licenza Apache versione 2.0, mentre le modifiche del kernel di Linux vengono pubblicate sotto la GNU General Purpose License (dall'inglese, licenza generica). Solitamente il codice Android viene sviluppato in privato da Google e in seguito viene rilasciato pubblicamente il codice sorgente della nuova versione.

2.1.5 Android API

Android fornisce un framework ricco di funzionalità per poter costruire software innovativo per dispositivi mobile. [AD14] Comunemente le applicazioni Android vengono scritte con il linguaggio di programmazione Java e in seguito l'Android SDK compila il codice in un file APK (Android Package). All'interno di un archivio APK si trova tutto il contenuto dell'applicazione che ogni dispositivo Android potrà installare.
Una volta installata, ogni applicazione vive all'interno in un'area di sicurezza isolata in quanto:

- Il sistema operativo Android è un sistema Linux multi-utente in cui ogni applicazione è un utente differente.
- Per sua natura il sistema assegna ad ogni applicazione un identificativo utente di Linux univoco. Il sistema assegnerà i permessi a tutti i file dell'applicazione in modo che solo il suo identificativo utente potrà accedervi.

- Ogni processo ha la sua macchina virtuale, perciò il codice di un applicazione viene eseguito isolato dalle altre.
- Ogni applicazione ha il suo processo Linux che viene lanciato quando uno qualsiasi dei componenti dell'applicazione viene eseguito. Tale processo viene poi chiuso quando non avrà più alcuna utilità o quando il sistema dovrà recuperare memoria per altro software.

In questo modo, il sistema di Android implementa il principio del minor privilegio, cioè ogni applicazione ha accesso solo ed esclusivamente ai componenti di cui ha bisogno. Questo crea un ambiente sicuro in cui il software non può accedere a parti del sistema per cui non ha avuto i permessi. In ogni caso, esiste la possibilità per applicazioni diverse di condividere dati e di poter accedere a servizi del sistema:

- Si può fare in modo che dei software diversi condividano lo stesso identificativo utente, in tal caso possono accedere l'uno ai file dell'altro. Per conservare le risorse del sistema, applicazioni con lo stesso identificativo possono essere eseguite con lo stesso processo e dividere la stessa macchina virtuale.
- Possono richiedere i permessi di accedere a dati esterni come i contatti dell'utente, messaggi SMS, memoria di massa portatile, fotocamera, Bluetooth e altri. Tutti i permessi devono essere approvati dall'utente al momento dell'installazione.

2.1.5.2 Componenti principali

Esistono quattro diversi tipi di componenti in un'applicazione Android, ognuno di essi ha uno scopo specifico e un ciclo di vita distinto che definisce quando il componente viene creato e distrutto. I componenti sono i seguenti:

- Activity (attività): un'activity rappresenta la singola schermata con l'interfaccia utente. Ad esempio un'applicazione per la gestione di posta potrebbe avere un'activity che mostra la lista delle nuove mail in entrata, oppure un'altra per la scrittura. Sebbene le activity lavorino insieme per formare un'esperienza coesa, ognuna di esse è indipendente dall'altra. Perciò un'applicazione differente può lanciare una qualsiasi di queste activity, ma solo nel caso possieda il permesso.

- Servizio: un servizio è un componente eseguito in background che si occupa di operazioni lunghe o processi remoti. Un servizio non fornisce l'interfaccia utente, ma può ricavare dati importanti senza bloccare l'interazione dell'utente con l'activity.
- Fornitori di contenuto: questo componente si occupa di gestire i dati dell'applicazione. Può memorizzare informazioni nel file system, in un database SQLite, sul Web o su qualsiasi locazione di memoria a cui il software ha accesso. Attraverso il fornitore di contenuto, altre applicazioni possono richiedere o anche modificare i dati se hanno ottenuto il permesso. Ad esempio, all'interno del codice di Android è presente un fornitore di contenuto che gestisce le informazioni dei contatti dell'utente, perciò ogni applicazione con le giuste credenziali può richiedere quei dati.
- Broadcast receiver (dall'inglese ricevitore di trasmissioni): questo oggetto risponde a comunicazioni tramesse a livello di sistema. Molte trasmissioni hanno origine dal sistema principale, ad esempio si potrebbe voler comunicare che lo schermo del dispositivo è stato disattivato o che la batteria è quasi scarica. Anche le applicazioni possono effettuare delle trasmissioni, come avvertire che alcuni dati sono stati scaricati e ora accessibili all'uso. Sebbene i broadcast receiver non visualizzino un'interfaccia utente, possono creare notifiche nella barra di stato per avvertire l'utente di un particolare evento. Comunemente, questo componente esegue poca quantità di lavoro e si occupa principalmente di lanciare altre azioni.

Uno degli aspetti unici di Android è che ogni applicazione può lanciare componenti di un'altra applicazione, non c'è bisogno di riscrivere né di collegare del codice, basta semplicemente far partire l'activity che esegue la funzione necessaria allo scopo. Si consideri un'applicazione che oltre alle sue normali attività, voglia anche accedere alla fotocamera del dispositivo. Per fare ciò, si vuole attivare il componente di un'altra applicazione apposita alla gestione della fotocamera, specificando al sistema la sua intenzione. Dal punto di vista dell'utente, questa catena di eventi verrà visualizzata senza interruzioni e sembrerà che la fotocamera venga gestita dall' applicazione originale.
L'intenzione di eseguire un secondo componente in Android viene esplicata tramite la classe Intent, che definisce un messaggio per attivare la funzione che si desidera.

```
Intent intent = new Intent(this,DisplayMessageActivity.class);
startActivity(intent);
```

Figura 2.1 Esempio di lancio di una nuova activity

In questa porzione di codice (Figura 2.1), viene creata un'istanza della classe di Intent per lanciare una seconda activity.

2.1.5.3 Manifest

Prima che Android possa far partire l'applicazione, il sistema dovrà essere a conoscenza di tutti i suoi componenti. A questo scopo, esiste un file apposito, il Manifest, dove vengono dichiarate le informazioni più importanti dell'applicazioni:

- I componenti dell'applicazione
- Identificare tutti i permessi necessari, come l'accesso ad Internet o di poter leggere o scrivere i contatti dell'utente.
- Dichiarare il livello API minimo richiesto dal software, basato sulla versione che viene usata.
- Dichiarare caratteristiche hardware o software usate o richieste dall'applicazione, come fotocamera, servizi bluetooth o schermi multitouch.

Il Manifest prende la forma di un file XML presente nella cartella principale del progetto, vediamo un esempio di una dichiarazione del componente activity:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
    <application android:icon="@drawable/app_icon.png ... >
        <activity
android:name="com.example.project.ExampleActivity"
                android:label="@string/example_label"... >
        </activity>
        ...
    </application>
</manifest>
```

Figura 2.2 Esempio di Manifest di un'applicazione Android

Nell'elemento `<application>` viene definita l'icona che identifica l'applicazione, mentre in `<activity>` il campo name specifica il nome completo dell'attività.

2.1.5.4 Risorse

Un'applicazione Android è composta da molto di più che solo il codice, necessita infatti di risorse separate dal codice sorgente, come immagini, file audio e qualsiasi altro contenuto relativo alla presentazione visiva del software. Per definire stili, menu, animazioni, colori e la disposizione degli elementi dell'interfaccia utente, il layout di un'activity, vengono utilizzati file XML. L'utilizzo di queste risorse rende facile l'aggiornamento di alcune caratteristiche del prodotto senza modificare il codice e permette di ottimizzarlo per varie configurazioni. Per ogni risorsa inclusa nel progetto, viene definito un identificatore univoco usato per referenziarla nel codice del software o in altri file XML.

Uno degli aspetti più importanti di questo sistema è la possibilità di fornire risorse diverse a seconda della configurazione del dispositivo. Ad esempio, definendo le stringhe per l'interfaccia utente in XML queste possono essere tradotte e salvate in file diversi. Con un'opportuna suddivisione di documenti, Android applicherà automaticamente il testo giusto a seconda della lingua del dispositivo.

2.2 Google Fusion Table

Fusion Table è un servizio gratuito basato su cloud per la gestione e l'integrazione di dati, lanciato nel giugno 2009 da Google. Questo servizio permette agli utenti di caricare informazioni sotto forma di diverse sorgenti, come fogli elettronici, file CSV o KML, con la possibilità di visualizzarle facilmente con grafici o mappe, a seconda del tipo di dati presenti. Le Fusion Table forniscono anche una semplice piattaforma per l'integrazione dei dati, infatti gli utenti possono eseguire unioni con le proprie tabelle o con quelle di altri. Per collaborare, gli utilizzatori possono condividere i loro dati con un certo numero di persone, oppure rendere pubbliche le proprie tabelle in modo che siano utilizzabili da chiunque e ottenibili tramite motori di ricerca. Inoltre è presente una funzione apposita per la discussione riguardo a specifiche celle, righe o colonne di una tabella per facilitare la comunicazione fra collaboratori.

Il servizio Fusion Table è stato creato con lo scopo di semplificare la gestione di dati e renderla accessibile ad un pubblico più vasto di utenti, anche a coloro che non sono pratici nell'utilizzare database. Inoltre è stato notato che uno dei maggiori impedimenti all'integrazione e diffusione dei dati è la mancanza di incentivi alla condivisione. Sono stati infatti progettati diversi meccanismi per facilitare e invogliare gli utenti a condividere le proprie informazioni, ovviamente quando attuabile. I progettisti di Fusion Table si sono concentrati sulle funzionalità che permettano agli utenti di collaborare efficacemente sulla gestione di dati conservati su cloud.

2.2.1 Architettura

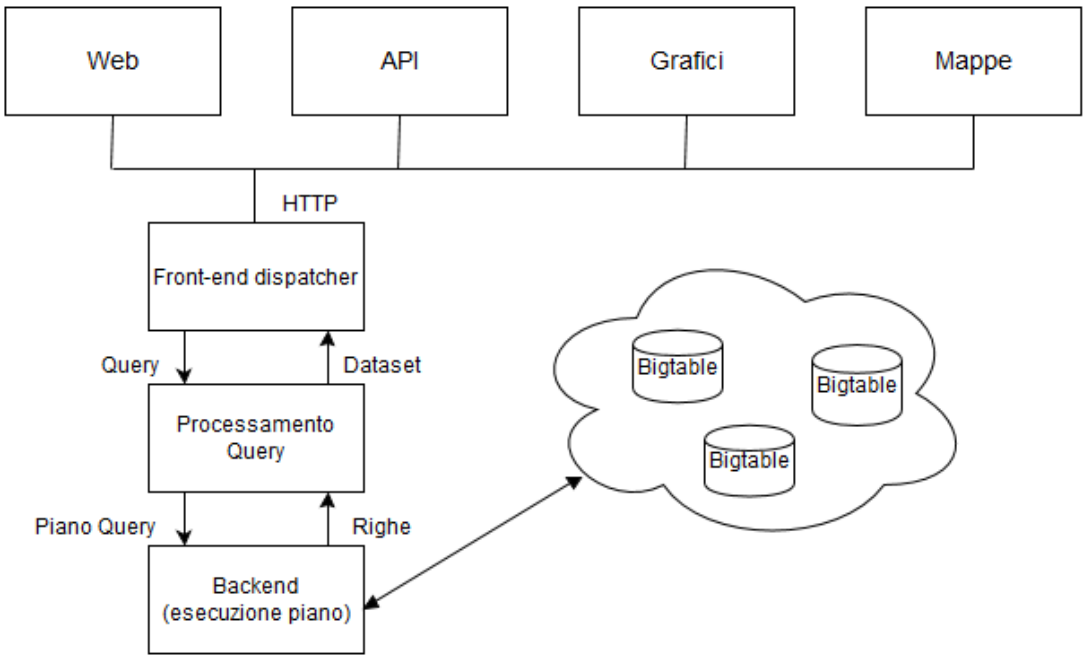

Figura 2.3 Architettura del servizio Fusion Table [GON10]

La Figura 2.3 mostra i componenti principali del servizio Fusion Table. Le richieste vengono generate da diverse origini: dal sito principale di Fusion Table, dalle applicazioni che utilizzano l'API (Application Programming Interface) per la gestione dei loro dati, oppure da visualizzazioni incorporate da altri servizi Web, come mappe o grafici. Il front end dispatcher converte le richieste in una rappresentazione comune e le passa al modulo che si occupa di processare le query, creando un piano per loro gestione. Il piano viene eseguito dal backend che usa un insieme di server Bigtable replicati in sincrono per gestire lo spazio di memoria.

2.2.2 Visualizzazione dei dati

Il servizio prevede un semplice client utilizzabile tramite browser Web con cui l'utente può manipolare facilmente le proprie tabelle. I dati possono essere visualizzati sotto forma di tabella oppure con delle "carte", un layout più espanso in cui ogni riga viene rappresentata come un riquadro avente all'interno le proprie informazioni.

Una delle componenti fondamentali del servizio è la possibilità di visualizzare istantaneamente i dati collegati ad una mappa geografica, se nella tabella è presente un campo apposito per la geo localizzazione. Si può perciò trasformare la tabella in una mappa, che si basa sempre sul servizio Google Maps, rappresentando i propri elementi sotto forma di punti, linee e poligoni, ad esempio per percorsi o aree di interesse. Oltre alla mappa, esiste anche la visualizzazione sotto forma di grafici di qualsiasi tipo, come istogrammi, grafi o cartesiani.

2.2.3 Condivisione dei dati e collaborazione

Oltre ad avere la possibilità di scaricare su hard disk l'intera tabella in vari formati come CSV e KML, l'utente può scegliere se condividere il collegamento relativo alla Fusion Table con altre persone. Una tabella può avere tre stati di condivisione: pubblica, ovvero che non necessita richiesta di autenticazione e aperta a chiunque; accessibile a chi possiede il link; condivisa in privato, perciò accessibile solo a specifici utenti che hanno ricevuto l'autorizzazione dal creatore della tabella. Per avere accesso alle Fusion Table è ovviamente necessario avere un account Google.

2.2.4 Fusion Table API

L'API di Fusion Table [FT14] è composta da un insieme di concetti principali:

- Tabelle: consistono nell'organizzazione dei dati in righe e colonne, simile ad un foglio di calcolo. Ogni tabella è identificata da un codice univoco.
- Colonne: ogni colonna ha un identificatore, un nome e un tipo di dati assegnato come "STRING", "NUMBER" o "LOCATION".
- Righe: i dati vengono organizzati in righe, ognuna con un codice univoco chiamato ROWID.

La Fusion Table API permette di usare richieste HTTP per eseguire le seguenti azioni:

- Creare o cancellare tabelle
- Leggere o modificare informazioni su una tabella, come nomi o tipi delle colonne
- Inserire, aggiornare e cancellare righe da una tabella
- Creare, aggiornare e cancellare impostazioni su certi tipi di visualizzazione di dati
- Eseguire richieste sulla tabella

È stata progettata con un'architettura di tipo REST (Representational State Transfer) che consiste nell'utilizzare il protocollo HTTP per ricevere o modificare dati memorizzati da Google. In un sistema REST, le risorse sono mantenute su un server, il quale le modifica internamente in base alla richiesta ricevuta e poi invia la risposta con il contenuto desiderato all'applicazione.

Per eseguire le richieste, viene utilizzato un indirizzo composto nel seguente modo:

```
https://www.googleapis.com/[nome API]/[versione dell'API]/[tipo
di risorsa]?[parametri]
```

Figura 2.4 URL per interagire con le Google API

In particolare, per interrogare le righe, viene impiegato un sottoinsieme dell'SQL, un linguaggio di programmazione progettato appositamente per gestire database. Ad esempio, per selezionare tutte le righe di una tabella si può creare l'indirizzo di richiesta come nell'esempio seguente:

```
https://www.googleapis.com/fusiontables/v1/query?sql=
"SELECT ROWID FROM [ID Tabella]"
```

Figura 2.5 Esempio di richiesta ad una Fusion Table

2.3 Foursquare

Foursquare [FSQ14] è un social network basato sulla geo localizzazione disponibile via sito web e come applicazione per dispositivi mobile, che permette agli utenti registrati di segnalare la loro posizione, azione soprannominata come check-in in questo contesto, interagendo con i propri amici.

Il servizio Foursquare è nato nel 2009 dagli sviluppatori Dennis Crowley e Naveen Selvandurai, ispirati dalle nuove possibilità derivanti dall’espansione del mercato degli smartphone. Precedentemente Crowley ha lavorato ad un progetto simile chiamato Dodgeball, che permetteva di segnalare agli altri utenti della zona la propria locazione tramite SMS. Questo servizio è stato acquisito da Google nel 2005 e successivamente chiuso nel 2009 per essere sostituito da Google Latitude. Foursquare prende l’idea di base di Dodgeball e la evolve, dando la possibilità di poter interagire con l’ambiente tramite i dispositivi mobile.

Al momento, oltre 45 milioni di utenti sono registrati al social network e sono stati eseguiti più di 5 miliardi di check-in, con una media 3 milioni di check-in al giorno.

2.3.1 Funzionamento

L’azione principale che l’utente compie su Foursquare è il check-in, a ciascuno dei quali corrisponde l’assegnazione di un certo punteggio. Gli utenti vengono incoraggiati ad essere il più specifici possibili con i loro check-in e possono scegliere di condividerli anche su altri social network, come Facebook o Twitter. Un’altra delle caratteristiche principali sono le medaglie che vengono assegnate agli utenti in seguito ad un certo numero e tipo di azioni, come visitare lo stesso posto più volte nello stesso mese, oppure fare check-in in luoghi con caratteristiche particolari. Gli sviluppatori aggiungono periodicamente nuove medaglie al servizio per invogliare le persone a continuare ad utilizzare Foursquare.

È anche possibile creare delle note o dei consigli riguardo ai luoghi frequentati, ad esempio un utente dopo aver fatto un check-in in un ristorante può condividere la propria opinione riguardo la sua esperienza del locale. Di conseguenza il servizio ha le caratteristiche di un social network ma anche quelle di una piattaforma in cui possono essere visualizzate le recensioni degli utenti.

Esistono anche il “sindaco” di un locale, cioè l’utente che ha eseguito più check-in di tutti gli altri nell’arco di 60 giorni in quel dato luogo. Questo porta ad una competizione fra le persone registrate per conquistare il titolo di sindaco e quindi ad un utilizzo maggiore del servizio.

2.3.2 Foursquare API

L'API di Foursquare [DFSQ14] permette agli sviluppatori di applicazioni o servizi di interagire con la piattaforma del social network. Come nel caso di Fusion Table, anche questa API si basa su un'architettura di tipo REST, utilizzando quindi richieste HTTP per ricevere o inviare informazioni. Lo sviluppatore ha la possibilità di scegliere se ricevere i dati in formato XML o JSON.

I tipi di richieste definite nell'API sono divise in numerose categorie, fra cui le più importanti sono:

- Users: restituiscono informazioni sull'utente specificato, come il nome o le sue medaglie.
- Venues: utilizzato per ricevere o inviare informazioni su un luogo.
- Checkins: inviano dettagli su un check e offre la possibilità di effettuare un check-in, se viene specificato l'utente e il luogo.
- Tips: restituisce dati su un tip, cioè un consiglio che gli utenti lasciano su un certo locale.

2.4 Google Maps

Google Maps [GM14] è un servizio adibito alla visualizzazione di mappe satellitari fornito e sviluppato da Google su cui si basano molte applicazioni basate su mappe. Oltre alla sua funzionalità di base, offre anche un pianificatore di percorsi per viaggi a piedi, in bicicletta, in auto o tramite mezzi pubblici. Inoltre include la possibilità di localizzare luoghi particolari come attività locali o monumenti nella maggior parte dei Paesi del pianeta.

Sebbene le immagini satellitari del servizio non vengono aggiornate in tempo reale, Google aggiunge regolarmente informazioni al suo database primario. In ogni caso, nessuna delle immagini è più vecchia di tre anni.

Nel giugno 2005, è stata resa pubblica l'API di Google Maps, in modo da permettere agli sviluppatori di integrare il servizio con i loro siti o applicazioni. Attraverso quest'interfaccia, è possibile arricchire il proprio prodotto, aggiungendo alla mappa uno strato superiore per visualizzare dati esterni, come marcatori o linee. La Google Maps API è gratuita anche per uso commerciale, a patto che l'applicazione sia pubblicamente accessibile e non richieda costi aggiuntivi per l'utilizzo, inoltre non deve

generare più di 25000 accessi al giorno. In caso il servizio non soddisfi questi requisiti, lo sviluppatore dovrà acquistare un abbonamento apposito a Google.

2.5 Tecnologie intermedie

Si definiscono ora le tecnologie intermedie utilizzate per sfruttare al meglio dei servizi utilizzati.

2.5.1 Java

Java [JAV14] è un linguaggio di programmazione creato da James Gosling di Sun Microsystem negli anni novanta, progettato appositamente per avere il minor numero di dipendenze possibili e quindi essere compatibile con molte piattaforme diverse.

Alla sua nascita, Java ha avuto un grande successo nella comunità mondiale degli sviluppatori di software e servizi per Internet, visto che permetteva agli utenti di utilizzare applicazioni sicure, indipendentemente dalla piattaforma usata. Infatti, una delle caratteristiche principali che differenziano Java dagli altri linguaggi di programmazione come C e C++ è il sistema con cui vengono compilati ed eseguiti i programmi. Il sorgente non viene tradotto direttamente in linguaggio macchina ma convertito in un codice chiamato Bytecode. Il codice generato viene poi eseguito tramite la JVM (Java Virtual Machine), una macchina virtuale che interpreta il Bytecode ed esegue l'applicazione sul computer su cui è installato. Grazie a questa macchina virtuale che crea uno strato intermedio fra il Bytecode e il sistema operativo, Java diventa un linguaggio indipendente dalla piattaforma, permettendo di eseguire lo stesso software scritto in Java su computer con sistemi operativi diversi.

Sebbene la compatibilità di Java con molti sistemi differenti sia un grande beneficio, il suo funzionamento porta anche a diversi svantaggi. Infatti, l'interpretazione del codice, infatti, porta sempre ad un'esecuzione più lenta rispetto a programmi compilati in eseguibili nativi. Questo problema è stato risolto introducendo dei compilatori JIT (Just-In-Time, tradotto appena in tempo dall'inglese), che consistono in una combinazione dei due metodi classici di compilazione, l'interpretazione e la traduzione in linguaggio macchina prima dell'esecuzione.

La sintassi di Java è derivata da C++, ma si differenzia da quest'ultimo per la sua progettazione principalmente orientata agli oggetti. Tutto il codice viene scritto in una

classe e tutto è un oggetto, ad eccezione dei tipi di dati primitivi come interi o valori booleani.

2.5.2 OAuth 2.0

OAuth, Open Authorization, è un protocollo di autenticazione che permette le applicazioni di accedere l'una ai dati dell'altra. Descrive diverse modalità in cui un'applicazione client può ottenere l'autorizzazione di accedere alle risorse presenti in un server a nome dell'utente.

La prima versione del protocollo è nata nel novembre del 2006, creata da Blaine Cook mentre stava sviluppando un'implementazione di OpenID per Twitter. OAuth 2.0 è l'evoluzione di questa versione iniziale, che si focalizza principalmente sulla semplicità dello sviluppo, fornendo specifiche modalità di autorizzazione per applicazioni web, desktop e mobile. Al momento molti social network e compagnie utilizzano OAuth 2.0 per la gestione dell'autorizzazione delle loro risorse, come Google, Facebook e Foursquare.

2.5.2.1 Flusso di autorizzazione

Il protocollo definisce quattro diversi ruoli:

- Il Resource Owner, la persona o l'applicazione a cui appartengono le informazioni che vanno condivise. Ad esempio un utente su Facebook o Google può essere un Resource Owner.
- Il Resource Server dove risiedono i dati.
- Client Application, cioè l'applicazione che richiede accesso alle risorse contenute nel Resource Server. Un gioco che richiede l'accesso all'account di Facebook dell'utente è una Client Application.
- Authorization Server che si occupa di autorizzare la Client Application ad avere accesso alle risorse del Resource Owner. Il Resource Server e l'Authorization Server possono essere contenuti nello stesso server o risiedere in istanze separate.

Prima che una Client Application possa richiedere l'autorizzazione alle risorse, dovrà prima essere registrata con l'Authorization Server associato al Resource Server. Solitamente è un'operazione che si compie una volta sola e rimane valida finché non

viene revocata. Alla fine della registrazione, alla Client Application vengono assegnati un Client ID e un Client Secret univoci dall’Authorization Server. Se l’applicazione viene registrata su server di servizi diversi, ogni Authorization Server assegnerà un proprio Client ID unico. Durante questo processo, deve essere indicato anche un indirizzo Web (URI) per il redirect. Quando l’autorizzazione viene concessa, la Client Applicazione viene reindirizzata automaticamente all’URI stabilito.

Il flusso di autorizzazione può essere descritto con questi passaggi:

- Il Resource Owner accede alla Client Application, la quale dice all’utente di eseguire il login attraverso un Authorization Server.
- Per compiere il login con il proprio nome e password, l’utente viene reindirizzato al server di autenticazione. L’applicazione invia anche il Client ID in modo che il server sappia quale client sta cercando di accedere alle risorse.
- L’utente esegue il login, a cui in seguito viene chiesto se vuole autorizzare alla Client Application l’accesso alle risorse protette. Se viene riscontrato un esito positivo, l’utente tornerà all’applicazione.
- Dopo il login, l’Authorization Server rendirizza l’utente all’URI specificato durante la registrazione insieme al codice di autorizzazione.
- Una volta che si ha accesso all’URI di redirect, l’applicazione sarà direttamente connessa al server, a cui manda il codice di autorizzazione insieme al Client ID e al Client Secret. Se questi valori vengono accettati, restituirà un token di accesso.
- Ora la Client Application può usare il token di accesso per richiedere le risorse dal server. Il token serve sia come autenticazione del client e per l’autorizzazione di accesso alle risorse.

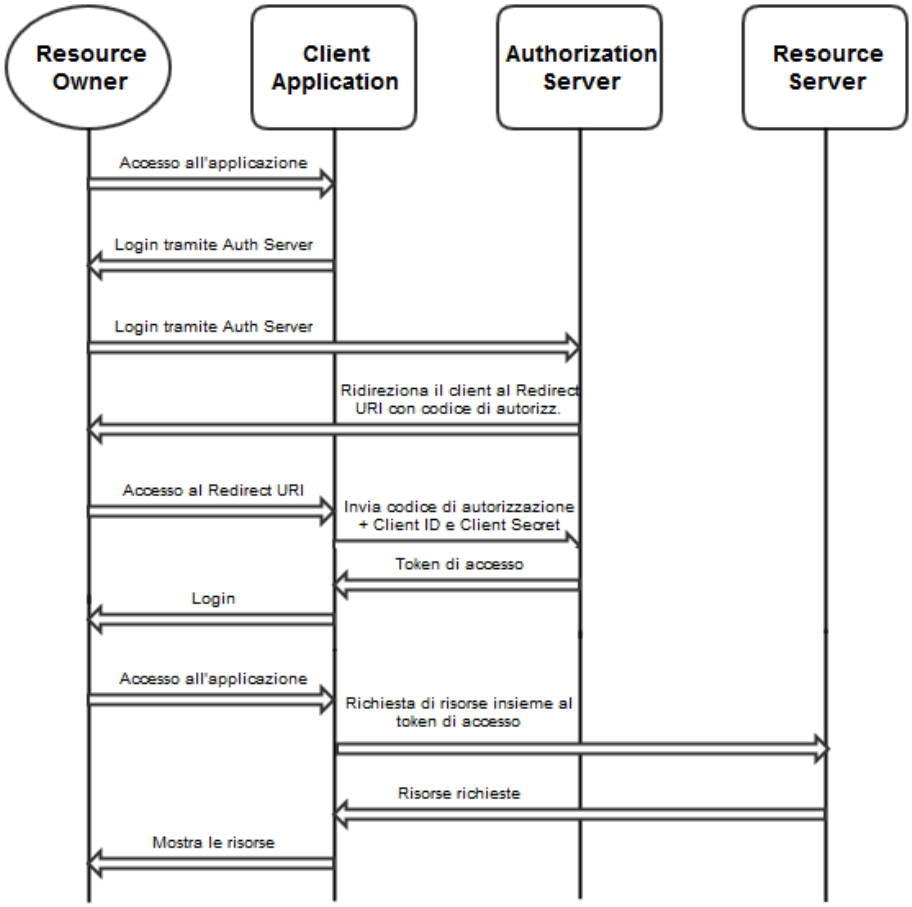

Figura 2.6 Diagramma del flusso di autorizzazione descritto dal protocollo OAuth 2.0 [JEN14]

2.5.3 HTTP

HTTP (HyperText Transfer Protocol) [HTTP14] è un protocollo usato come principale sistema per la trasmissione di informazioni sul Web, quindi basata sul concetto di client e server. La prima versione dell'HTTP risale alla fine degli anni ottanta e costituiva, insieme al linguaggio HTML (HyperText Markup Language) e URI (Uniform Resource Identifier), il nucleo base del World Wide Web sviluppata da Tim Berners-Lee al CERN di Ginevra per la condivisione delle informazioni tra le comunità dei fisici delle alte energie. La prima versione effettivamente disponibile del protocollo, la HTTP/1.0, venne implementata dallo stesso Berners-Lee nel 1991.

L'HTTP funziona come un protocollo di richiesta-risposta in un modello di computazione client-server. Ad esempio, un browser Web come Google Chrome può essere il client e un computer che ospita un sito può essere un server. Il client invita un messaggio di richiesta HTTP al server. Il server, che restituisce file HTML o altri

contenuti o esegue altri compiti a nome del client, restituisce un messaggio di risposta, contenente informazioni riguardo allo stato della richiesta o anche altro.

HTTP definisce particolari metodi per indicare l'azione desiderata che deve essere eseguita sulla risorsa indicata. Cosa rappresenti la risorsa, sia un dato già esistente o creato dinamicamente, dipende dall'implementazione del server. Solitamente corrisponde ad un file o il risultato di un eseguibile residente nel server. I metodi più utilizzati, che sono stati definiti inizialmente nella prima versione del protocollo sono:

- GET: richiede una rappresentazione di una specifica risorsa. Le richieste di tipo GET dovrebbero solo ricavare dati e non avere nessun'altro effetto.
- POST: questo metodo è utilizzato per domandare al server di accettare un'entità racchiusa all'interno del messaggio della richiesta come nuova risorsa. Un esempio di tipo di informazione che potrebbe essere inviato è un messaggio in un gruppo di discussione o un commento su una notizia.
- HEAD: esegue una richiesta identica ad una corrispondente al metodo GET ma senza il messaggio di risposta, perciò non chiede una particolare risorsa ma solo informazioni riguardo ad essa, come la data di ultima modifica.

2.5.4 JSON

JSON (JavaScript Object Notation) [JSON14] è uno formato standard che utilizza un testo codificato in modo da essere facilmente comprensibile per trasmettere dati composti da un insieme di coppie attributo-valore. È usato principalmente per trasmettere informazioni tra un server e un'applicazione client, come alternativa al XML.

Sebbene originariamente sia derivato dal JavaScript, JSON è indipendente da qualsiasi linguaggio e librerie per analizzare il formato sono disponibili per una grande varietà di linguaggi di programmazione.

Il formato JSON è stato creato da Douglas Crockford, nato dal bisogno di un tipo di comunicazione in tempo reale fra server e browser slegata da plugin come Flash o applet Java che dominavano la filosofia di progettazione del tempo.

I tipi principali in JSON sono:

- Numeri
- Stringhe
- Booleani (vero o falso)

- Array
- Oggetti (una collezione di coppie attributo-valore)
- Null (vuoto)

Il contenuto di un dato in formato JSON è racchiuso da una coppia di parentesi graffe. In seguito, vengono definiti tutti i valori assegnati ai vari attributi, i quali devono essere sempre racchiusi fra doppie virgolette. Un esempio di codice JSON di una rappresentazione delle informazioni riguardanti una persona qualsiasi può essere questo:

```
{
    "nome": "Mario",
    "cognome": "Rossi",
    "età": 25,
    "indirizzo": {
        "nominativo": "Via Ampere 15",
        "città": "Milano",
        "codicePostale": 20131
    },
    "numeroTelefono": [
        {
            "tipo": "casa",
            "numero": "051316412"
        },
        {
            "tipo": "cellulare",
            "numero": "3483543205"
        }
    ]
}
```

Figura 2.7 Esempio di dati in formato XML

Dall'esempio diventa chiaro il concetto di coppia attributo-valore: l'attributo "nome" ha come valore la stringa "Mario", così come il campo "cognome". Inoltre, si può notare che "indirizzo" sia definito come oggetto, infatti è composto da tre valori distinti, mentre "numeroTelefono" è un array di oggetti.

3 Implementazione

In questo capitolo verrà presentata nel dettaglio l'implementazione dell'applicazione Android. Questa comunica con una Fusion Table, dove vengono mantenuti tutti i dati sulle segnalazioni relative all'accessibilità dei luoghi di interesse. Permette inoltre di sfruttare il servizio Foursquare per facilitare la ricerca dei luoghi d'interesse e la condivisione delle informazioni. Le altre funzionalità offerte dal sistema sono: visualizzare tutti i dati raccolti, compilare questionari sull'accessibilità dei luoghi, creare un suo profilo, inserire le proprie preferenze e personalizzare i contenuti

Inizialmente verranno quindi presentate le classi principali sviluppate per implementare le funzionalità di base dell'applicazione e comunicare con i servizi in rete. In seguito sarà mostrato in modo dettagliato il funzionamento e l'aspetto dell'applicazione, sin dal primo avvio.

3.1 Classi principali

Verranno ora descritte alcune classi, implementate nel codice dell'applicazione, che si occupano di gestire le comunicazioni con i vari servizi utilizzati e di memorizzare informazioni fondamentali.

3.1.1 Implementazione Fusion Table

Di seguito vengono definiti tutti i metodi necessari ad interagire con la Fusion Table implementati in una classe creata appositamente e chiamata FTClient.

3.1.1.1 Autorizzazione

Solitamente il primo compito eseguito da questa classe, al momento della sua inizializzazione, è quello di autorizzare l'applicazione ad utilizzare i servizi di Google con il protocollo OAuth e quindi ottenere il token di accesso. Questo è possibile attraverso la classe AccountManager, presente nell'API di Android, che semplifica tale processo utilizzando l'account Google dell'utente per compiere i primi passi del protocollo.

```
private void requestToken() {
   AccountManager accountManager=AccountManager.get(context);
   ..
   //Ricava i dati dell'account dell'utente
   ..
   accountManager.getAuthToken(userAccount, "oauth2:" + SCOPE,
null, (Activity)context,new OnTokenAcquired(), null);
}
```

Figura 3.1 Metodo per la richiesta del token di accesso

In questa porzione di codice (Figura 3.1) possiamo notare come venga creato l'oggetto `accountManager`, che tramite il metodo `.getAuthToken()` ricava il token di autorizzazione. A tale metodo è necessario fornire in particolare l'account di Google dell'utente, il servizio d'interesse (in questo caso FusionTable) e la funzione di callback, che verrà lanciata al termine dell'operazione di autorizzazione. Se il token è stato ottenuto senza errori, viene memorizzato nei file locali dell'applicazione, in modo da poterlo riutilizzare successivamente per le richieste alla tabella. In caso contrario, viene mostrato un messaggio di errore all'utente.

3.1.1.2 Richieste

Per eseguire le richieste viene definito un metodo pubblico, che crea un nuovo processo per non bloccare quello principale dell'interfaccia:

```
public void queryOnNewThread(final String code)
{
      new Thread()
      {
            @Override
            public void run()
           {
                query(code);
           }
      }.start();
}
```

Figura 3.2 Metodo di creazione nuovo thread per le richieste alla Fusion Table

La funzione `queryOnNewThread()`, che riceve in ingresso la query in SQL da eseguire sulla tabella, crea il nuovo processo ed esegue il metodo `query()` che si occupa di compiere la richiesta HTTP. A seconda del tipo di query SQL, viene usato un diverso metodo HTTP: se si vuole solo ricavare dei dati specifici tramite SELECT si userà un metodo GET, altrimenti un metodo POST per aggiornare o modificare la tabella. Si presenterà ora un esempio di richiesta HTTP con metodo GET con il seguente codice.

```
url=query_url+instruction+"&access_token="
+authPreferences.getToken().toString();
HttpClient client=new DefaultHttpClient();
HttpGet get=new HttpGet(url);
HttpResponse resp=client.execute(get);
HttpEntity httpEntity = resp.getEntity();
String output=EntityUtils.toString(httpEntity);
```

Figura 3.3 Codice per una richiesta HTTP

Inizialmente viene composto l'indirizzo a cui fare la richiesta, specificando l'istruzione SQL e il token di accesso fornito precedentemente. In seguito si istanzia il client che si occuperà di inviare la richiesta al server, la cui risposta in formato JSON viene memorizzata in un oggetto di tipo stringa.

Infine la risposta viene mandata all'activity, che l'ha richiesta originariamente, che potrà manipolare tale oggetto per ottenere i dati di cui ha bisogno attraverso i metodi per la gestione del JSON presenti nell'API di Android.

3.1.2 Implementazione Foursquare

FoursquareApp è la classe principale dedita alla comunicazione con Foursquare, offrendo metodi per il login dell'utente al social network, per ricavare i dati vicini ad una certa posizione geografica e per eseguire check-in.

3.1.2.1 Login a Foursquare e autorizzazione

Per permettere ad un'applicazione di interfacciarsi al social network è necessario prima di tutto registrarla al servizio (attraverso la pagina Web per sviluppatori Foursquare), come descritto nel protocollo OAuth 2.0. Dalla registrazione si ottengono due valori chiamati Client ID e Client Secret, che nel progetto vengono memorizzati in

una classe statica chiamata Constants, in modo che siano sempre disponibili in qualsiasi parte del codice.

Il metodo `authorize()` della classe mostra una finestra Web, dove l'utente può eseguire il login con il suo account di Foursquare, inserendo e-mail e password, compiendo così il primo passo dell'autorizzazione. Se il login ha successo, viene restituito il codice da utilizzare per ottenere il token di accesso, con il seguente metodo:

```
private void getAccessToken(final String code)  {
. . .
     URL url = new URL(mTokenUrl + "&code=" + code);
     HttpURLConnection urlConnection =
(HttpURLConnection)url.openConnection();
     urlConnection.setRequestMethod("GET");
     urlConnection.connect();
     JSONObject jsonObj  = (JSONObject) new
JSONTokener(streamToString(urlConnection.getInputStream())).next
Value();
     mAccessToken = jsonObj.getString("access_token");
. . .
}
```

Figura 3.4 Funzione per ottenere il token di accesso per Foursquare

In questo stralcio di codice (Figura 3.4), viene eseguita una richiesta HTTP di tipo GET, specificando nell'indirizzo il Client ID, il Client Secret e il codice ottenuto precedentemente. Una volta ricavato il token di accesso dalla risposta in formato JSON, sarà possibile richiedere informazioni a Foursquare.

3.1.2.2 Acquisizione dei luoghi vicini

Se si vogliono ottenere sulla schermata i luoghi vicini all'utente è necessario innanzitutto ottenere la sua posizione. Grazie al GPS del dispositivo, è infatti possibile ricavare la sua locazione geografica facilmente. A tale scopo è stata definita una funzione apposita all'interno della classe FoursquareApp, che ricevendo in ingresso la latitudine e longitudine dell'utente restituisce i trenta locali più vicini registrati da Foursquare.

Per ottenere informazioni sui luoghi è necessario effettuare una richiesta HTTP con metodo GET all'indirizzo: `https://api.foursquare.com/v2/venues/`
I parametri da specificare sono: la locazione geografica, il token di accesso, la data corrente ed eventualmente una chiave di ricerca. In caso la richiesta vada a buon fine, i dati vengono restituiti in formato JSON.

Il codice JSON viene analizzato per ricavare le informazioni richieste, le quali vengono memorizzate in una lista di oggetti FsqVenue, una classe rappresentante un luogo generico presente nel database di Foursquare.

```
public class FsqVenue
{
      private String id;
      private String name;
      private String address;
      private String type;
      private Double latitude;
      private Double longitude;
      private String distance;
      …
}
```

Figura 3.5 Definizione della classe FsqVenue

In questa porzione di codice (Figura 3.5) viene definita la classe FsqVenue, con gli attributi per l'identificativo, il nome, l'indirizzo, il tipo (monumento, ristorante, etc…), la latitudine, la longitudine e la distanza dalla posizione dell'utente.

3.1.2.3 Check-in

L'implementazione del check-in è simile alla ricerca dei locali vicini, ma vi sono due differenze principali. La prima è l'indirizzo Web: in questo caso `https://api.foursquare.com/v2/checkins/add`; la seconda è il metodo impiegato, ovvero il POST dato che deve inviare dati a Foursquare.

Nei parametri è necessario specificare l'identificatore del luogo in cui si vuole eseguire il check-in, oltre alla data e al token di accesso.

3.1.3 Implementazione profilo utente

All'interno dell'applicazione il profilo utente è definito da un nominativo, la sua immagine di profilo, il numero di segnalazioni che ha eseguito e le sue preferenze. Inoltre possiede un registro, dove vengono memorizzate tutte le azioni che ha compiuto dalla sua registrazione.

Per implementare il profilo è stata creata una classe chiamata User, che ne gestisce tutte le informazioni. Per poter registrare queste ultime è stata impiegata la classe SharedPreferences, presente nell'API di Android, la quale si occupa di registrare dati in un file locale dell'applicazione in modo che siano utilizzabili in sessioni diverse. Inoltre SharedPreferences assegna ad ogni informazione una chiave in modo che sia possibile ottenerle facilmente.

La seguente porzione di codice descrive la creazione della classe, nella quale si ottengono le SharedPreferences relative all'utente in modalità privata, affinché non siano accessibili da altre applicazioni.

```
public User (Context _context)
{
           context=_context;
           pref=context.getSharedPreferences("user",
Context.MODE_PRIVATE);
}
```

Figura 3.6 Costruttore della classe User

All'interno di User sono definiti i metodi per ottenere o modificare gli attributi del profilo, questo è possibile utilizzando sempre le SharedPreferences, come nel seguente esempio per il nome.

```
public String getName(){
     return pref.getString(name, null);
}
public void setName(String _name){
     Editor editor = pref.edit();
     editor.putString(name, _name);
     editor.commit();
}
```

Figura 3.7 Funzioni per leggere o scrivere il campo nome in User

Nel primo caso viene semplicemente restituito il valore richiesto; nel secondo metodo adibito alla modifica del campo nome invece è necessario istanziare un oggetto Editor per modificare le preferenze, questo si occuperà di inserire il nuovo valore ottenuto in input.

3.1.3.1 Registro delle azioni

Insieme al profilo, è stato implementato un database utilizzato per mantenere uno storico di tutte le azioni che ha eseguito l'utente sin dal momento della registrazione, come ad esempio le segnalazioni da lui inviate. Il database ha la seguente struttura:

- un numero intero come identificativo dell'azione,
- data in cui è stata compiuta,
- una stringa dove viene descritta l'azione compiuta,

Per implementare questo registro, è stata creata una classe chiamata LogDbManager che sfrutta il servizio SQLite [SQL14] presente all'interno di Android. LogDbManager contiene i metodi per aprire e chiudere il database alla lettura e alla scrittura, ottenere i dati al suo interno e inserirne dei nuovi.

3.2 Avvio applicazione e creazione profilo

Il primo compito svolto dall'applicazione al momento dell'avvio è quello di verificare la possibilità o meno del dispositivo di accedere ad una connessione Internet. In caso positivo viene richiesto il token di accesso alla Fusion Table, tramite la classe FTClient, dato che tale valore è necessario nella schermata successiva per richiedere le segnalazioni da mostrare sulla mappa. Se non è presente nessuna connessione Internet, viene mostrato all'utente un messaggio di errore, dopodiché l'applicazione si chiude automaticamente. Questo controllo è stato implementato con il seguente codice presente nel metodo di creazione dell'activity.

```
if (checkConnection()) {
      FTClient ft=new FTClient(context);
      ft.authorize();
}
else {     ... Visualizzazione finestra di errore ...
}
```

Figura 3.8 Codice per controllare la presenza di connessione a Internet

In seguito al successo dell'autorizzazione con la Fusion Table, al primo avvio dell'applicazione da parte dell'utente, viene mostrata la schermata di benvenuto, dove si può scegliere la modalità di creazione del profilo.

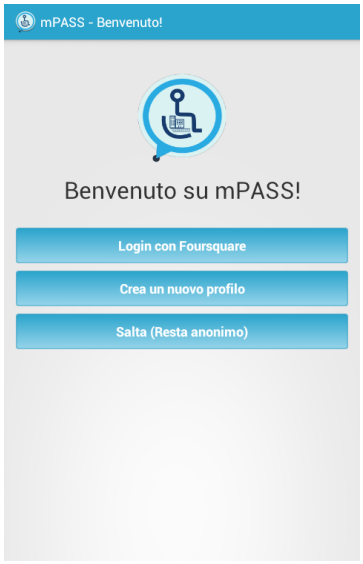

Figura 3.9 Schermata di benvenuto dell'applicazione

Nella Figura 3.9 si può osservare la schermata di benvenuto, dove l'utente può stabilire se eseguire il login con Foursquare, creare manualmente il proprio profilo o saltare la registrazione e rimanere perciò anonimo. Descriviamo dunque il comportamento dell'applicazione nei tre casi:

- Login con Foursquare:

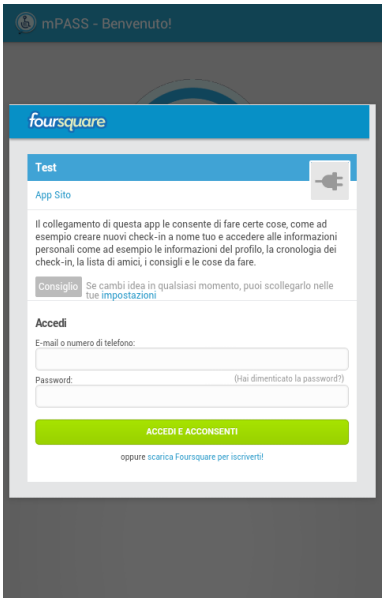

Figura 3.10 Pagina Web per il login a Foursquare

Questa modalità è stata progettata principalmente per permettere all'utente di velocizzare il processo di registrazione e allo stesso tempo autorizzare l'applicazione a comunicare con Foursquare. Ciò è possibile attraverso la classe FoursquareApp, che visualizza una finestra Web con il login al social nework come mostrato nella Figura 3.10. In questa pagina, è possibile inserire il proprio account di Foursquare, autorizzando così l'applicazione a comunicare con esso. Successivamente vengono richiesti attraverso HTTP i dati dell'account, tra cui nome e immagine, che vengono memorizzati in locale con la classe User.

- Creazione nuovo profilo:

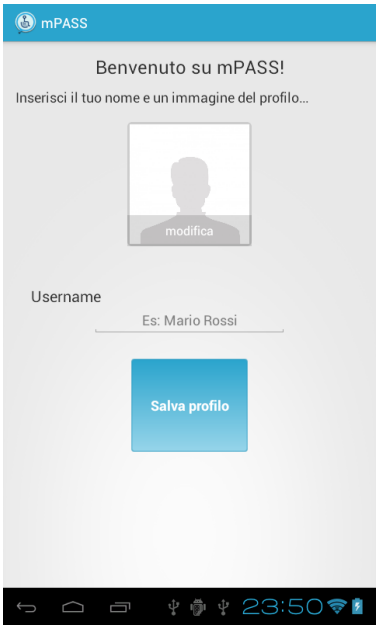

Figura 3.11 Schermata per la creazione del profilo

In questo caso l'utente deve inserire manualmente lo username e facoltativamente un'immagine del profilo, premendo il pulsante apposito. È possibile selezionare un'immagine dalla memoria del dispositivo, oppure scattare una foto con la telecamera, se disponibile.

- Salta (Resta anonimo): semplicemente viene salvato il profilo utente avente nome "Anonimo" e nessuna immagine del profilo.

3.3 Visualizzazione dei dati

In seguito alla prima fase di avvio dell'applicazione, la schermata seguente è la mappa fornita da Google Maps, dove vengono visualizzate tutte le segnalazioni presenti nella Fusion Table, come si può osservare dalla Figura 3.12.

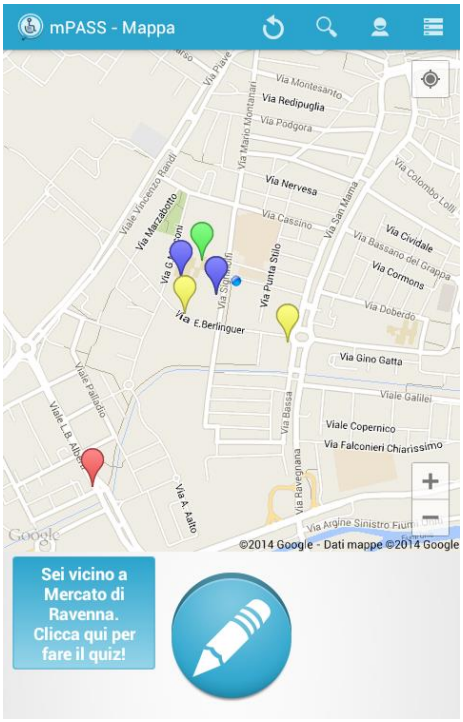

Figura 3.12 Mappa delle segnalazioni

```
...
mMap=((MapFragment)getFragmentManager().findFragmentById(R.id.ma
p)).getMap();
ftclient.setQuery(query_venues);
ftclient.queryOnNewThread("setmarkers");
...
```

Figura 3.13 Codice per inizializzare la mappa

Questo codice (Figura 3.13) mostra le prime azioni eseguite dall'activity. Prima di tutto viene inizializzata la mappa, in seguito viene richiesto alla Fusion Table di restituire i cento luoghi più vicini su cui sono state inviate delle segnalazioni. Il comando SQL utilizzato per compiere questa operazione è il seguente:

```
SELECT ROWID, fsqid, name, geo, accessLevel FROM
"+Costants.tableId+" ORDER BY ST_DISTANCE(geo,
LATLNG(@LAT,@LNG)) LIMIT 100
```

Figura 3.14 Comando SQL per ottenere le segnalazioni vicine

Nell'istruzione precedente (Figura 3.14) vengono richiesti in particolare l'identificatore, il nome, la locazione geografica e il livello di accessibilità dei locali (solo i primi cento più vicini alla posizione dell'utente, la quale viene ricavata tramite una classe apposita per gestire il GPS del dispositivo). Si limita il numero di luoghi restituiti perché scaricare un numero troppo alto di dati può peggiorare significativamente le prestazioni dell'applicazione, specialmente se il dispositivo è datato.

Una volta completata la richiesta alla Fusion Table, viene posizionato un marcatore sulla mappa per ogni segnalazione, con un colore diverso a seconda del suo livello di accessibilità:

- verde se è accessibile,
- giallo se è parzialmente accessibile,
- rosso se non è accessibile,
- grigio se l'utente che ha inviato la segnalazione non ha specificato un livello di accessibilità.

Figura 3.15 Finestra con i dettagli di una segnalazione

Alla pressione del marcatore viene mostrato un riquadro con il nome del luogo, che se selezionato provoca l'apertura di una nuova finestra contenente tutti i dettagli sulla segnalazione, come mostrato dall'esempio nella Figura 3.15. A tale scopo l'applicazione esegue un'ulteriore richiesta alla Fusion Table, richiedendo tutte le informazioni inviate

sul luogo interessato. Il titolo della finestra prende il colore relativo al livello di accessibilità e per ogni questionario compilato vengono visualizzati data di segnalazione, nome dell'utente a cui appartiene, commento e dati relativi alla presenza di porte, ascensori, scale mobili e parcheggi per disabili se presenti.

Nel menu della schermata principale sono presenti anche i pulsanti per ricaricare la mappa o per applicare dei filtri su di essa:

- Ricarica: alla pressione di questo tasto, l'utente ha la possibilità di ricaricare i dati mostrati sulla mappa, scegliendo se vuole che si visualizzino solo i luoghi vicini oppure tutti quelli presenti nella Fusion Table. In seguito vengono eliminati tutti i marcatori ed eseguita una nuova richiesta dei dati, come all'avvio dell'activity.
- Filtri: questo pulsante mostra una nuova finestra dove si possono decidere quali filtri applicare alla mappa, mostrata nella Figura 3.16.

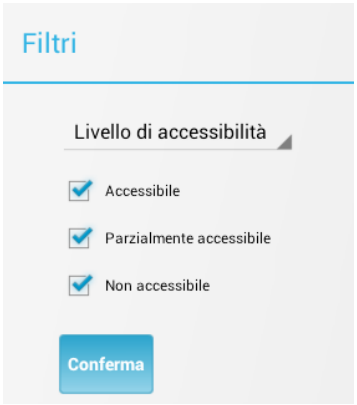

Figura 3.16 Finestra per la gestione dei filtri

Al suo interno è presente un menu a tendina, nel quale è possibile selezionare una categoria su cui applicare il filtro (per esempio nella figura è stato scelto il livello di accessibilità). Tramite le checkbox si può decidere quali valori considerare durante la visualizzazione della mappa. Una volta premuto il tasto "Conferma", la finestra si chiude e viene fatta una query alla tabella, richiedendo solo le segnalazioni che rispettino le condizioni selezionate.

All'interno dell'applicazione, è stato progettato un'ulteriore metodo per mostrare i dati dei luoghi, sfruttando una visualizzazione in lista piuttosto che con la mappa, raggiungibile sempre attraverso il menu.

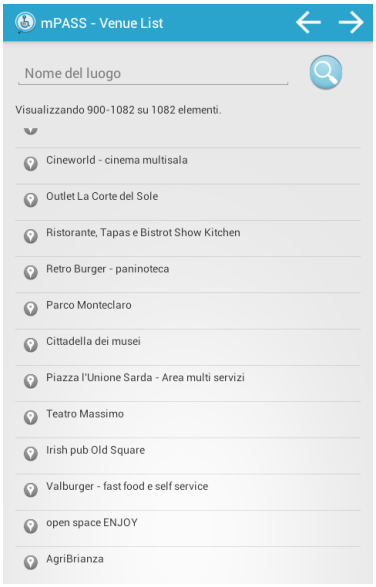

Figura 3.17 Lista dei luoghi

La Figura 3.17 rappresenta l'activity di visualizzazione in lista, pensata per offrire un modo diverso e immediato per cercare un luogo specifico, piuttosto che navigare la mappa. A tale scopo è presente un campo di testo dove l'utente può immettere il nome del locale che vuole trovare. Alla pressione del tasto a fianco viene eseguita la ricerca. Data la grandezza del database, la lista è divisa in pagine da cento elementi l'una, navigabili tramite le frecce nel menu della schermata.

3.4 Profilo utente

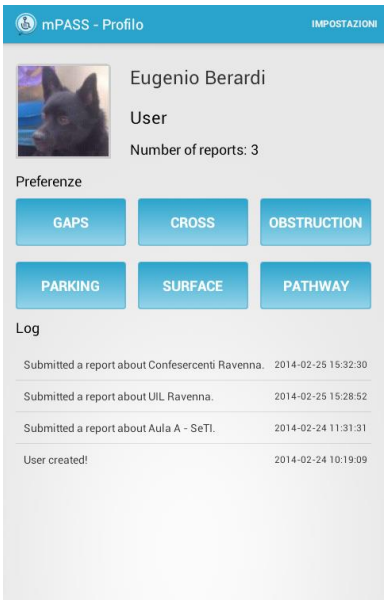

Figura 3.18 Profilo utente

Nel menu della schermata principale è presente il pulsante per accedere al profilo utente, mostrato dalla Figura 3.18. All'inizio dell'activity viene inizializzata la classe User, che si occupa di ricavare tutti i dati del profilo dalle SharedPreferences, i quali vengono poi mostrati nella schermata. Per il registro delle azioni invece viene aperto il database con LogDbManager ed in seguito eseguita una query per ottenere tutte le informazioni al suo interno. Tutte le voci vengono inserite in una ListView in fondo al profilo. Inoltre sono state inseriti i pulsanti per permettere di scegliere le proprie preferenze fra sei categorie di barriere architettoniche. Ad ogni categoria è associata una finestra dove vengono mostrati diversi tipi di ostacoli, su cui l'utente può indicare se ne è interessato o meno.

Figura 3.19 Modifica profilo

Nel menu del profilo si può passare alle impostazioni del profilo (Figura 3.19), dove è possibile modificare diverse caratteristiche dell'utente, fra cui il nome e l'immagine. Anche in caso l'utente si sia registrato manualmente, è comunque presente un pulsante per effettuare il login a Foursquare in qualsiasi momento, attraverso la classe FoursquareApp. L'ultima funzionalità di questa schermata è il tasto per resettare il profilo, alla cui pressione viene mostrato un messaggio per confermare la volontà da parte dell'utente di cancellare i suoi dati. In caso positivo, vengono eliminati tutti i file locali con le preferenze dell'utente e azzerato il database delle azioni. In seguito l'applicazione verrà ravviata mostrando la schermata di benvenuto.

3.5 Creazione segnalazione

Figura 3.20 Schermata dei luoghi vicini all'utente

Il pulsante sotto la mappa della schermata principale è adibito alla selezione dei luoghi per la creazione di nuove segnalazioni. Alla sua pressione vengono mostrati i trenta luoghi più vicini alla posizione geografica dell'utente ricavati da Foursquare. Le informazione sui locali, nome, indirizzo e distanza, vengono poi inseriti nella ListView presente nel layout della schermata, rappresentata nella Figura 3.20.

Visto che non sempre il luogo di interesse dell'utente potrebbe essere presente nella lista a causa della sua posizione, è stato aggiunto un pulsante per la ricerca. La funzionalità di tale elemento consiste nel visualizzare un campo di testo, dove sia possibile inserire una chiave di ricerca da utilizzare per le richieste a Foursquare.

Se attraverso il social network non viene trovato il luogo che si stava cercando, si può ricorrere ad un'altra funzionalità per crearne uno nuovo, raggiungibile dal menu dell'activity.

Figura 3.21 Creazione di un nuovo luogo

Per inserire un nuovo locale, è necessario specificare il suo nome con il campo di testo apposito e la sua locazione geografica, indicandola nella mappa attraverso il posizionamento di un marcatore. Una volta inserite le informazioni si può terminare l'activity premendo il tasto di conferma nel menu. Dato che gli identificatori dei luoghi creati con questa modalità sono diversi da quelli presenti su Foursquare, viene utilizzata una stringa con prefisso "NF" seguito da un numero intero. A tale scopo viene eseguita una query alla Fusion Table per ricavare la chiave primaria con intero più grande, in modo da creare l'identificatore per il luogo appena creato.

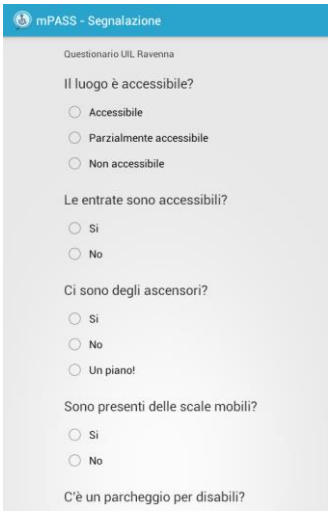

Figura 3.22 Segnalazione di un luogo

In seguito all'inserimento o selezione di un luogo l'applicazione mostra il questionario di accessibilità. Al suo interno è possibile esprimere un giudizio sui seguenti aspetti:

- livello di accessibilità del luogo,
- accessibilità delle entrate,
- presenza di ascensori,
- presenza di scale mobili,
- presenza di parcheggi per disabili nelle vicinanze,

Oltre a queste valutazioni, è presente un campo di testo nel quale l'utente può inserire dei commenti sul luogo.

Al termine della compilazione del questionario vengono compiute le seguenti azioni:

- Invio dei dati alla Fusion Table attraverso la classe FTClient.
- Inserimento dell'azione appena compiuta nel database del registro.
- Infine nel caso l'utente sia collegato a Foursquare, viene eseguito il check-in sul luogo appena segnalato.

Quando viene ricevuta la conferma che i dati sono stati salvati nella Fusion Table, l'applicazione tornerà alla schermata principale, mostrando un messaggio di avvenuta segnalazione all'utente.

Conclusioni

Il progetto ha dato frutto ad un'applicazione per Android con tutte le funzionalità previste, quindi che possa permettere agli utenti di condividere informazioni sugli ostacoli incontrati durante la loro vita quotidiana. Inoltre il software sviluppato fornisce diversi mezzi di lettura dei dati, attraverso mappe e visualizzazione in lista, potenziati da funzionalità di ricerca e filtraggio. Ciò permette alle persone di potersi informare sullo stato di accessibilità dei luoghi, per evitare quelle che chiamiamo barriere architettoniche.

L'applicazione è stata costruita per il sistema operativo Android essendo quest'ultimo gratuito, oltre che semplice per lo sviluppo grazie alla sua natura open source e all'ampia documentazione disponibile sull'API. I servizi disponibili in rete si sono resi fondamentali per la realizzazione delle funzionalità di base dell'applicazione: Foursquare è un ottimo strumento per ottenere dati su luoghi di qualsiasi genere, grazie al suo ampio database è stato utile nella ricerca di punti di interesse per l'utente. Inoltre le sue funzioni di check-in e tip permettono all'applicazione di sfruttare il social network per diffondere in modo più vasto le segnalazioni inviate. Il servizio impiegato per la memorizzazione dei dati è stato Google Fusion Table: una piattaforma gratuita e progettata per essere facile da utilizzare per chiunque. Le sue caratteristiche ci hanno permesso di integrare le nostre informazioni con quelle di altri servizi pubblici, i quali a loro volta si occupano di accessibilità urbana, permettendoci di offrire sin dall'inizio indicazioni su un vasto numero di punti di interesse. Infine Google Maps ha fornito la mappa su cui vengono visualizzate le segnalazioni inviate. Sono state implementate numerose funzionalità per filtrare i dati, in modo da permettere agli utenti di decidere quali categorie di luoghi mostrare sullo schermo. Inoltre all'interno dell'applicazione, è possibile creare il proprio profilo, inserendo le proprie informazioni oppure attraverso il login con Foursquare. Il profilo viene principalmente utilizzato per mantenere in memoria le preferenze dell'utente, le quali gli permettono di personalizzare l'applicazione.

Il progetto può evolversi ulteriormente in diversi modi. Sebbene Fusion Table sia un servizio gratis di facile utilizzo grazie alla sua API, può risultare limitato per certi scopi. La sua natura aperta infatti non permette di memorizzare dati sensibili, inoltre Google limita il numero di accessi giornalieri alla tabella, rendendola inadatta ad un uso costante da parte di numerosi utenti in rete. Una possibile soluzione potrebbe essere la

creazione di un server adibito alla gestione delle informazioni sulle segnalazioni, cosicché possano essere implementate funzioni non previste dal servizio di Google. Inoltre sarebbe interessante utilizzare il server anche per gestire gli utenti e permettere loro di interagire, creando così una vera e propria comunità online, il cui scopo è quello di migliorare l’accessibilità delle città. Questo però porterebbe ad un’ulteriore grado di complessità nell’implementazione del progetto, oltre che alla necessità di assicurarsi di mantenere la privacy degli utenti. Un’altra eventuale modifica al progetto potrebbe essere l’utilizzo di Open Street Map (OSM) per la visualizzazione della mappa, sostituendo così Google Maps. OSM è un progetto open source di mappatura del pianeta, che negli ultimi anni si è evoluto costantemente, rivelandosi un’interessante alternativa ad altri servizi simili. Utilizzare Open Street Map consentirebbe pieno controllo sulla rappresentazione della mappa dell’applicazione, slegando quest’ultima da servizi esterni che si possono dimostrare limitanti. Oltretutto il progetto potrebbe essere realizzato anche per altri sistemi operativi mobile, come iOS, in modo da raggiungere il maggior numero di utenti possibili, dato che lo scopo del progetto è effettivamente quello di ottimizzare la qualità di vita di ognuno, senza distinzioni.

Bibliografia

[BA14] Wikipedia, “Barriera architettonica”, 2014, http://it.wikipedia.org/wiki/Barriera_architettonica

[VES06] F. Vescovo, “Accessibilità: concetti base”, 2006, http://www.progettarepertutti.org/normativa/cap1/ppt_cap1.html

[ZEC12] E. Zecchini, C. Agnesi, “Barriere architettoniche e barriere sensoriali”, 2012, http://www.arcipelagosordita.it/pdf/Barriere%20architettoniche%20e%20barriere%20sensoriali%201.pdf, pp. 64-65

[MOR06] L. Morelli, M. Nocco, A. Petrillo, S. Stolfi, “La destinazione accessibile”, 2006, http://www.euforme.net/css/it/project_work/destinazione.pdf, pp. 14-15

[SC07] European Smart Cities, “Smart cities, Ranking of European medium-sized cities”, 2007, http://www.smart-cities.eu/download/smart_cities_final_report.pdf

[JOS14] J. M. Hernàndez-Munoz, J. B. Vercher, L. Munoz, J. A. Galache, M. Presser, L. A. H. Gomez, J. Pettersson, “Smart Cities at the Forefront of the Future Internet”, http://www.smartsantander.eu/downloads/Presentations/fia_book_2011_smartcities.pdf

[SAS14] Casa e Clima, “Sensori, app e smartphone: il modello della smart city Santander”, http://www.casaeclima.com/ar_13316_ESTERO-Europa-santander--smrt-city-Sensori-app--e-smartphone-il-modello-della-smart-city-Santander.html

[PRI13] Priscila Pacheco, “Is crowdsourcing the next revolution in urban accessibility?”, 2013, http://thecityfix.com/blog/rota-acessivel-app-accessibility-crowdsourcing-priscila-kichler-pacheco/

[CRW14] Wikipedia, “Crowdsourcing”, 2014, http://en.wikipedia.org/wiki/Crowdsourcing

[VSC13] Pagamenti digitali, “Varese diventa una Smart City grazie all’NFC”, 2013, http://www.pagamentidigitali.it/pubblica-amministrazione/116_varese-diventa-una-smart-city-grazie-all-nfc.htm

[CRI12] E. De Cristofaro, C. Soriente, "Participatory Sensing: Enabling Privacy in Participatory Sensing", 2012, http://arxiv.org/pdf/1201.4376v1.pdf

[AVI12] Jonathan Avila, "The Mobile Accessibility Landscape", 2013, https://www.ssbbartgroup.com/blog/2012/06/29/the-mobile-accessibility-landscape/

[MA13] Web Accessibility Initiative, "Mobile Accessibility", 2013, http://www.w3.org/WAI/mobile/

[SRS14] WebAIM, "Survey of Preferences of Screen Readers Users", http://webaim.org/projects/screenreadersurvey/

[TRA14] Wikipedia, "Trackball", http://it.wikipedia.org/wiki/Trackball

[TTY14] Wikipedia, "Teleprinter", http://en.wikipedia.org/wiki/Teleprinter

[BRA14] Wikipedia, "Refreshable braille display", http://en.wikipedia.org/wiki/Braille_display

[SCR14] Wikipedia, "Screen reader", http://en.wikipedia.org/wiki/Screen_reader

[ASC14] Apps for Smart Cities, "The Apps for Smart Cities Manifesto", http://www.appsforsmartcities.com/?q=manifesto

[IOT14] Wikipedia, "Internet of Things", http://en.wikipedia.org/wiki/Internet_of_Things

[SWA13] H. Swan, G. F. Williams, J. Avila, "BBC Mobile Accessibility Standards and Guidelines", 2013, http://www.bbc.co.uk/guidelines/futuremedia/support/028_MobileSandG_v7_production.pdf

[AND14] Android, Wikipedia, http://en.wikipedia.org/wiki/Android_(operating_system)

[AD14] Android Developers, "Introduction to Android", http://developer.android.com/guide/index.html

[GON10] H. Gonzales, A. Halevy, C. S. Jensen, A. Langen, J. Madhavan, R. Shapley, W. Shen, "Google Fusion Tables: Data Management, Integration and Collaboration in the Cloud", 2010, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.298.273&rep=rep1&type=pdf

[FT14] Google Developers, "Google Fusion Table API", https://developers.google.com/fusiontables/

[FSQ14] Wikipedia, “Foursquare”, http://en.wikipedia.org/wiki/Foursquare

[DFSQ14] Foursquare Developers, “Foursquare API”, https://developer.foursquare.com/start

[GM14] Wikipedia, “Google Maps”, http://en.wikipedia.org/wiki/Google_maps

[JAV14] Wikipedia, “Java (programming language)”, http://en.wikipedia.org/wiki/Java_(programming_language)

[JEN14] Jakob Jenkov, “OAuth 2.0 Tutorial”, http://tutorials.jenkov.com/oauth2/index.html

[HTTP14] Wikipedia, “Hypertext Transfer Protocol”, http://en.wikipedia.org/wiki/Http

[JSON14] Wikipedia, “JSON”, http://en.wikipedia.org/wiki/Json

[SQL14] SQLite, http://www.sqlite.org/