Universitat
Autònoma
de Barcelona

(2876-2: ONTHEBUS)

Memòria del Projecte Fi de Carrera

d'Enginyeria en Informàtica

realitzat per

Eric Xavier Lara Amat y León

i dirigit per

Jordi Roig de Zárate

Bellaterra,12 de Setembre de 2011

Universitat Autònoma de Barcelona

Escola Tècnica Superior d'Enginyeria

El sotasignat, Jordi Roig de Zárate

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Eric Xavier Lara Amat y León

I per tal que consti firma la present.

Signat: .....

Bellaterra, 12 de Setembre de 2011

Agraïments:

En primer lloc, voldria agrair a Jordi Roig Zárate i a Marc Vallribera i Ros el haver-me donat la oportunitat d'entrar a formar part d'aquest projecte tan interessant i haver-me fet de brúixola fins arribar al final del trajecte.

També voldria agrair als meus companys pels bons moments que hem passat, que no han sigut pocs.

No puc oblidar a Ana, que sense els seus consells i la seva motivació, això no hauria acabat així i als meus pares, que sempre estan presents.

Índex

1 Introducció

1.1 Motivació

Aquest projecte neix de les creixents necessitats de mobilitat actuals. Cada dia ens movem més i més lluny, per estudiar, treballar, comprar, visites al metge, oci, etc. Aquestes necessitats són també presents en els grans nuclis urbans, on pot ser necessari cobrir grans distàncies per poder arribar a un determinat destí. Moltes persones utilitzen el transport privat com a forma de desplaçament, però cada cop més gent opta per el transport públic.

Si be els mitjans de transport públic eren anys enrere considerats una forma de transport per persones amb pocs recursos econòmics, com estudiants o jubilats, cada dia més gent opta per aquest tipus de transport per diversos motius, que poden ser econòmics o ecològics d'entre d'altres. Els parcs automobilístics de les ciutats no paren de créixer, i es comencen a fer patents problemes derivats de d'aquest augment de vehicles, com, l'augment de la pol·lució ambiental, la dificultat per aparcar en nuclis molt concorreguts o la saturació d'algunes vies. És per això que les autoritats cada cop són més conscients de la necessitat de pacificar el trànsit creant zones peatonals, zones trenta i impulsar el transport públic com a forma preferent de desplaçament, creant noves línies de metro i bus a les ciutats.

Aquest increment en el número de línies d'autobús ha provocat que actualment sigui molt complex determinar quina és la línia d'autobús que hem d'agafar per arribar al nostre destí, determinar quins transbords podem fer o quina es la disponibilitat horària de cada línia. Per això, les companyies d'autobús que operen en cada zona, proporcionen tota aquesta informació en una pagina web, o en marquesines especialment ubicades a les parades. Les pagines web però, són de difícil consulta des del carrer, especialment si opera més d'una companyia a la zona, i la informació no es troba unificada en una única pàgina. Les marquesines en canvi, tenen més aviat una funció de validació, ja que podem determinar si a la parada on ens trobem hi circula la línia que volem agafar, encara que aquestes marquesines poden ser vandalitzades. El problema d'accedir a aquesta informació, es pot agreujar més encara en persones que no estan familiaritzades amb la zona o persones discapacitades, així com persones d'edat avançada.

Per sort, en el món actual, la presencia de terminals mòbils amb accés a Internet és tota una realitat. Aquestes plataformes, a més de disposar de connexió a Internet, incorporen tota

mena de sensors, com brúixoles, giroscopis, acceleròmetres i GPS, una molt bona combinació per poder desenvolupar aplicacions que solucionin aquestes necessitats creixents.

1.2 Què volem fer, on volem arribar

El que és vol aconseguir, es unificar tota la informació referent a les línies d'autobús de diverses ciutats en una única aplicació, amb una interfície universal de fàcil accés i ús, per tal de poder guiar al usuari fins a la parada d'autobús, donar informació dels temps d'espera, i informar-lo durant el trajecte amb l'autobús, indicant a l'usuari quan ha de baixar de l'autobús per arribar a un determinat destí.

Es vol aconseguir una aplicació intuïtiva, fàcil d'utilitzar, tant per aquelles persones acostumades a fer servir dispositius mòbils, com per aquelles persones que hi tenen algunes dificultats, com les persones grans.

Degut a que els terminals mòbils actuals no disposen de software adaptat per a persones amb discapacitats visuals, volem utilitzar el disseny universal per fer el més accessible possible la nostre aplicació, perquè incloses aquestes persones puguin fer-la servir de forma fàcil, senzilla i pràctica.

No es vol que la aplicació es quedi al nivell d'una prova pilot en una ciutat concreta, volem estendre la possibilitat d'utilitzar l'aplicació en diverses ciutats del món, per presentar una solució realista al problema del guiatge en autobús dins les ciutats.

2 Estudi de viabilitat

2.1 Estat de l'art

En aquesta secció es comentarà l'estat de l'art de les aplicacions de guiatge en transport públic accessible.

En una primera part s'explicarà l'estat actual d'aquest tipus d'aplicacions, per a passar després a detallar l'estat de cadascuna de les vessants.

2.1.1 ESTAT ACTUAL DEL MERCAT

En els últims anys, els sistemes de guiatge personal, han cobrat especial importància, es poden trobar nombrosos desenvolupaments de sistemes de guiatge, tant gratuïts com de pagament, així com exemples de implementacions senzilles de sistemes de seguiment de rutes de codi obert.

No obstant això, aquests sistemes solen estar orientats al desplaçament autònom (sense considerar l'ús de sistemes de transport públic), i no s'han trobat referències a sistemes de guiatge per a persones discapacitades.

Amb l'arribada de dispositius de cost relativament reduït, amb la tecnologia necessària per a implementar aquests sistemes de guiatge, és d'esperar que el nombre d'aplicacions d'aquest àmbit augmentin amb rapidesa, sobre tot tenint el compte els costos reduïts, per al seu desenvolupament, així com l'ampli ventall d'usuaris.

2.1.2 SISTEMES DE GUIATGE

Els sistemes de guiatge, que estan àmpliament difosos en les plataformes, es poden prendre com a referència gran nombre d'aplicacions com GoogleMaps, TomTom o AndNav.

Aquestes aplicacions es basen en la utilització de sistemes de cartografia externs al dispositiu que es descarreguen a través de serveis online utilitzant la connexió a Internet del dispositiu o bé incorporen mapes juntament amb l'aplicació, tant gratuïts com de pagament, així com el sistema civil de GPS (Geo Posicionament per Satèl·lit).

En aquest àmbit a l'hora de utilitzar tant els sistemes cartogràfics, com el sistema de GPS s'han trobat diversos problemes, alguns d'ells àmpliament coneguts i d'altres amb escassa literatura disponible, que han fet que s'hagin hagut de modificar diverses especificacions de

desenvolupament. Totes aquestes problemàtiques seran àmpliament detallades en els respectius punts.

2.1.3 SISTEMES D'AJUDA A L'ÚS DE TRANSPORT PÚBLIC

Els últims dos anys s'ha incrementat la demanda d'ajuda a l'hora de desplaçar-se per la ciutat en transport públic, degut a l'alt volum de tràfic produït pel transport privat i les despeses associades a ell, així com per la poca informació de qualitat proporcionada per les pròpies companyies de transport.

Aquesta mala gestió per part de les companyies de transport, està essent lentament suplida mitjançant aplicacions proporcionades per companyies de desenvolupament de software (UrbanStep), projectes universitaris (ValenciaBus), o aplicacions desenvolupades per la pròpia companyia de transports (TMB Virtual).

Paradoxalment, per tal implementar els sistemes d'ajuda a l'ús del transport públic ha sigut necessari fer mineria de dades amb la informació proporcionada per les companyies de transport públic, ja que la gran majoria de companyies no proporcionen la informació en un format fàcil de manipular informàticament. Per exemple, en alguns casos ha sigut necessari programar un software per tal d'extreure automàticament les dades de les línies i parades d'autobusos a partir de la informació mostrada en la pàgina web de la companya, o ha calgut fer una transformació dels sistemes de coordenades utilitzats per les companyies a un sistema únic per a que totes les dades funcionin correctament a OnTheBus.

2.1.4 SISTEMES D'AJUDA ALS USUARIS AMB PROBLEMES D'ACCESSIBILITAT

Els sistemes d'ajuda al discapacitat en dispositius basats en *Android* es troben en fases molt verdes de la seva evolució. Tot i que vàries funcionalitats que són necessàries per a implementar un sistema d'aquest tipus es troben disponibles mitjançant la pròpia arquitectura proporcionada pel sistema, aquestes manquen de la estabilitat i maduresa necessàries per a fer aplicacions comercials per a col·lectius de discapacitats.

D'altre banda les limitacions no sempre venen donades per a l'arquitectura, sinó que pot ser que sigui el propi dispositiu el que dificulti l'accessibilitat, degut a la utilització de components més assequibles i de baixa qualitat per a abaratir el preu final del dispositius.

2.2 Tecnologia a emprar

En aquest apartat es defineixen les tecnologies requerides per al desenvolupament d'aquest projecte.

2.2.1 El sistema operatiu Android

Android és un sistema operatiu per a dispositius mòbils en general, com poden ser telèfons, tablets, reproductors MP3, etc, encara que recentment s'està expandint el seu ús per a tot tipus de dispositius, com poden ser televisors o fins i tot microones o rentadores.

Inicialment, la companya *Android Inc.* va començar el seu desenvolupament, i posteriorment va ser comprada per Google a l'any 2005. La seva llicència és *Apache Software Licence,* de codi obert.

Google, a l'any 2007, va fundar la *Open Handset Alliance (OPH)*, que es tracta d'una aliança comercial formada per 78 companyies que desenvolupen estàndards oberts per dispositius mòbils. En formen part, entre d'altres, *HTC*, *Motorola, NVidia* i *Samsung*.

Al novembre de 2007, es va publicar la primera versió de la API d'*Android*, que és una interfície de programació d'aplicacions que conté el conjunt de procediments en forma de llibreria de software que permet utilitzar els components dels terminals *Android* (sensors, pantalla, SO, etc) per desenvolupar aplicacions.

Existeixen altres sistemes operatius per a dispositius mòbils, com per exemple IOS de l'empresa *Apple*, *BlackBerry OS* de *RIM (Research in Motion)*, *Symbian* de *Nokia* o *Windows Phone* de *Microsoft*. Un dels motius de que s'hagi escollit el sistema operatiu *Android* és que ofereix més possibilitats, facilitat de programació i garanties d'èxit que qualsevol dels altres esmentats.

Per exemple, en el cas de *IOS* el *kit* de desenvolupament només està disponible per a ordinadors amb sistema operatiu *MAC OS*, també fabricat per *Apple*, i no tots els integrants del grup de OnTheBus en posseeixen un. *Symbian* era el sistema operatiu que més quota de mercat posseïa en el moment de la decisió, però en els últims anys havia anat perdent usuaris i quedant-se antiquat: fins i tot la companya que el desenvolupa, *Nokia*, havia començat a desenvolupar d'altres sistemes operatius per substituir-lo. *Windows Phone* era un sistema operatiu massa nou i no existia molta literatura per poder aprendre'n ràpidament el llenguatge i sistema de programació.

Però el que va acabar decidint que es faria servir *Android* és el ràpid creixement i acceptació que estava tenint, que feia presagiar que en pocs mesos seria el sistema operatiu

més utilitzat. A més a més, al ser un sistema operatiu ofert a terminals de diferents marques, existeix una àmplia varietat de dispositius de diferents preus, pel que es va pensar que seria molt més fàcil arribar a tots els tipus d'usuaris que es pretenia: d'elevada edat, estudiants, persones amb problemes d'accessibilitat o no.

2.2.1.1 Versions de la API d'Android

Actualment existeixen 13 versions de la API d'*Android*. Cada vegada que apareix una nova versió, incorpora noves funcionalitats i s'actualitza constantment. A la següent *Taula 1* es mostren les diferents versions i el seus noms.

Platform Version	API Level	VERSION_CODE
Android 3.2	13	HONEYCOMB_MR2
Android 3.1.x	12	HONEYCOMB_MR1
Android 3.0.x	11	HONEYCOMB
Android 2.3.4 Android 2.3.3	10	GINGERBREAD_MR1
Android 2.3.2 Android 2.3.1 Android 2.3	9	GINGERBREAD
Android 2.2.x	8	FROYO
Android 2.1.x	7	ECLAIR_MR1
Android 2.0.1	6	ECLAIR_0_1
Android 2.0	5	ECLAIR
Android 1.6	4	DONUT
Android 1.5	3	CUPCAKE
Android 1.1	2	BASE_1_1
Android 1.0	1	BASE

Taula 1 - Versions de la API d'Android

2.2.1.2 Justificació de la versió de la API utilitzada

Durant les primeres reunions de grup, es va veure que els seus membres ja havien començat a fer projectes de prova per anar aprenent a programar per *Android*, però no tots havien fet servir la mateixa versió. Es va investigar i es va veure que en principi no hi havia cap

problema amb qualsevol de les versions disponibles en aquell moment, ja que, malgrat que a la API s'indicava que algunes funcions no estaven disponibles per les primeres versions, en principi cap d'elles semblava que anés a suposar un inconvenient per al desenvolupament del projecte. Però davant de tantes versions diferents, calia posar-se d'acord en utilitzar la mateixa versió tots els components del grup, per evitar problemes que poguessin sorgir més endavant.

Es van mirar les estadístiques de les versions que incorporaven els terminals a la pàgina web de la API, on s'observà que gairebé el 80% dels terminals tenien una versió igual o superior a *Android* 2.1. A més a més, es va veure que molts fabricants havien anunciat actualitzacions per els terminals que tenien al mercat amb les versions 1.5 i 1.6.

Aleshores, es va prendre la decisió de que la versió mínima de la API que suportaria *OnTheBus* seria *Android* 2.1 en un inici.

A la següent *Figura 1* i *Taula 2*, es poden veure les estadístiques actuals de les versions utilitzades.

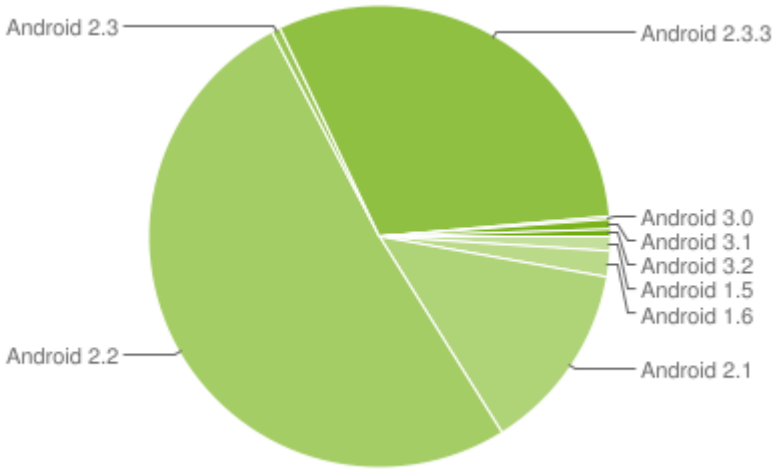


Figura 1- Gràfic circular de les versions d'Android que incorporen els terminals a Setembre de 2011.

Platform	Codename	API Level	Distribution
Android 1.5	Cupcake	3	1.0%
Android 1.6	Donut	4	1.8%
Android 2.1	Eclair	7	13.3%
Android 2.2	Froyo	8	51.2%
Android 2.3 - Android 2.3.2	Gingerbread	9	0.6%
Android 2.3.3 - Android 2.3.4		10	30.7%
Android 3.0	Honeycomb	11	0.2%
Android 3.1		12	0.7%
Android 3.2		13	0.5%

Taula 2 - Resum dels percentatges de terminals de cada versió d'Android a Setembre de 2011

Sumant els percentatges de les versions majors a 2.1, es pot observar que són el 97.2% dels terminals.

Per tant, la decisió inicial va ser encertada, ja que molts terminals s'han anant actualitzant amb el temps i els nous terminals que han anat sortint al mercat ja incorporen les últimes versions. A la següent *Figura 2* es pot veure aquesta tendència, on destaca l'estancament de les versions inferiors a *Android* 2.1 i també el ràpid creixement de la versió 2.3 *Gingerbread*, que al inici del desenvolupament d'aquest projecte, encara no estava disponible.

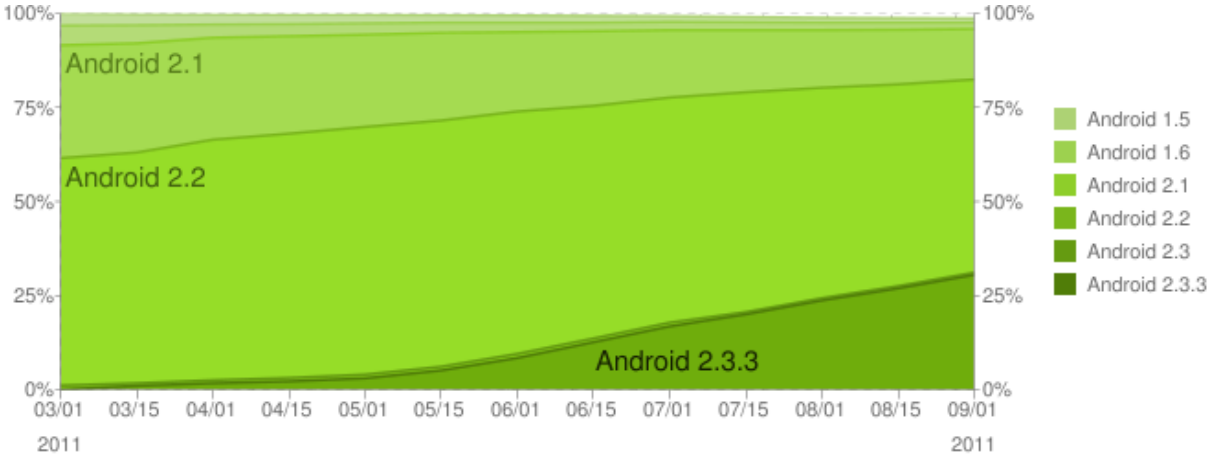


Figura 2 - Evolució de les versions dels terminals entre els mesos de Març i Setembre de 2011

2.2.2 SERVEIS DE CLOUD COMPUTING

Cloud Computing (computació en el núvol o núvol de còmput) és una tecnologia informàtica que presta un servei de computació en servidors remots o centres de dades emmagatzemades a Internet.

L'usuari del servei no ha de tenir coneixements en la matèria del càlcul ni ha de crear i mantenir la infraestructura que requereix: l'únic requisit per utilitzar-lo és tenir connexió a Internet.

En aquest projecte, hi ha tres càlculs que no és possible fer-los per diversos motius.

- Calcular les rutes caminant a partir de dos punts, amb indicacions pas a pas.
- Aconseguir el temps d'espera del bus en temps real (quan l'usuari arriba a la parada) de totes les línies i parades d'autobús de sis ciutats diferents.
- Saber l'adreça completa (país, ciutat, carrer i número) a partir de dues coordenades geogràfiques, i també el pas invers.

Aleshores s'ha decidit que es faran servir serveis de computació en el núvol en aquests tres casos. Per tant, és necessari utilitzar un servei de *Cloud Computing* que calculi la ruta caminant entre dos punts, indicant el sentit de cada gir i la orientació (Nord, Sud, etc) de cada part de la ruta. També és necessari un altre servei que faci les funcions de geocodificació (procés que permet obtenir l'adreça completa a partir d'un parell de coordenades geogràfiques) i geocodificació inversa (obtenir les coordenades exactes a partir del nom del carrer, número i ciutat). Per últim, és necessari consultar a les companyies d'autobusos de les diferents ciutats el temps d'espera en temps real, sempre i quan la companyia ho permeti.

2.2.3 SUBVERSION

Subversion és un sistema de control de versions amb llicència lliure *GNU/GPL* que automatitza la tasca de guardar, recuperar i barrejar versions de fitxers, utilitzant el protocol *SVN*.

Permet que diferents usuaris utilitzin el mateix repositori on s'hi allotja el codi font del programa, amb la possibilitat d'actualitzar les modificacions de qualsevol usuari o restaurar l'aplicació a un estat anterior en cas de que sigui necessari. A més a més, permet crear línies de desenvolupament paral·leles amb noves funcionalitats i més endavant, quan es tingui la seguretat de que funcionen correctament, incorporar-les a la versió principal i estable de l'aplicació.

Hi ha d'altres sistemes de control de versions, com *Mercurial* o *Git*, aquest últim molt utilitzat en l'àmbit del software lliure i en el projecte *AOSP* (*Android Open Source Project*), on col·laboren diferents fabricants de dispositius i processadors per millorar el sistema operatiu *Android*. Però es va decidir utilitzar *Subversion* perquè ja se sabia instal·lar i configurar, i l' utilització d'un altre sistema hagués suposat invertir més temps en aprendre a instal·lar-lo i utilitzar-lo, temps que va ser preferible invertir en aprendre a programar en *Android*.

2.2.4 ENTORN DE DESENVOLUPAMENT

En aquest apartat es descriu el software necessari per poder desenvolupar OnTheBus

Ecllipse IDE i Java Standard Development Kit (SDK): *Eclipse* és un entorn de desenvolupament integrat que ens permet programar en diferents llenguatges de programació, entre ells el llenguatge *Java*, compilar el nostre codi i executar-lo. És necessària també la instal·lació del SDK de *Java*, que és un kit de desenvolupament pel llenguatge *Java*. Hi ha altres entorns per programar en Java perfectament vàlids per aquest projecte, com per exemple *NetBeans*, però *Eclipse* és l'*IDE* recomanat per *Google* i per al que existeixen les últimes versions dels *plugins* que es descriuran seguidament, i és l'*IDE* que els integrants del grup estan acostumats a fer servir i dominen.

Android SDK: instal·lant aquesta eina podrem connectar el nostre terminal Android, ja que ens proporciona els drivers del dispositiu per al nostre sistema operatiu (està disponible per Windows, Linux i MacOS). Si no tenim un dispositiu *Android*, podem simular el sistema operatiu mitjançant un emulador.

ADT Eclipse plugin: com el seu nom indica, és un afegit d'*Eclipse*. És molt complet, ens permet descarregar la versió d'*Android* que desitgem, crear emuladors amb les característiques que necessitem simular (acceleròmetre, *GPS* i altres tipus de sensors, resolució de pantalla, emmagatzemament extern, memòria *RAM*, etc), veure el *Log* d'informació i errors del terminal o emulador que estem executant, dissenyar la nostra interfície gràfica amb un editor gràfic, simular el nostre geoposicionament i fins i tot un recorregut introduint les coordenades geogràfiques de cada punt en un fitxer, entre d'altres coses interessants pel projecte. S'ha fet servir aquest perquè és l'únic que existeix que permet emular el sistema i, per tant, poder provar OnTheBus sense necessitat de tenir un dispositiu físic.

Subclipse: Es tracta d'un *plug-in* per *Eclipse* que permet connectar al repositori *SVN* de l'aplicació i efectuar les operacions modificació de codi i actualització del codi dels altres usuaris dins el mateix entorn de desenvolupament integrat *Eclipse*.

2.3 Planificació del projecte

El projecte s'ha dividit en diferents etapes de dissenys i implementació definint fites a la finalització de cadascuna de les etapes i dependència directe entre la finalització i inici d'algunes d'elles.

1. Anàlisi de requeriments: En aquesta etapa es posa de manifest què es vol desenvolupar i es defineix a alt nivell que es farà.

2. Disseny i implementació objectes comuns: En aquesta etapa es dissenya i implementa els objectes de cada mòdul que seran visibles per la resta.

3. Disseny i implementació dels mòduls: Durant aquesta etapa es desenvolupen les funcionalitats que ha d'oferir cada un dels mòduls.

4. Proves unitàries: Durant aquesta etapa es realitzen proves a nivell modular tant en emulació com en terminal físic.

4. Disseny i implementació de comunicació entre mòduls: Durant aquesta etapa es conjunten tots els mòduls definint la informació a traspassar entre els mateixos i com es traspassa.

5. Proves de l'aplicació: Durant aquesta etapa es realitzen diferents proves de l'aplicació tant en emulació com en terminal físic.

6. Proves de camp (UAT): Durant aquesta prova es realitzen proves de camp fent ús de l'aplicació en escenaris reals. També es realitzen unes proves d'acceptació d'usuari que es duen a terme amb la col·laboració del director del projecte.

La durada de cada una de les etapes així com la seva planificació es pot veure en la següent taula (*Figura 3*):

		Nombre	Duración	Inicio	Terminado
1		Gestió del projecte	1.062 days	22/10/10 16:00	28/07/11 17:00
2		Dinamització	1.062 days	22/10/10 16:00	28/07/11 17:00
9		Gestió general	164 days	10/12/10 8:00	27/07/11 17:30
10		Documentació general	842,667 days	1/11/10 17:00	9/06/11 17:30
16		Definició document de requerimients	277,333 days	25/10/10 8:00	4/01/11 17:30
24		Desenvolupament	442,667 days?	5/01/11 8:00	29/04/11 17:30
25		Mòdul algoritme de cerca	256,633 days	5/01/11 16:03	14/03/11 17:00
26		Disseny / Eleccio de l'algorisme	85,967 days	5/01/11 16:03	27/01/11 17:30
30		Implementació	170,667 days	28/01/11 8:00	14/03/11 17:00
33		Test de l'algorisme	117,633 days	5/01/11 16:03	4/02/11 16:30
43		Mòdul core	111,833 days	15/03/11 8:00	12/04/11 16:45
44		Disseny	26,167 days	15/03/11 8:00	21/03/11 16:15
52		Implementació	53,667 days	21/03/11 16:15	4/04/11 16:45
56		Test del Controlador	32 days	4/04/11 16:45	12/04/11 16:45
60		Mòdul BDD	187,222 days	5/01/11 16:03	23/02/11 16:53
61		Buscar informació del servei d'autobusos	17,037 days	5/01/11 16:03	28/01/11 16:06
62		Aprovisionament dades	18,519 days	28/01/11 16:06	23/02/11 16:53
63		Estudi i Disseny E/R Base dades "Autobús"	2,222 days	5/01/11 16:03	7/01/11 16:23
64		Normalitzar E/R e implementar Base dades "Autobús".	4,444 days	7/01/11 16:23	13/01/11 17:03
65		Omplir Base de Dades "Autobús" amb les dades del ser...	2,667 days	14/01/11 8:00	18/01/11 17:00
66		Implementar (dintre de l'aplicació) la Base de Dades (SQ..	1,481 days	19/01/11 8:00	20/01/11 16:43
67		Generar els mètodes per les diferents connexions, acce..	8,148 days	5/01/11 16:03	17/01/11 16:16
68		Parsejar la informació dels temps que triguen els difere..	11,111 days	5/01/11 16:03	20/01/11 16:13
69		Modul Intericie d'usuari	442,667 days?	5/01/11 8:00	29/04/11 17:30
70		Esbossar un disseny	213,333 days?	5/01/11 8:00	1/03/11 17:00
75		Prototipatge	149,333 days	2/03/11 8:00	8/04/11 17:00
80		Mòdul de só	43,3 days	5/01/11 16:03	17/01/11 17:03
90		Mòdul de texte	21,967 days	5/01/11 16:03	11/01/11 17:03
93		Mòdul Gestures	53,333 days	5/01/11 16:03	19/01/11 16:03
97		Mòdul de Vibració	21,333 days	5/01/11 16:03	11/01/11 16:03
100		Interficie final	80 days	11/04/11 8:00	29/04/11 17:30
105		Mapa	165,967 days	5/01/11 16:03	17/02/11 17:00
106		Estudi de la API de Google Maps	32,633 days	5/01/11 16:03	13/01/11 17:03
107		Control i visualització del Mapa	32 days	14/01/11 8:00	21/01/11 17:00
111		Possicionament en Pantalla	31,667 days	24/01/11 8:00	31/01/11 16:30
114		Rutes al Mapa	69,667 days	31/01/11 16:30	17/02/11 17:00
117		Mòdul Geoposicionament	272 days?	18/02/11 8:00	29/04/11 17:00
118		Geoposicionament GPS	208 days	18/02/11 8:00	13/04/11 17:30
119		Obtenció posició actual	37 days	18/02/11 8:00	28/02/11 16:30
122		Obtenció geoposició d'una adreça	37,667 days	28/02/11 16:30	9/03/11 17:00
127		Guiat rutes a peu - Backend	74,667 days	10/03/11 8:00	29/03/11 17:00
132		Guiat rutes en autobus - Backend	58,667 days	30/03/11 8:00	13/04/11 17:30
136		Obtenció rutes caminades	64 days?	13/04/11 17:00	29/04/11 17:00

Figura 3- Planificació temporal del projecte

Com es pot observar, la data prevista de finalització es el 28 de juliol de 2011, superant el primer termini d'entrega del projecte, tenint com a marge per a la resolució de problemes el mes d'agost.

S'ha de tenir en compte que s'ha definit que la dedicació de cada un dels recursos es de unes 4 hores dia.

2.3.1 DIAGRAMA DE GANTT

A continuació podem veure un diagrama de Gantt (*Figura 4*) amb el calendari del projecte així com les dependències entre les tasques globals.

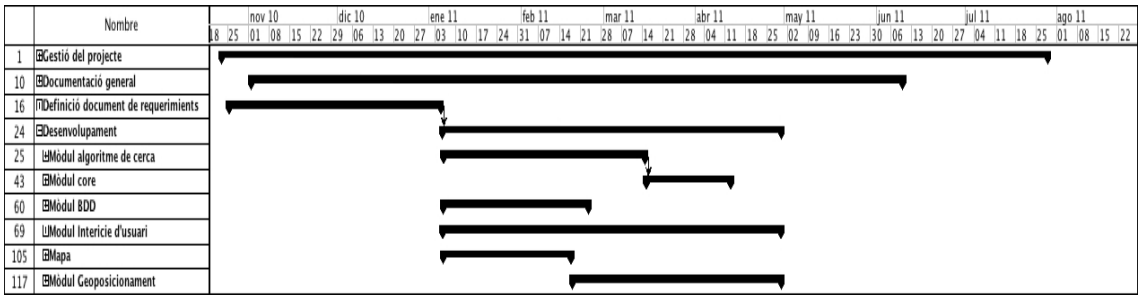


Figura 4 - Diagrama de Gantt

2.4 El disseny universal

> ***"The opposite of useful, the opposite of something that is well designed is… useless" – by @JHockenberry***

> ***"No es pot pensar en la discapacitat de forma general, perquè cadascuna té les seves necessitats específiques. A cada problema, una solució" – Per @Carlos Martínez, vicepresident d'ASOCIDECAT, l'Associació de Sord-Cecs de Catalunya***

La idea de disseny universal, va començar a prendre forma als voltants dels anys 90 com a evolució lògica del disseny sense barreres, accessible i de la tecnologia assistida de recolzament, amb la intenció d'aprofitar-se en un rang ampli de disciplines.

El Disseny Universal es refereix a un conjunt de principis acordats per un grup d'arquitectes, dissenyadors de productes, enginyers i investigadors del disseny ambiental que serveixen com a guia per l'obtenció de productes i entorns que siguin el més flexibles, equitatius, intuïtius possibles entre d'altres característiques, per poder ser utilitzats pel nombre màxim possible de persones, sense necessitats d'adaptacions o de disseny especialitzat. No ens hem d'oblidar que hi ha diversos problemes d'accessibilitat per a un

percentatge d'usuaris, els quals normalment es deixen de banda a l'hora de dissenyar *software*.

En essència són 7 principis que a més de servir per futurs desenvolupaments, també tenen funció avaluadora per dissenys actuals.

Al ser d'àmbit general, és possible que segons el tipus de disseny, algun d'aquests principis es pugui obviar o no ser del tot rellevant en el context de l'aplicació.

Els 7 principis, dels quals s'ha basat el disseny de l'aplicació, són:

1. **Ús equitatiu:** El disseny ha de ser fàcil d'usar i adequat per a totes les persones independentment de les seves capacitats i habilitats. Així doncs, cal evitar segregar a qualsevol usuari i proporcionar les mateixes formes d'ús per a tots. Idèntiques quan sigui possible i equivalents quan no hi hagi cap altre alternativa. Per tant, ha de ser utilitzable i a un preu raonable independent de les problemes d'accessibilitat dels usuaris.
2. **Ús Flexible:** El disseny ha de poder adequar-se a un ampli rang de preferències i habilitats individuals. Ha d'oferir opcions per tal que la interfícies es pugui utilitzar de diverses formes, tan si l'usuari es esquerrà com dretà i que sigui el més adaptable possible al ritme de l'usuari.
3. **Ús Simple o intuïtiu**: El disseny ha de ser fàcil d'entendre independentment de l'experiència, els coneixements, les habilitats o el nivell de concentració de l'usuari. Cal dissenyar una interfície sense complexitat innecessària i que sigui molt intuïtiva i guiable per a l'usuari, independentment del seu grau de coneixement amb les tecnologies i la seva edat. Es pot proporcionar d'informació complementaria per tal de que sigui més guia, ja sigui ajudes, o avisos retroalimentats durant la tasca.
4. **Informació fàcil de percebre:** El disseny ha de ser capaç d'intercanviar informació amb l'usuari, independentment de les condicions ambientals o les capacitats sensorials del mateix. Per tal de que sigui fàcil de percebre, caldria utilitzar diferents mitjans, ja siguin sonors, vibracions, visuals i lumínics, per a que la presentació sigui més completa. La informació ha de ser llegible i els elements s'han de diferenciar entre si per a que es puguin percebre per si sols.
5. **Tolerància a l'error:** El disseny ha de minimitzar les acciones accidentals o fortuïtes que poden tenir conseqüències fatals o no desitjades. Per a que hi hagi aquesta tolerància, s'ha fer un ampli testeig del disseny, cal tenir en compte totes les màximes accions que pot arribat a fer un usuari per tenir en compte que no comportin errors. Així doncs, cal advertir dels errors de forma clara, que s'entenguin els errors i per a que s'han produït, fet que l'usuari al entendre-ho no repetiria certes accions per tal de no conduir al mateix error.
6. **Mínim esforç físic**: El disseny s'ha de poder utilitzar de forma còmoda, eficaç i amb el

mínim esforç possible. Cal que es pugui utilitzar minimitzant certs aspectes, com ara les accions repetitives, amb el mínim d'esforç físic constant, i en una posició neutral. En cas de que calgui un esforç, que la força utilitzada de la operació sigui raonable.

7. **Dimensions apropiades d'ús i aproximació:** Les mides i espais han de ser apropiats per l'abast, manipulació i ús per part de l'usuari independentment de la seva mida, posició i mobilitat. Proporcionar una interfície, on la mida dels elements siguin proporcionals a la mida del terminal. A més a més, que sigui lo suficientment grans com per a que siguin llegibles i de fàcil accés independentment de la mida dels dits dels usuaris o de la seva mobilitat.

L'objectiu principal d'aquesta filosofia és aconseguir una societat on totes les persones puguin participar, independentment de la seva condició, incloses les persones amb discapacitats (ja siguin limitació funcional visual, auditiva, cognitiva o de mobilitat), gent gran, de diferents grups de poblacions, etc...

Una manera de millorar el disseny universal és treballar directament amb un grup d'usuaris amb problemes d'accessibilitat, els quals et poden explicar quins problemes troben en els dissenys d'aplicacions i en l'avaluació de prototips realitzats.

D'altre banda, també cal sensibilitzar als dissenyadors. Per exemple el propi dissenyador pot simular certs problemes d'accessibilitat per veure en primera persona, quins problemes es pot trobar. Una manera és simular les limitacions de mobilitat, limitant els moviments amb objectes, embenatges, barres de ferro per immobilitzar articulacions, etc...

D'aquesta forma canviaria la manera de veure les conseqüències que poden tenir un mal disseny per als grups d'usuaris amb problemes d'accessibilitat.

En el propi context particular, s'ha trobat dificultats en el fet de que actualment, la gran majoria de telèfons mòbils amb *S.O*. *Android* instal·lat pateixen fragmentació, és a dir, hi ha molta diversitat de models**,** amb dispositius diferents, i cada companyia ha decidit els dissenys del seus aparells, amb o sense teclat, amb o sense botons i ni tan sols s'han posat d'acord en el nombre estàndard de botons a utilitzar.

Aquests detalls han portat a haver de cercar informació de tots o gairebé tots els dispositius mòbils amb *Android* per tal de poder concretar uns criteris perquè el disseny de la interfície fós el més universal possible.

A l'actualitat, s'ha pogut establir un nombre mínim de botons, que són**:**

- Els botons de Home, per minimitzar la *App*.
- El botó de Menú, per obtenir més opcions.
- El de *Back*, per tornar a la pantalla anterior.

Un altre factor important, ha sigut el fet de realitzar una aplicació de geo-localització. Aquest fet ha condicionat totalment en el moment de prendre les decisions més importants, en base a la realització de la interfície de guiatge en temps real i la selecció de destinació. El disposar d'un mapa i de la capacitat de representar informació sobre ell, juntament amb altres factors estètics, ha portat a dissenyar dues interfícies paral·leles i equivalents (principi [1]) per tal de poder treure el màxim rendiment a cada perfil.

La primera d'elles està destinada a persones que no necessiten cap tipus d'accessibilitat especial. Basant-se en el principi [2], es dóna a l'usuari la opció d'activar/desactivar la forma en que rebrà la informació, ja sigui per veu o per text i la possibilitat de seleccionar el seu mode d'entrada de dades per defecte, encara que un cop a la pantalla corresponent es pot triar un altre.

Aquesta interfície utilitza *layouts* (fitxer que conté la distribució dels elements en una pantalla) amb botons virtuals, taules per sintetitzar la informació de les rutes, texts aclaridors i funcionalitats bàsiques amb els botons del dispositiu. D'aquesta manera s'aconsegueix acomplir el principi [3].

La segona interfície està més adaptada per tal de poder proporcionar la millor experiència a l'usuari que tingui una necessitat d'accessibilitat, solucionant el problema de la localització dels botons i de la informació, substituint-los per una barra lateral (per dretans o esquerrans, segons la configuració) que s'encarregarà de fer saber a l'usuari a on es troba en cada moment.

S'ha afegit ajudes vibratòries, visuals i sonores per informar a l'usuari sobre l'estat del guiatge i que no es trobi desorientat [1].

Per tal de poder intercanviar informació de manera eficient (principi [4]), els integrants del mòdul d'interfície gràfica han trobat formes apropiades per transmetre informació, com s'ha comentat abans i per poder rebre informació per part de l'usuari, s'han fet servir mètodes de reconeixement de veu, teclat virtual, *gestures*, historial, etc...

Els mètodes emprats en les dues interfícies, s'han hagut de testejar per poder minimitzar les seves fallades, fent l'aplicació més robusta, basant-nos en el principi [5]. També hem hagut de tenir en compte les mides dels botons i dels texts [7] perquè siguin entenedors i no cal dir que pel fet de ser una aplicació mòbil, no cal fer cap esforç físic [6].

Per poder arribar al nombre màxim d'usuaris, hem implementat l'aplicació de manera que sigui multillenguatge, aprofitant les facilitats que ens dóna el SDK *d'Android* i pel moment podem oferir l'aplicació en 3 idiomes: anglès, castellà i català. No obstant, està preparada per incorporar altres llenguatges en un futur.

La gran demanda de telèfons amb aquest S.O. ha portat a les companyies a dissenyar aparells per tal de poder arribar al nombre màxim d'usuaris i això ha portat a fer una classificació general de 3 tipus en quant als preus:

Gamma baixa: ZTE Blade, Huawei Ivy, Huawei Sonic, etc…

Gamma mitjana: Samsung Galaxy SCL, HTC Desire, etc…

Gamma alta: Samsung Galaxy S2, LG Optimus 2x, HTC Sensation, HTC Desire HD, etc…

Aquesta diferenciació de gammes, també ha suposat una diferenciació en els components dels aparells i això ha portat per exemple, a haver de realitzar una interfície preparada pel màxim nombre de resolucions i densitats de píxels (*Figura 5*). Per aquesta raó hem fet servir com a unitats els dp (*density pixels*) per les mides dels layouts, i sp per les mides de les tipografies.

	Low density (120), *ldpi*	Medium density (160), *mdpi*	High density (240), *hdpi*
Small screen	• QVGA (240x320), 2.6"-3.0" diagonal		
Normal screen	• WQVGA (240x400), 3.2"-3.5" diagonal • FWQVGA (240x432), 3.5"-3.8" diagonal	• HVGA (320x480), 3.0"-3.5" diagonal	• WVGA (480x800), 3.3"-4.0" diagonal • FWVGA (480x854), 3.5"-4.0" diagonal
Large screen		• WVGA (480x800), 4.8"-5.5" diagonal • FWVGA (480x854), 5.0"-5.8" diagonal	

Figura 5 - Resolucions de pantalla i densitat de píxels

2.5 Anàlisi de terminals

2.5.1 BOTONS FÍSICS EN DISPOSITIUS ANDROID

Els terminals mòbils han evolucionat molt durant els últims anys. Els primers terminals tenien botons físics, els quals es podien localitzar amb facilitat sense mirar, cosa que feia que fossin fàcilment accessibles pels usuaris amb problemes d'accessibilitat visuals. Avui en dia la majoria d'aquests botons han passat a ser tàctils i per tant ja no els podem diferenciar com abans.

Per tal de veure aquesta clara tendència a la desaparició dels botons físics per botons tàctils, s'ha realitzat un petit estudi dels diversos terminals que hi ha per al sistema operatiu *Android*.

El primer terminal va aparèixer els 2008, l'anomenat *HTC Dream*. Aquest terminal compta amb un teclat *qwerty* físic amagat sota la pantalla, a més a més, té una bola de navegació (*trackball*) amb la tecla enter per seleccionar els elements de la pantalla situada a la part central inferior del terminal. Conté 5 botons físics, dos a cada costat del *trackball* i un a sobre d'ell. Aquests botons, ordenats d'esquerra a dreta són el trucada, *home*, menú, *back* i penjar trucada/encendre i apagar el terminal. A part d'aquests botons visibles en la cara de la pantalla, en els laterals s'hi troba les tecles de volum i el disparador de la càmera. Així doncs, es pot veure que inicialment els terminals anaven carregats de botons físics.

Com s'ha comentat en l'apartat anterior, el tema de la fragmentació s'ha tingut ben en compte en el procés de disseny de la interfície. S'han estudiat nombrosos models de telèfons mòbils i cercat documentació al respecte i s'ha pogut observar que actualment, només hi han 3 botons que predominen a tots els telèfons amb S.O. *Android*. Ni tan sols el botó de la càmera, o cercar han estat respectats en alguns models, dificultant les interfícies i les implementacions associades.

2.5.2 DIFERENCIACIÓ ENTRE BOTONS FÍSICS I TÀCTILS

S'ha realitzat un anàlisi sobre dispositius de diferents companyies i s'ha seleccionat uns models que destaquen sobre la resta en lo referent al seu disseny a nivell de botons fent la classificació en botons físics, tàctils i digitals creats per *S.O*.

2.5.3 DISPOSICIÓ DELS BOTONS

En els dispositius analitzats tot i que porten botons semblants en els dispositius que porten botons semblants, s'ha pogut veure que la disposició d'ells és diferent, augmentant la dificultat en la implementació i en les decisions de disseny. Per exemple, fent servir la interfície 2, s'ha vist que en molts dispositius, la barra de desplaçament coincidia amb la posició del botó *Back*, fent inestable el seu funcionament en cas d'un desplaçament erroni fora de la pantalla. Per aquesta raó es va decidir bloquejar el botó per *software*. De la mateixa manera, es va mirar de bloquejar la resta de botons que habitualment es situen sota la pantalla i es va trobar informació en la que es parlava de la impossibilitat de bloquejar el botó de *Home* en les versions actuals del S.O. per no posar a l'usuari en una situació en la que no pogués avortar una aplicació, encara que en possibles versions, potser no serà així. Al ser impossible el poder bloquejar totalment els botons, es va realitzar una doble vibració als extrems de la barra per tal de mantenir sempre orientat a l'usuari.

2.5.4 ANÀLISI

A continuació es pot veure una petita, però significativa, mostra de diferents tipus de dispositius mòbils amb *Android*. S'ha triat dispositius de les principals marques per ser lo més representatius possibles. Es posa especial èmfasi en la disposició dels botons físics i tàctils. En l'*Annex A* si pot trobar una taula amb una extensa mostra de dispositius mòbils, els quals han servit per a realitzar l'anàlisi complet.

2.5.4.1 Motorola

MILESTONE

Teclat	Analògic extern lliscant
Trackpad/Trackball	Trackpad analògic òptic
Botons tàctils	Back, Menú, Home, Cerca, Càmera
Botons físics	Volum
Extres	No

Taula 3 - Especificacions Motorla Milestone

2.5.4.2 HTC

MAGIC

Teclat	No
Trackpad/Trackball	Trackball
Botons tàctils	No
Botons físics	Back, Menú, Home, Cerca, Trucar, Penjar, Volum
Extres	No

Taula 4 - Especificacions HTM MAGIC

DESIRE Z

Teclat	Analògic extern lliscant
Trackpad/Trackball	Trackpad analògic òptic
Botons tàctils	Back, Menú, Home, Cerca
Botons físics	Volum, On/Off, Càmera
Extres	No

Taula 5 - Especificacions HTC DESIRE Z

CHACHACHA

Teclat	Analògic extern lliscant
Trackpad/Trackball	No
Botons tàctils	Menú, Home, Back, Cerca
Botons físics	Back, Facebook, volum
Extres	No

Taula 6 - Especificacions HTC CHACHACHA

2.5.4.3 SONY ERICSSON

XPERIA PLAY

Teclat	Analògic extern lliscant
Trackpad/Trackball	TouchPad analògic capacitiu
Botons tàctils	No
Botons físics	Back, Menú, Home, Cerca, Volum
Extres	Control Jocs

Taula 7 - Especificacions SONY XPERIA PLAY

2.5.4.4 SAMSUNG

GALAXY SII i9100

Teclat	No
Trackpad/Trackball	No
Botons tàctils	Back, Menú
Botons físics	On/Off, Volum, Home
Extres	No

Taula 8 - Especificacions SAMSUNG GALAXY SII

GALAXY SII amb teclat físic (variant SGH-I927)

Teclat	Analogic extern lliscant
Trackpad/Trackball	No
Botons tàctils	Back, Menú, Home, Cerca
Botons físics	On/Off, Volum
Extres	No

Taula 9 - Especificacions SMASUNG GALAXY SII (TF)

2.5.5 PROPERES TENDÈNCIES

S'ha cercat informació sobre futurs models de telèfons mòbils amb aquest *S.O*. i fins i tot es pot trobar algun futur model que no tindrà cap botó físic. Per part d' *Android*, també s'ha vist la mateixa tendència. En properes versions del Sistema Operatiu, la totalitat dels botons físics seran opcionals, deixant la decisió en mans dels fabricants. Possiblement aquesta decisió s'ha degut a l'objectiu de fusionar les versions 2.3 per *Smartphones* i *Honeycomb*, per *tablets*.

S'hauria d'estudiar si aquesta tendència continuarà en augment o no, per tal de decidir si caldria afegir com a futures millores la remodelació de la interfície.

Un altre punt important, conseqüència de la desaparició dels botons físics, és la inutilització de l'actual revisor de pantalla d'*Android*, el *Talkback*, el qual el seu funcionament és donar una descripció dels elements de la pantalla per els quals es van seleccionant. Aquesta acció es pot a dur a terme, si el terminal mòbil conté un *Trackball* o *Trackpad*. Com es pot veure, si la tendència dels terminals és que no continguin aquests elements, deixa inaccessible el terminal per part d'aquest grup d'usuaris i per tant, aquests hauran de buscar terminals on hi hagin el màxim nombre de botons físics.

2.6 Conclusions estudi de viabilitat

Un cop realitzat l'estudi de viabilitat del projecte es pot decidir la seva viabilitat. Si es decideix que no és viable s'haurà de buscar alternatives i realitzar un nou estudi des de l'inici d'altra banda si és viable es durà a terme.

En primer lloc el que s'ha realitzat és un estudi de les aplicacions de guiatge i d'ús dels transport públics que es poden trobar en el mercat i s'ha determinat que la aplicació cobreix certes mancances front a les aplicacions existents. Les grans aportacions de l'aplicació és l'accessibilitat que ofereix amb diferents tipus d'entrada de dades i un alt nivell de personalització de l'aplicació sense influir en la seva senzillesa.

D'altra banda s'ha realitzat un estudi dels costos que suposa el desenvolupament del projecte i es conclou que són assumibles donat que els costos materials son mínims i el cost més elevat és el cost per persona/hora en el desenvolupament de l'aplicació.

La planificació que s'ha realitzat dóna un calendari raonable encara que el marge per a possibles endarreriments és força ajustat.

Finalment i tenint en compte tots els estudis realitzat es decideix que el projecte és viable i es procedeix a la seva realització.

3 Mòdul de la Interfície d'Usuari

Tal i com s'ha indicat en l'apartat Estudi de viabilitat, el projecte s'ha dividit, entre els recursos disponibles, en 4 subgrups. Així doncs, a partir d'aquest capítol es tractarà sobre el Mòdul de la Interfície d'Usuari, el qual està format pel subgrup 2 format per Eric Lara i Monica Esteve.

Amb aquest capítol, es pretén fer una petita introducció al mòdul indicant quines són les eines en les quals es pot treballar amb Android per a desenvolupar la interfície. També es vol informar de quins objectes es fan servir dins d'aquest mòdul, per tal que es tingui una visió global de l'entorn del mòdul.

En la resta de capítols, seguit d'aquest, es reflectirà la feina duta a terme per a cada integrant del mòdul.

3.1 Introducció del Mòdul

Per poder elaborar amb èxit una aplicació mòbil d'aquestes característiques, cal donar importància al disseny i a la implementació de la interfície d'usuari. L'objectiu que s'ha marcat és realitzar un programa per a tothom, sent lo més polivalent possible. Per aquesta raó, en primer lloc és indispensable detectar quins són els nostres potencials usuaris, per tenir en compte els seus trets diferenciadors. Com que la nostra aplicació és d'un caire molt obert en aquest sentit, tenim un rang molt ampli d'edats d'usuaris, i això ens porta cap a la implementació d'una interfície basada en les directrius del *El disseny* universal.

3.2 Components de la GUI D'Android

En aquesta secció, es farà un breu repàs als components més utilitzats en l'aplicació i es definiran els seus atributs principals

Per tal de poder realitzar correctament totes les funcions de l'aplicació mòbil, ha calgut dedicar bona part del temps per conèixer i dominar els diferents tipus d'elements que composen la interfície d'usuari de les aplicacions *Android*.

Com es veurà més endavant, la unitat bàsica d'una aplicació per aquesta plataforma és l'*Activity*. Es podria dir que és l'equivalent a una pantalla d'usuari en una aplicació convencional, a on es pot posar els diferents tipus de *widgets* (elements de la *UI* que componen la pantalla) per interactuar amb l'usuari.

Cada *Activity* té un *layout* (esquema que defineix la ordenació dels elements en pantalla) associat, que conté *ViewGroups* i *Views*. Hi ha dues maneres per definir aquestes interfícies, de forma estàtica, mitjançant arxius XML o dinàmica, mitjançant codi Java dins l'*Activity*.

Les *Views* utilitzen els recursos, com ara cadenes de text, colors, estils, que estan compilats en format binari. La classe *R.java* es genera automàticament per *Android* i es la referència a recursos i actua com a pont entre el codi que sol·licita el recurs i les referències binaries.

En el *Diagrama 1* es pot veure la relació entre aquests tres elements.

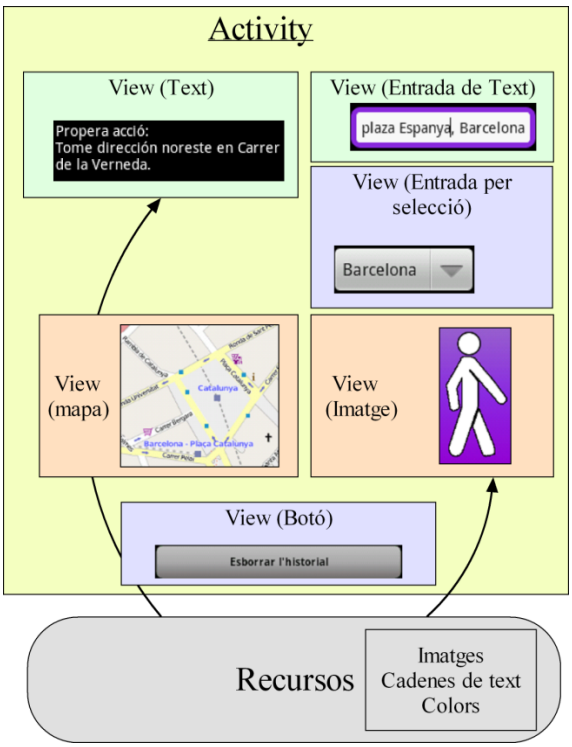


Diagrama 1 - Relació entre Activity - view - recursos

A continuació es farà un senzill repàs a les principals *Views* y *ViewGroups* utilitzades per la realització d'aquest projecte.

3.2.1 VIEWGROUPS I VIEWS

Les *Views* són uns tipus de *widgets* de la interfície gràfica d'una aplicació d'*Android*, ja que les activitats les content. A més a més les *Views* representen elements de la pantalla i s'encarreguen d'interactuar amb l'usuari a través dels events que proporciona el propi *Android*.

Una *Viewgroup* és una agrupació de *Views*, i en si mateixa, també es considera una altra *View*.

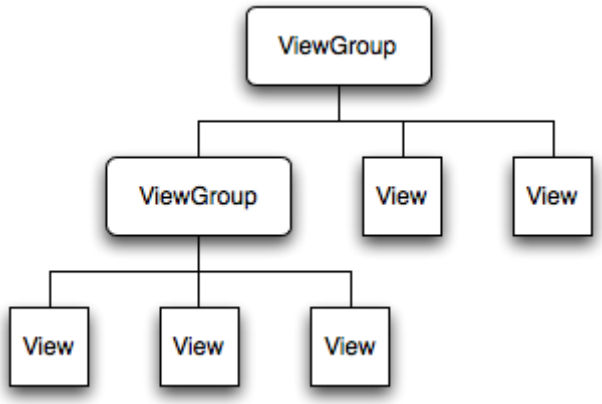


Diagrama 2 - Jerarquia de les View

Així doncs, les *Views* són un punt molt important per aquest mòdul, mitjançant les quals s'ha intentat dissenyar la interfície de l'aplicació seguint el disseny universal.

Cada *Activity* conté un arbre jeràrquic d'elements *View*, el qual els seus elements adopten diferents formes i mides.

En el *Diagrama 2* es pot veure un diagrama amb la jerarquia de *Views*, les quals s'agrupen per *ViewGroups*.

En el següent exemple (*Figura 1* i *Codi 1*), es pot veure el mètode d'inicialització de les *Activities*, a través del mètode *setContentView*, s'associa l'arxiu *main.xml* que es troba dins el directori */res/layout/* del mòdul de la interfície gràfica

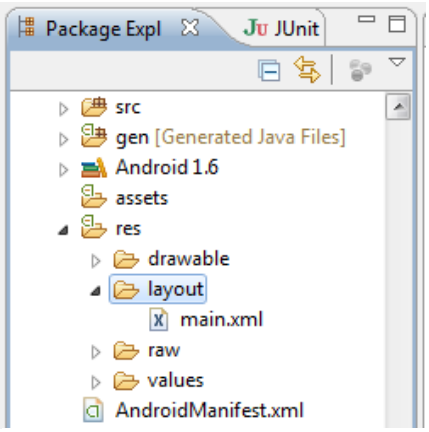


Figura 6 - Sistema de Carpetes per els Layouts

```
@Override
    public void onCreate(Bundle
savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.main);
 }
```

Codi 1 - Associació del layout amb l'activity

Tant *Views* com *ViewGroups* tenen atributs comuns que són de molta importància, com el *id* (a l'XML es defineix amb android: *id = "@+id/nom_id"*), que serveix per identificar i modificar els seus valors de manera dinàmica, el *layout_height* i el *layout_width* que com els seus noms indiquen, permeten modificar les seves mides.

Un altre atribut de gran importància es el *visibility*, que ens permet mostrar, ocultar o fer desaparèixer (sense ocupar res) una vista.

Per tal de realitzar les tasques del mòdul de la Interfície d'usuari, s'han utilitzat els dos mètodes d'incorporar les *View* a l'activitat de manera conjunta. O sigui, s'ha realitzat el *layout.xml* i s'ha associat amb l'activitat. Seguidament s'han realitzat canvis sobre els elements de l'activitat dinàmicament en el codi. Aquesta acció s'ha dut a terme així, degut a que depenent de la interacció de l'usuari amb l'aplicació requeria realitzar canvis de colors, mida,... de forma dinàmica.

Un *layout* està estructurat per una sèrie de *tags*(etiquetes), els quals indiquen que és un element *View* de la pantalla. D'aquest element se li incorporen una sèrie de propietats per a

definir-lo, com per exemple, la posició, el color, la mida, si és clicable, etc... En el *Codi 2* es pot observar el format que té, on el *LinearLayout*, *TextView*, *Button* són elements *View* incorporats de forma jeràrquica.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
              android:layout_width="fill_parent"
              android:layout_height="fill_parent"
              android:orientation="vertical" >
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hola, sóc TextView" />
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hola, sóc Button" />
</LinearLayout>
```

Codi 2 - Exemple de format d'un Layout

Per tal de tenir una visió més detallada dels elements que component les *Views* i *ViewGroups*, s'ha adjuntat en l'annex un apartat amb descripció dels components més utilitzats en la *GUI* de l'aplicació.

A més de tots els elements que s'han comentat en l'annexa, també són necessaris altre tipus d'elements com són els menús, les notificacions, etc... però per qüestió d'espai, no s'ampliarà en aquesta ocasió.

3.2.2 ACTIVITY

Tal i com s'ha introduït, una activitat representa una de les pantalles d'una aplicació. Aquesta té un cicle de vida des de que s'inicia fins que es tanca l'aplicació, determinat per el sistema *Android*, per tal de poder garantir flexibilitat amb els processos del terminal.

El cicle de vida d'una activitat són funcions que s'executen dins el codi de l'activitat, les quals representant diversos estats en que pot estar la pantalla. En el cas d'*Android*, en fa servir 7 estats. Per tal de veure quins són, a continuació es mostren els estats indicats i la seva funcionalitat i l'estat d'interacció amb l'usuari així com el seu *Diagrama 3.*

- ***onCreate():*** Una vegada s'inicia l'aplicació, s'executa aquesta funció, la qual contindrà el codi per tal d'inicialitzar la pantalla amb les *View* corresponents. També dona accés a qualsevol estat emmagatzemat prèviament en forma de *Bundle*.
- ***onStart():*** S'invoca quan l'activitat està visible en pantalla per a l'usuari.
- ***onPause():*** Aquesta funció s'utilitza al parar l'activitat al passar en un segon pla. La funció guarda l'estat de l'activitat per quan es vulgui continuar amb l'activitat en el estat en que es va parar.
- ***onResume():*** Funció la qual es fa servir sempre, ja que s¡invoca quan l'activitat comença a interactuar amb l'usuari, tan si ve del *onCreate*, com si ve del *onRestart*.
- ***onStop():*** S'activa quan l'activitat que esta en segon pla, fa molta estona que ho esta i es vol alliberar els recursos que utilitza o bé quan es vol passar a parar l'activitat. En aquest cas, l'activitat passa a un estat d'invisibilitat per a anar als següents cicles de vida.

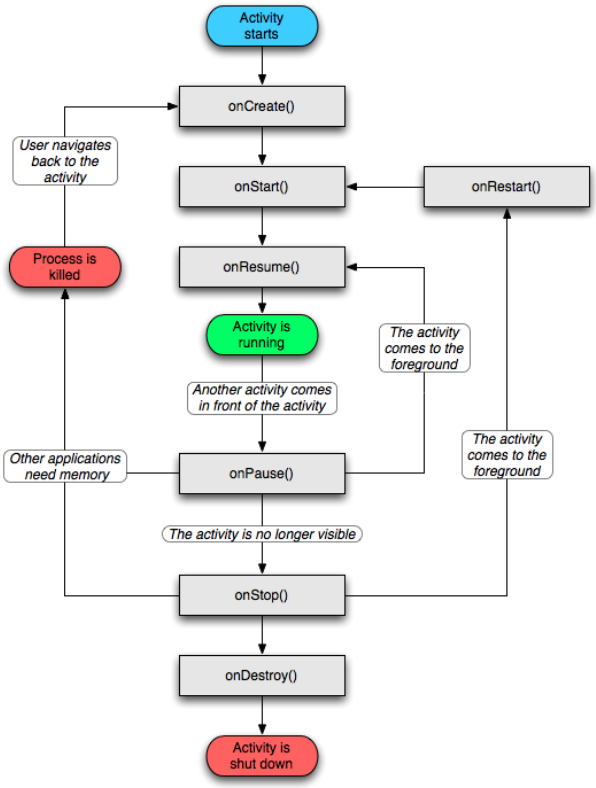


Diagrama 3 - Diagrama del cicle de vida d'una Activity

- ***onRestart():*** Serveix per a tornar reiniciar l'activitat i no haver de inicialitzar-la de nou. Això es possible si l'activitat encara està en la pila d'activitats.
- ***onDestroy():*** Funció per a eliminar l'activitat en el cas que se li hagi indicat o bé si el sistema decideix parar-la per alliberar recursos.

Així doncs, a l'hora de implementar el codi de totes les pantalles de l'aplicació, s'ha de tenir en compte que per a una correcta implementació, s'hauria de posar les funcions més importants del cicle de vida depenent de la funcionalitat de l'*Activity*.

3.3 POJOs (Plain Old Java Objects)

Definició i descripció dels POJOs fets servir a la interfície gràfica.

Aquesta denominació va aparèixer cap al Setembre de l'any 2000, durant la preparació d'una conferència on es parlaven dels beneficis de la codificació de la lògica de negocis en

objectes senzills regulars, en comptes de fer servir *beans* (dependents de *frameworks* especials que segueixen els estàndards EJB anteriors al 3.0).

No va ser pas una nova tecnologia sinó un nom nou per revaloritzar la típica programació orientada a objectes (Rebecca Parsons, MacKenzie Josh i Martin Fowler).

S'ha fet ús d'aquest tipus d'objectes per tal de poder enviar, rebre informació, o d'emmagatzemar utilitats i constants de manera adient. A continuació es podrà veure una breu explicació dels més importants que s'han fet servir al mòdul de la interfície gràfica, però més endavant s'explicaran al seu context particular.

PREFERENCE

Al seu interior s'hi pot trobar totes les preferències de l'usuari per la utilització de l'aplicació, com poden ser: el seu *theme* (tema, aparença) utilitzat, si necessita accessibilitat o no, si es dretà o esquerrà, si vol missatges d'informació addicionals, ... En el *Diagrama 4* es poden veure els seus atributs.

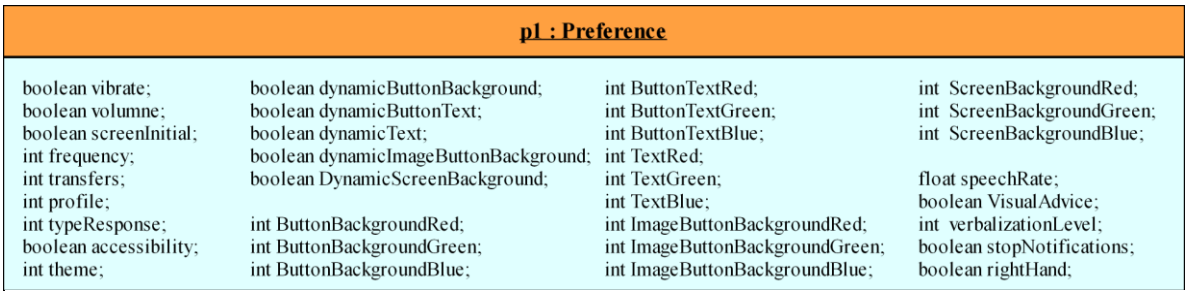


Diagrama 4 - Atributs del objecte Preference

ROW

Aquesta classe inicialment es va crear per tal de poder començar les *Activities* d'informació i selecció de mapa quan els mòduls que havien de donar la informació encara estaven implementant la solució. Afortunadament, quan aquesta solució va estar implementada, es va poder re aprofitar, per tal de mostrar la informació desglossada dels temps de viatge, adaptada a l'usuari, de manera més entenedora. En el *Diagrama 5* es poden veure els seus atributs.

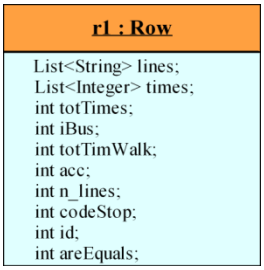


Diagrama 5 - Atributs del objecte Row

MENUROWPRINCIPAL

Per tal de poder mostrar les *ListView* amb un contingut més complex per cada fila, que no pas una simple cadena de text, cal tenir

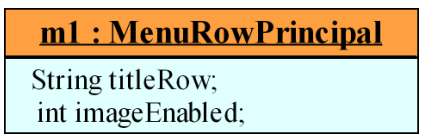


Diagrama 6 - Atributs del objecte MenuRowPrincipal

una llista d'objectes. Aquesta llista d'objectes contenen la informació de cada fila del *ListView*. Per tal de poder personalitzar la llista del menú principal, s'ha utilitzat aquest objecte el qual conté una imatge i una cadena de text. En el *Diagrama 6* es poden veure els seus atributs.

GUIDANCE

Per poder rebre tota la informació referent al guiatge, es va haver de dissenyar unes classes, lo més senzilles possibles. Aquesta classe en particular, només conté un identificador de tipus *int*, que serà heretat per els objectes de les seves classes derivades. S'ha implementat el seu *Parcelable (i a les classes derivades també)* degut a que *Android* ho demanava per tal de poder ser enviat dins dels paquets als *Intents*.

Classes derivades:

Depenent del tipus de guiatge que es necessiti en cada moment, es va decidir dissenyar 3 tipus de *Guidance* ben diferenciats i un altre tipus, per poder representar una brúixola. En el *Diagrama 7* es poden veure els seus atributs.

GUIDANCE_WLK

Aquesta classe porta tota la informació necessària per orientar a l'usuari en el moment del guiatge a peu, com pot ser informació sobre els metres que falten per girar, indicacions de la acció actual i de la propera, etc...

GUIDANCE_WAIT

Els objectes d'aquesta classe porten la informació referent al moment en que l'usuari es troba a la parada de l'autobús esperant que arribi el proper. Conté els minuts que falten per arribar, el codi de la parada, la línia, les característiques del bus que arribarà, etc...

GUIDANCE_BUS

Conté la informació important per el moment en el que l'usuari es troba a l'interior de l'autobús, com el nom de la parada, si es la darrera o no, etc...

GUIDANCE_COMPASS

Aquesta classe ha de permetre orientar a l'usuari, tant en la interfície *blind* com en la no accessible visualment, per aquesta raó, conté els graus a girar, direcció, sentit, etc...

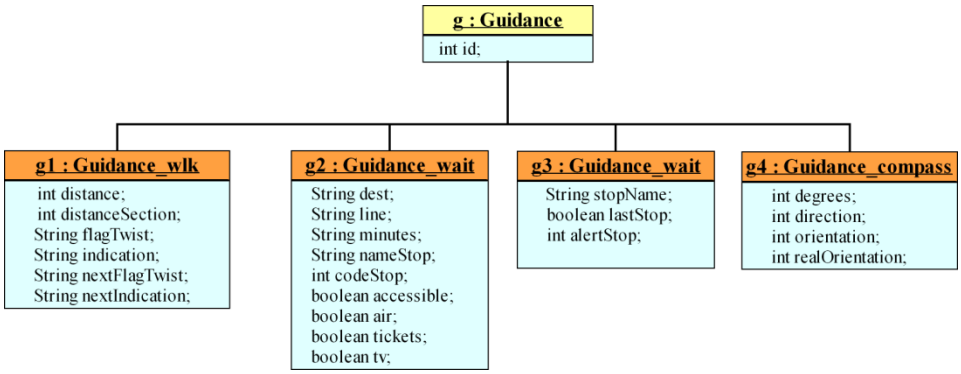


Diagrama 7 - Atributs dels objectes Guidances

FAVORITS

Per tal de poder guardar la informació d'una localització del recorregut en la base de dades de Favorits, s'ha creat un objecte que conté la informació la qual es requereix per aquesta funcionalitat. En el *Diagrama 8* es poden veure els seus atributs

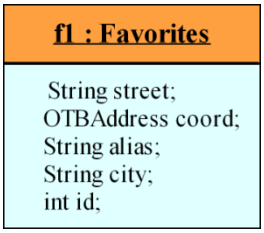


Diagrama 8 - Atributs dels

CONTACTES

Una altre manera de guardar la informació d'una localització del recorregut és fer servir l'agenda. En aquest cas, la informació que es pot incorporar en ella es molt gran i per això s'ha dissenyat un objecte el qual conté la informació més rellevant a guardar i recuperar. En el *Diagrama 9* es poden veure els seus atributs.

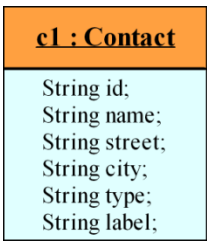


Diagrama 9 - Atributs dels objecte Contact

4 Disseny d'Interfícies

En primer lloc, ha sigut necessari identificar els nostres possibles usuaris per tal de realitzar una llista d'especificacions de la interfície gràfica, per poder cobrir, de la manera més satisfactòria, les seves preferències. Per realitzar aquesta tasca, s'han elaborat enquestes per obtenir una visió general. També s'ha cercat informació sobre les característiques de la gent que requereix d'una accessibilitat especial, rebent informació d'organitzacions com la O.N.C.E. [1](Organización Nacional de Ciegos de España) i consultat altres associacions com la P.T.P. (Promoció del Transport Públic)[2], sense oblidar pàgines web com Transport.cat i s'han decidit quines eren les combinacions de colors més adients a utilitzar, tenint en conte els diferents tipus de daltonisme, per poder arribar a tothom.

Un cop recopilada tota aquesta informació, els objectius principals de la interfície s'han classificat en aquests 4:

- Interfície amigable
- Accessibilitat
- Possibilitat de triar diferents rutes
- Guiatge pas a pas

Amb totes les dades obtingudes, s'ha decidit iniciar la primera fase de prototipatge. En aquesta fase, s'han fet servir aquestes aplicacions:

- *Microsoft Visio*
- *Adobe Fireworks*
- *Adobe Photoshop + Gimp*

I investigat quins són els mètodes habituals per aconseguir resultats lo més reals possibles. Amb l'ajuda de plantilles d'interfícies *Android*, s'han pogut dissenyar els primers prototips de la interfície, i després d'una avaluació en conjunt i de descartar alguns, s'ha triat el més adient, aplicant-li algunes millores indicades pels tutors.

A continuació es pot veure alguna de les plantilles utilitzades durant una de les fases de l'evolució de les propostes i finalment, la resultant:

[1] http://www.once.es/

[2] http://transportpublic.org/

4.1 Plantilla utilitzada

Aquesta va ser la plantilla principal, però es van agafar elements d'altres plantilles complementàries.

Plantilla per dissenyar interfícies Android

4.2 Proposta d'interfície inicial (versió número 2)

4.2.1 PANTALLA PRINCIPAL:

Càrrega de l'aplicació i pantalla inicial

4.2.2 VERSIÓ SENSE ACCESSIBILITAT ESPECIAL:

Possible recorregut del menú fins arribar al guiatge

4.2.3 VERSIÓ ACCESSIBLE:

Proposta de la interfície en el perfil amb accessibilitat

Es pot apreciar com en aquestes versions s'havien de millorar certs aspectes com la manera d'estructurar la informació, el disseny de la pantalla de *Gestures*, la forma de donar-li la informació a l'usuari en la pantalla de selecció de la ruta.

Sense oblidar la importància de definir altres com el flux que tindrien els menús per tal de poder configurar adequadament totes les característiques que es volia que tingués l'aplicació però sense perdre la senzillesa i la facilitat d'ús.

A més d'aquest tipus de prototip, també es va parlar de la possibilitat d'afegir perfils personalitzats a l'aplicació, però després es va descartar degut a que es va considerar que els telèfons són unipersonals i que no cal afegir dificultats a la navegació. Finalment es va arribar a la conclusió que calia diferenciar entre 5 tipus de perfils genèrics, que vindrien amb les opcions preseleccionades, per tal de guanyar rapidesa i estalviar esforç a l'iniciar l'aplicació:

- **Perfil sense cap tipus d'accessibilitat:**

  Aquest perfil ve sense cap opció especial activada, amb el mode d'introducció de dades mitjançant teclat seleccionat per defecte. Finalment es va decidir deixar activada la vibració, per fer més entenedor algun event d'alguna de les pantalles.
- **Perfils amb accessibilitat:**
  - **Visual**: era necessari crear una interfície especial per tal de mantenir les mateixes opcions que als altres perfils. Per defecte es deixaria el só i la vibració

activats i s'utilitzarien combinacions de colors especials, per afavorir la visualització en el cas de que les dificultats visuals no arribessin a la ceguesa.

- **Cognitiva**: en aquest cas es tracta que l'usuari estigui orientat en tot moment, principalment utilitzi les destinacions emmagatzemades als seus Favorits o Agenda i si ha d'introduir-ho ell, que sigui per reconeixement de veu. En aquest cas, la vibració del telèfon vindria desactivada per defecte, per tal de no distreure'l.
- **Auditiva**: aquest perfil és semblant a l'estàndard però degut a la impossibilitat de transmetre informació per veu, es mostren certs missatges segons el context de l'usuari en la aplicació.
- **De Mobilitat**: en aquest apartat, el mòdul de cerca només retorna rutes accessibles (dins de lo possible, perquè encara no es pot disposar de rutes a peu accessibles)

Perfils disponibles a l'aplicació

4.3 Interfície resultant

Després d'afegir les millores:

Successió de pantalles del prototip finalista

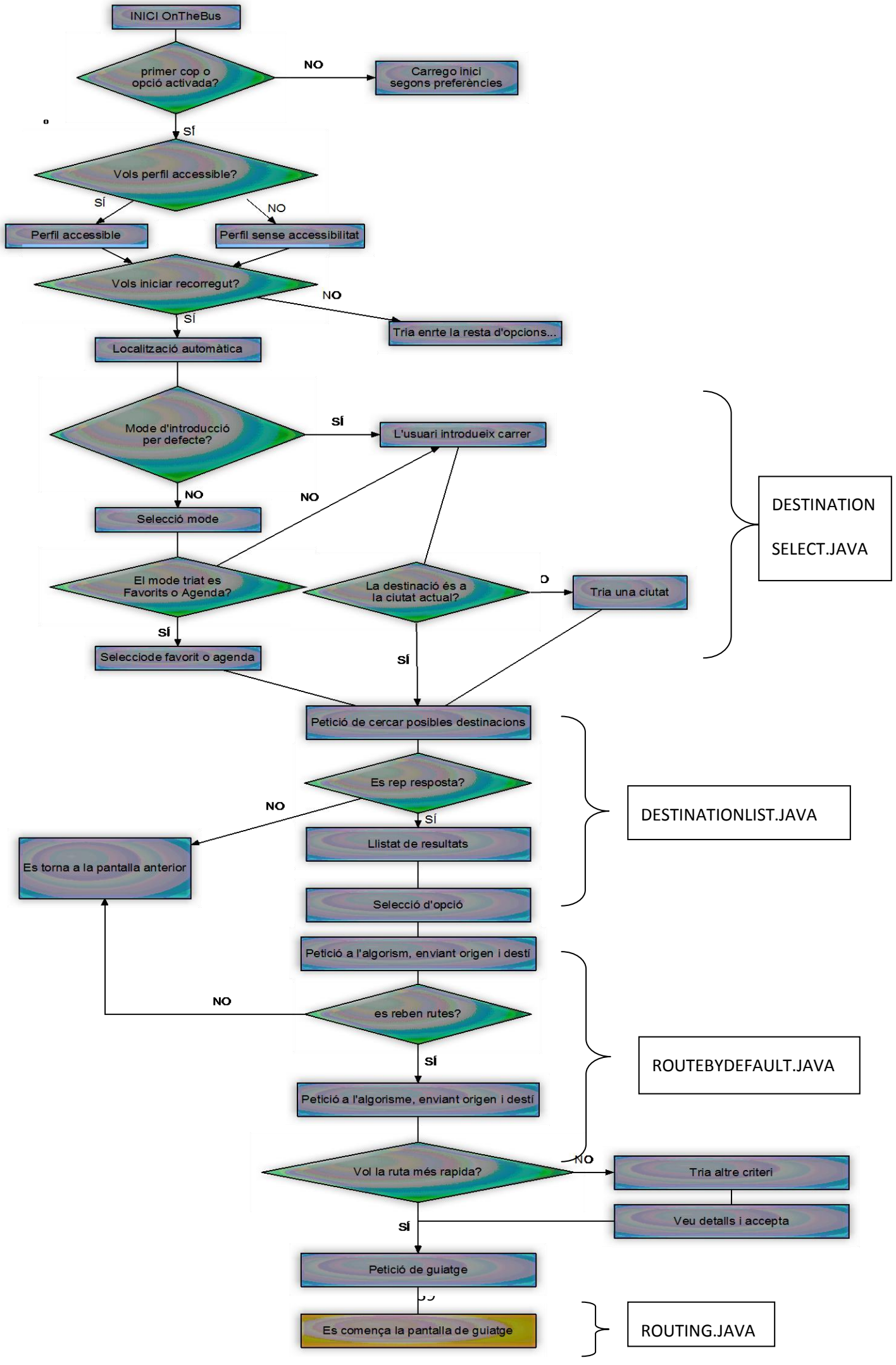
Es pot apreciar objectivament com s'havia millorat en diferents aspectes, sobretot en lo que respecta a l'arquitectura de la informació a representar i en l'accessibilitat de la interfície, encara que faltava definir una resposta efectiva per millorar la navegació per part de gent que requeria d'accessibilitat visual.

Per altra banda, també faltava familiaritzar-se físicament amb els tipus d'elements que formen part de la interfície gràfica en els menús habituals del S.O. *Android*, perquè no era suficient amb les plantilles, degut a que segons quin element de la pantalla, només es pot fer servir en situacions determinades o normalment s'utilitza sota uns criteris preestablerts.

Per aquesta raó, es va adquirir amb la major brevetat que va ser possible, un terminal mòbil amb *Android* versió 2.1 (*Huawei Ivy*), per la qual es va decidir inicialment implementar l'aplicació, i ens vam instal·lar *l'SDK d'Android* per tal de provar l'emulador a *l'Eclipse*. Posteriorment, es van adquirir més dispositius *Android* que ens van permetre realitzar proves reals de comportament de l'aplicació amb diferents característiques tècniques i diferents versions del S.O.

Un cop realitzats els canvis a la interfície resultant, ja es disposava d'una bona guia per seguir, encara que més endavant sorgissin modificacions a l'hora d'implementar la solució per detalls en la implementació.

Després d'això, s'ha destinat més temps a l'anàlisi dels elements existents en l'*SDK d'Android ,* tant a nivell gràfic, com de mòduls d'accessibilitat, per dissenyar una bona interfície gràfica. L'objectiu ha sigut transmetre i representant tota la informació recopilada del prototipus finalista afegint-li tot lo aprés en el moment de buscar informació sobre les característiques de/ El disseny universal.


INICI OnTheBus
primer cop o opció activada?
NO
Carrego inici segons preferències
SÍ
Vols perfil accessible?
SÍ
NO
Perfil accessible
Perfil sense accessibilitat
Vols iniciar recorregut?
NO
SÍ
Tria enrte la resta d'opcions...
Localització automàtica
Mode d'introducció per defecte?
SÍ
L'usuari introdueix carrer
NO
NO
Selecció mode
El mode triat es Favorits o Agenda?
La destinació és a la ciutat actual?
Tria una ciutat
SÍ
Selecciode favorit o agenda
SÍ
DESTINATION SELECT.JAVA
Petició de cercar posibles destinacions
Es rep resposta?
NO
SÍ
Es torna a la pantalla anterior
Llistat de resultats
Selecció d'opció
DESTINATIONLIST.JAVA
Petició a l'algorism, enviant origen i destí
NO
es reben rutes?
SÍ
Petició a l'algorisme, enviant origen i destí
Vol la ruta més rapida?
NO
Tria altre criteri
Veu detalls i accepta
SÍ
Petició de guiatge
Es comença la pantalla de guiatge
ROUTEBYDEFAULT.JAVA
ROUTING.JAVA

5 Mòduls d'interacció amb l'usuari

S'ha fet una recopilació d'informació de cada mòdul que ha participat en la tasca d'intercanviar informació amb l'usuari

5.1 Gestures

Aquesta *API* va ser introduïda a la versió 1.6 d'*Android*, degut a la popularitat que van anar adquirint els gestos tàctils a la pantalla dels dispositius portàtils durant la època de les *Palm Pilot* amb el seu S.O. PalmOS que portava l'alfabet *Graffiti* i fa poc, amb el *plugin* de *Firefox* (*All-in-One Gestures*[3]) que consisteix en *Gestures* de moviments de ratolí. En realitat, els Gestures, són molt més antics, sembla ser que *Xerox* manté una patent d'un alfabet fet sense aixecar el dit que posseeix des de 1997. Per aquesta raó, va enviar a judici a Palm [4] i temps després, Palm ha patentat una versió *multi-stroke* (aixecant el dit) d'un altre alfabet propi.

Alfabet de Xerox

Existeixen diferents tipus de Gestures predefinits, com per exemple:

Tap / Double Tap

Drag

Fling

Slide

Pinch

[3] https://addons.mozilla.org/es-es/firefox/addon/all-in-one-gestures/

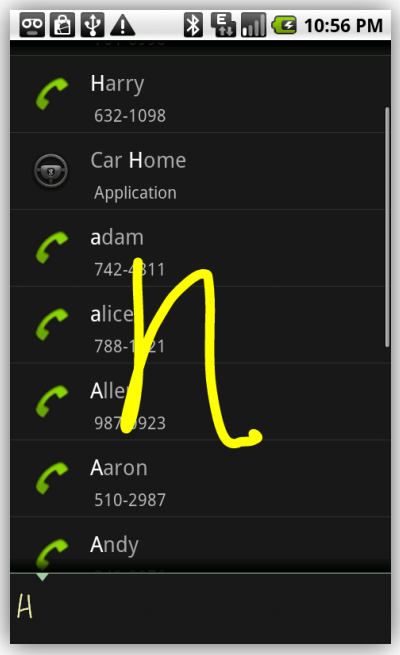
[4] http://sandbox.parc.xerox.com/parctab/csl9501/node4.html, Patente USPTO n.º 5596656

Els quals es poden associar a diferents accions depenent de l'aplicació, però per la nostra aplicació en concret, no ens era suficient amb això, necessitàvem alguna manera d'introduir lletres i que fossin reconegudes, mitjançant aquesta tecnologia.

Es va estar cercant informació al respecte, i a l'*SDK d'Android*[5] es va trobar que hi havia una referència a la classe *LetterRecognizer* encara que, es va trobar un *report* a http://code.google.com parlant de que es tractava d'un "*bug*" a la documentació[6], perquè en realitat no existeix actualment. Encara que diuen que hi serà en futures releases.

> ***"Class Overview***
>
> ***(…) A user-defined gesture can be recognized by a GestureLibrary and a built-in alphabet gesture can be recognized by a LetterRecognizer."***

Per aquesta raó, ha sigut necessari cercar formes alternatives com alfabets predefinits, i actualment, es pot dir que encara no hi ha cap projecte interessant per resoldre aquest problema. No obstant, al 2010 *Google* va publicar la seva *App Google Search*[7], disponible per versions d'*Android* 1.6 i superiors, per tal de suplir les seves mancances amb el reconeixement de veu, ja que en aquell moment, només reconeixia cerques en anglès.



Google Search

Aquesta *App* és, com el seu nom indica, una aplicació per facilitar les cerques al telèfon. Permet escriure a qualsevol lloc de la pantalla i a mesura que s'introdueixen els caràcters mitjançant *Gestures*, realitza el filtratge de resultats, tant de contactes de l'agenda, com d'altres aplicacions instal·lades, cançons de música o altres tipus d'elements.

Després d'haver testejat la *App*, es va veure que el seu promig d'èxit era molt elevat, independentment del tipus de lletra que es fes servir per escriure i la seva velocitat de resposta també era molt bona. A continuació, es van buscar exemples d'implementacions que fessin servir la *Api* de *Google Search* per tal de poder adaptar el seu reconeixement de caràcters a la nostra aplicació, ja que

[5] http://developer.android.com/reference/android/gesture/Gesture.html

[6] http://code.google.com/p/android/issues/detail?id=16637

[7] http://gesturesearch.googlelabs.com/

la resta de la App no era pas accessible; es va trobar un exemple bastant representatiu, que va ajudar a descartar aquesta opció, ja que per poder aprofitar aquesta *API* ens veiem limitats a enviar una petició (*Intent*), en forma de caixa negra, sense tenir accés directe al reconeixement de caràcters i sense poder fer modificacions a la interfície de cerca.

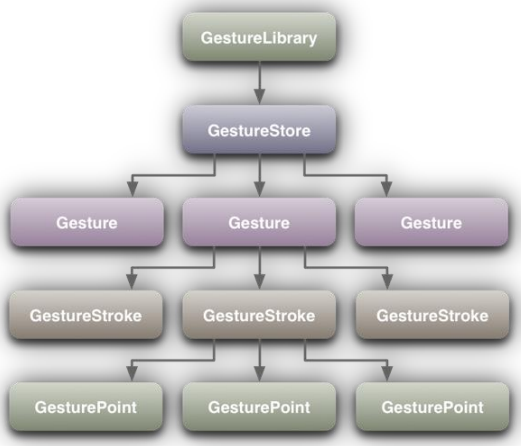
Posteriorment, s'han explorat altres alternatives per tal d'utilitzar aquesta *API*, com per exemple la *JSON/Atom Custom Search API*[8], que permet desenvolupar *websites* i aplicacions per obtenir resultats equivalents a la *App* comentada anteriorment, però finalment s'ha desestimat la seva utilització, ja que l'ús d'aquesta API només permet 100 cerques diàries de manera gratuïta.

Elements que composen Gestures:

- Llibreries *Gestures*
- *GestureOverlayView*
- Prediccions

Llibreries Gestures

Després d'aquests resultats, es va començar a experimentar amb els exemples més genèrics, però amb possibilitat de personalització, per poder dissenyar una llibreria fàcil de memoritzar, que reconegués lletres i números amb una bona taxa d'encert .



Estructura de la llibreria

A la documentació de l'*SDK d'Android* hi han principalment, dos mètodes que es fan servir per carregar les llibreries d'aquest tipus. El primer d'ells es el *fromRawResource* i com el seu nom indica, permet carregar les llibreries que es trobin dins de la carpeta /res/raw del projecte, és a dir, només en mode lectura.

```
mLibrary = GestureLibraries.fromRawResource(this, R.raw.spells);
if (!mLibrary.load()) {
    finish();
}
```

[8] http://code.google.com/intl/it/apis/customsearch/v1/overview.html
http://stackoverflow.com/questions/2499324/google-search-api-for-android-systems

Per altra banda, el mètode *fromFile* ha permès l'elaboració de la llibreria, i la posterior edició dels caràcters per tal de millorar la seva resposta en el moment del reconeixement.

```
private final File mStoreFile = new File(
        Environment.getExternalStorageDirectory(),GestureConstants.GESTURES_LIBRARY_AZ);
```

```
if (sStore == null) {
    // al hacerlo de esta manera en vez de frawFromResource,
    // podemos leer y escrbir en el archivo, recuperándolo posteriormente
    // Ruta: sdcard/gestures
    sStore = GestureLibraries.fromFile(mStoreFile);
}
```

GestureOverlayView

Aquest component hereta de *ViewGroup* i de *FrameLayout,* es pot utilitzar a dins d'altres *layouts* o fer-lo servir com *layout* pare, incloent altres *Views* filles. En la pràctica, és una capa transparent que es fa servir com la capa superior, per tal de poder realitzar traços *Gestures* a sobre. Un cop s'ha fet la vinculació amb ella al *onCreate* de *l'Activity*, se li ha d'afegir algun/s dels *listeners* que aniran monitoritzant el seu estat.

OnGesturePerformedListener: es fa servir per classificar el *Gesture* realitzat, segons la puntuació de la predicció.

OnGestureListener: és el més complet, perquè permet capturar el moment d'inici i de finalització del *Gestures*, el monitoritza mentre es realitza el traç i en cas de cancel·lació també.

OnGesturingListener: aquest *listener* només permet rebre dades de l'inici i el final del *Gesture.*

GestureOverlayView

Com a paràmetres interessants, comentar que a *l'XML* es pot definir si volem admetre Gestures *single-stroke* / *multi-stroke.* A OnTheBus s'ha fet servir la segona opció encara que finalment l'alfabet consta només de caràcters d'un sol traç, perquè tenen una millor resposta en aquesta modalitat[9]. També hi ha un altre paràmetre que ens permet definir la orientació de les *Views* que es troben al seu interior, per si utilitzem un GestureOverlay contenint un ListView (amb l'estil de *Google Search*) que no es confongui el *Scroll* d'aquesta View amb un

[9] http://stackoverflow.com/questions/6380908/android-gestures-single-stroke-multi-stroke-produce-inconsistent-results

Gestures. Per aquesta mateixa raó, no es pot fer servir un *Gesture* que sigui una línia vertical senzilla, degut a que no ho reconeix com a tal. Si s'hagués fet servir una altra orientació, si que hauria funcionat, però llavors els símbols per esborrar el darrer caràcter i per inserir un espai, no.

Prediccions

Com més endavant s'explicarà (apartat *GesturesTime*), s'ha adaptat un exemple de l'*SDK d'Android* per fer el testeig i veure la validesa dels símbols creats de manera ràpida.

Quan el *listener* que té associat (*onGesturePerformed*) es dispara, tracta de reconèixer el traç introduït i per cada element de la llibreria, elabora una puntuació, ordenada de més alta a més baixa.

```
public void onGesturePerformed(GestureOverlayView overlay, Gesture gesture) {
    ArrayList<prediction> predictions = mLibrary.recognize(gesture);

    // We want at least one prediction
    if (predictions.size() > 0) {
        Prediction prediction = predictions.get(0);
        // We want at least some confidence in the result
        if (prediction.score > 1.0) {
            // Show the spell
            Toast.makeText(this, prediction.name, Toast.LENGTH_SHORT).show();
        }
    }
}
```

Codi per obtenir la primera predicció

5.1.1 ALFABET GESTURES

Degut a la quantitat de col·lisions que sorgeixen amb facilitat, s'ha decidit realitzar un conjunt tancat de Gestures perquè l'usuari no els pugui modificar, deixant com a futura millora aquesta funcionalitat.

Inicialment, es va voler realitzar una llibreria de *Gestures* que fós lo més extensa possible, per tal de poder reconèixer els caràcters escrits amb diferents tipus de números, lletres, majúscules i minúscules, però ràpidament, després de fer unes proves, es va arribar a la conclusió de que hi havia masses col·lisions, com per exemple:

- la "i" minúscula i el número "1" o "7"

- el "0" i la lletra "o" majúscula

- el número "9" i la lletra "g" minúscula, etc...

Això va portar a la determinació de dividir l'alfabet en dues llibreries, una per els números i altra per les lletres, per aquesta raó, es va confeccionar aquests nous alfabets, que incloïen modes d'edició de text, com moure el cursor cap a l'esquerra una posició, cap a la dreta, anar al principi o final de text, esborrar el darrer caràcter introduït, suprimir el següent o canviar entre mode text i mode numèric.

Encara que el reconeixement havia millorat molt, sobretot en la llibreria numèrica, després de fer testeigs, per part de diferents persones, es va arribar a la conclusió que encara hi havia bastants col·lisions a millorar en la llibreria de text, com:

- "a" - "o"
- "B" - "R"
- "a" - "d"
- "e" - "l"

Altre dels caràcters a millorar era el de canvi de mode, que tenia col·lisions amb la "J". Més factors determinants en el reconeixement dels caràcters van ser la mida i tipus de pantalla utilitzat per el dispositiu mòbil, on es va apreciar que a les pantalles resistives, com la del telèfon *Huawei*, el tipus de traçat reconegut era diferent que no pas el de les pantalles capacitives. Per aquesta tasca, l'emulador no era de gran ajuda, perquè passa bastant sovint que es queda alentit i la tasca de reconeixement, depèn de la fluïdesa del dispositiu en aquell moment, variant molt la seva resposta amb el mateix caràcter d'entrada.

Després de fer més modificacions per intentar adaptar les lletres de manera que es poguessin reconèixer tal i com les coneixem i degut a la dificultat d'elaborar un alfabet sense col·lisions, es va arribar a la determinació de simplificar i remodelar l'alfabet, encara que distés una mica del tradicional, per tal de millorar substancialment els resultats.

Es van eliminar la majoria de símbols d'edició, considerant que el fet d'haver d'aprendre Gestures nous, afegia massa complexitat a la navegació de l'usuari, però es va mantenir el símbol per realitzar l'espai, el d'eliminar el darrer caràcter introduït i el de canviar de mode text/numèric, això si, simplificant la seva realització i fent més fàcil la seva memorització. Uns pocs caràcters es van modificar prenent com a base l'alfabet *Graffiti* de *Palm* i es va reduir a 1 símbol per caràcter, triant en cada cas, el símbol més diferenciador de la resta, segons la lletra majúscula o minúscula associada, per no col·lapsar el nombre de *pattern-matching* del mòdul Gestures.

Els únics *Gestures* que són diferents de les tipografies de l'alfabet tradicional són els corresponents a les lletres f, k, ñ, t, x, però s'ha aconseguit millorar totalment els resultats en els reconeixements i els temps de resposta.

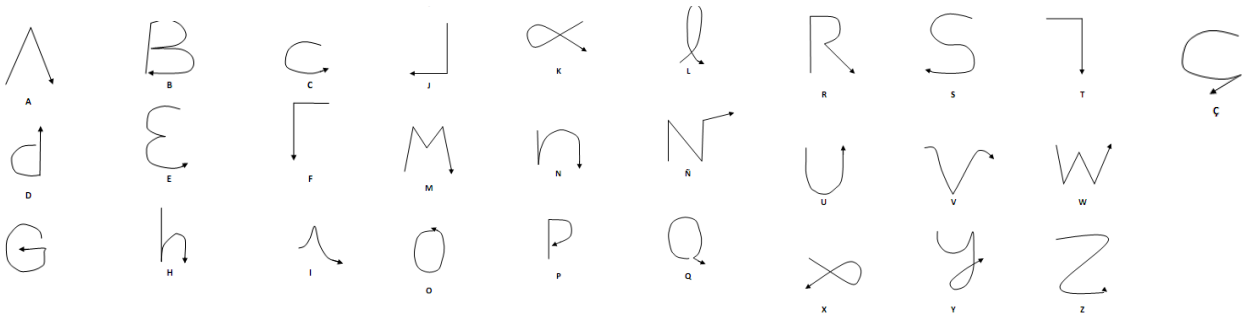
5.1.2 GESTURES D'EDICIÓ

Esborrar darrer caràcter

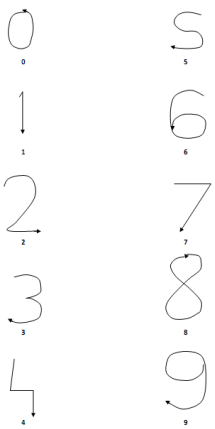
Espai

Canvi de mode text/numèric

5.1.3 MODE TEXT



5.1.4 MODE NUMÈRIC



5.2 Vibració

Encara que no ho sembla, en les interfícies d'aquesta aplicació, ha tingut bastant importància el fet d'haver pogut afegir prestacions hàptiques[10] (relacionades amb les sensacions tàctils). Li ha donat una millor expressivitat en cada context i això ha millorat

[10] http://www.isfh.org/
http://en.wikipedia.org/wiki/Haptic_technology

definitivament la experiència d'usuari i la facilitat d'ús de la interfície per gent amb necessitats d'accessibilitat visuals, principalment. Ha permès, juntament amb el mòdul TTS, l'enviament d'informació a l'usuari, sense necessitat de mirar la pantalla.

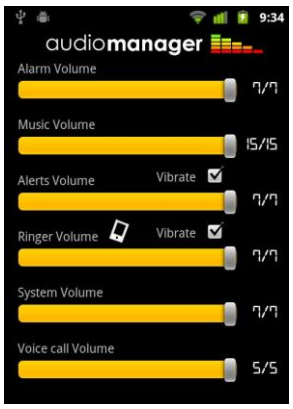
A *l'SDK d'Android*, podem veure que només hi han 4 mètodes per aquesta classe:

- ***cancel()*** **:** s'encarrega com el seu nom diu, de finalitzar la vibració.
- ***hasVibrator()*** **:** serveix per confirmar que el dispositiu consta amb un mòdul hardware capaç de produir vibració.
- ***vibrate(long tim)*** **:** engega l'element vibrador durant els mil·lisegons que se li indiquin per paràmetre.
- ***vibrate(long[] pat, int rep)*** **:** aquesta opció és bastant interessant ja que ens permet fer que el dispositiu vibri seguint el patró que se li enviï (un array de mil·lisegons). Amb el segon paràmetre se li especifica si es vol fer només un cop o si es vol repetir la vibració fins que es premi la tecla de *Back*.

No s'ha d'oblidar que per poder tenir accés a aquests mètodes, abans s'ha d'especificar el permís corresponent al Manifest d'Android.

5.3 Tipus de fluxos d'àudio als dispositius Android

Al *S.O. Android* cada flux d'àudio ve associat a un tipus de flux, definits a l'*AudioManager*. Hi han 6 tipus de fluxos diferents, la majoria té 7 nivells de volum, però són independents:



AudioManager

Alarm**:** per gestionar el flux del despertador.

Music**:** defineix el volum de l'apartat multimèdia del telèfon. De manera especial, té 15 nivells de só, per poder obtenir més volum a l'hora de reproduir-lo.

Alerts**:** s'encarrega de tot tipus de notificacions.

Ringer**:** és el volum de la melodia associada a les trucades.

System**:** controla diversitat de sons del dispositiu, com el volum del disparador de la càmera o els sons que es reprodueixen a l'introduir un número de telèfon.

Voice call**:** aquest flux s'encarrega de la veu durant les trucades. Només té 5 nivells a diferència del altres.

Inicialment, es va associar el flux de dades de l'aplicació al flux del sistema, però després de fer proves es va comprovar que no era suficientment fort com per escoltar les indicacions al carrer, així que finalment ha quedat associat al volum de la música.

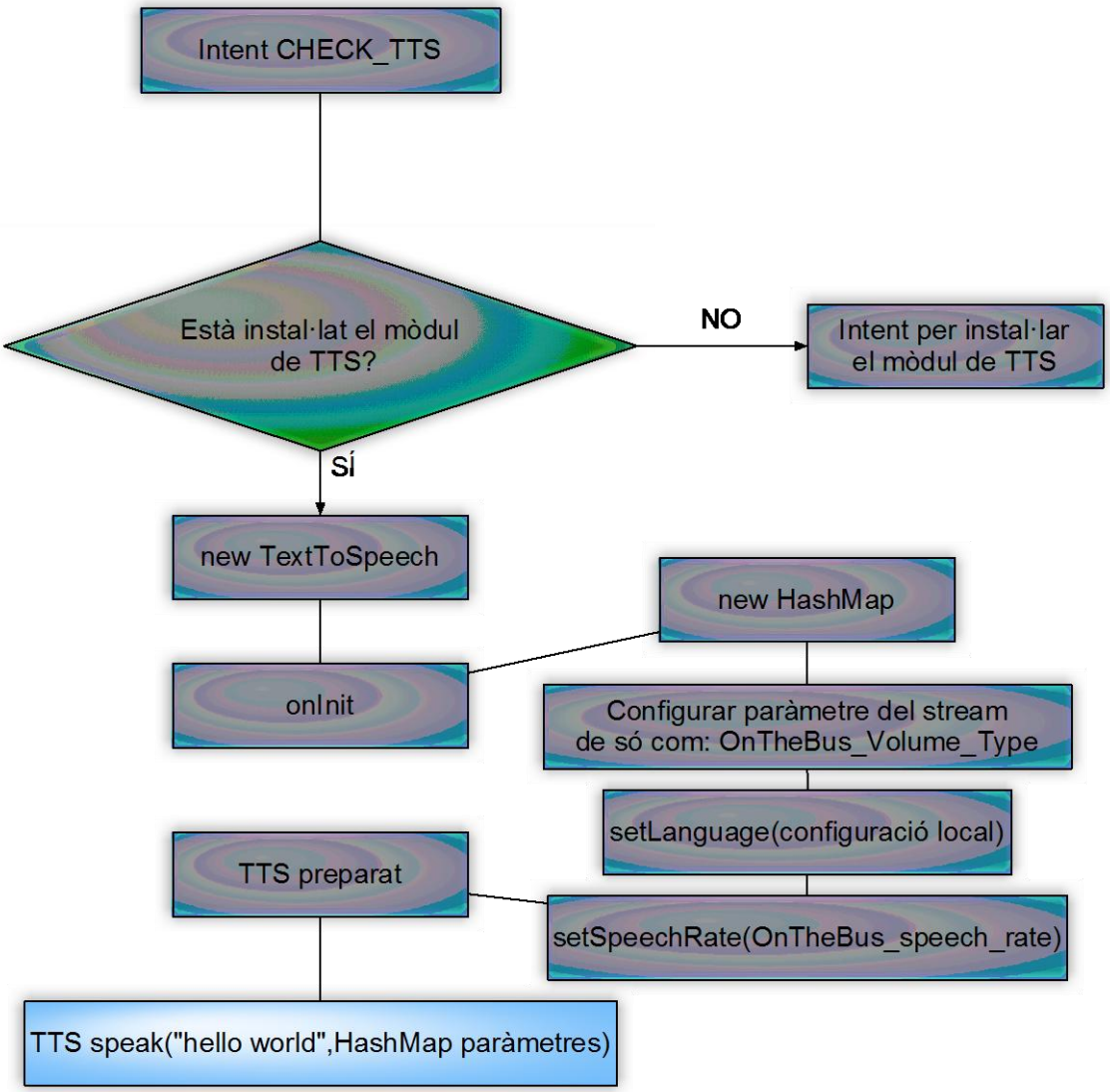
5.4 TTS (Text-to-Speech)

Una de les avantatges *d'Android* és el gran potencial dels mòduls de que es disposen. Un dels més populars és el *TTS*, que va arribar a partir de la versió 1.6 *d'Android*, el qual proporciona una veu sintetitzada capaç de llegir de manera dinàmica cadenes de text.

Abans de poder utilitzar-lo, s'ha de configurar mínimament l'objecte *TTS*. Entre les opcions que podem definir, es troba el llenguatge en que haurà de funcionar perquè pugui pronunciar amb l'accent adequat o la velocitat de reproducció. Una altra de les opcions que s'ha de definir és el tipus de flux de dades que farà servir i ens serà útil per poder controlar el seu volum posteriorment.

Es va buscar informació per tal de veure els accents disponibles i es va trobar que actualment la llista d'accents és bastant limitada: anglès (amb els accents americà i britànic), francès, alemany, italià i espanyol, però encara no existeix suport per accent català.

L'*engine* de veu gestiona una cua global de totes les peticions de parla (*utterances*). Per aquesta raó, hi han dos modes de parla, *QUEUE_ADD* on s'afegeix a la cua les peticions i *QUEUE_FLUSH* on s'allibera la llista de peticions de sintetització.

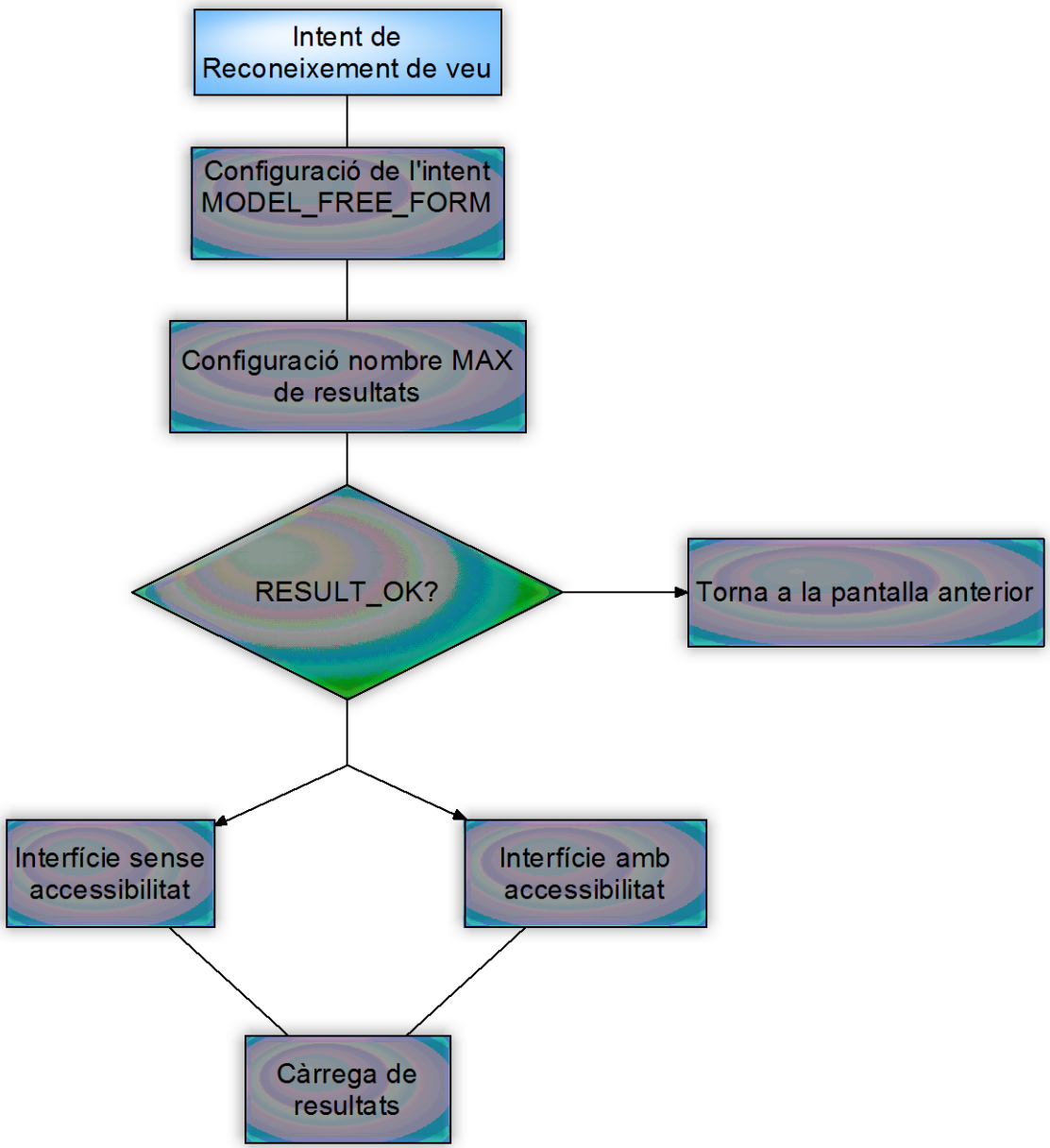


Funcionament TTS

5.5 STT (Speech-to-Text)

El fet de poder incorporar de manera senzilla funcionalitats de reconeixement de veu, ha millorat substancialment la fluïdesa per introduir la destinació, segons el perfil de l'usuari. Aquest revolucionari mode d'entrada d'informació va aparèixer a la versió 2.1 *d'Android*.

El procés general de funcionament el podem apreciar en el següent diagrama:



Estructura TTS

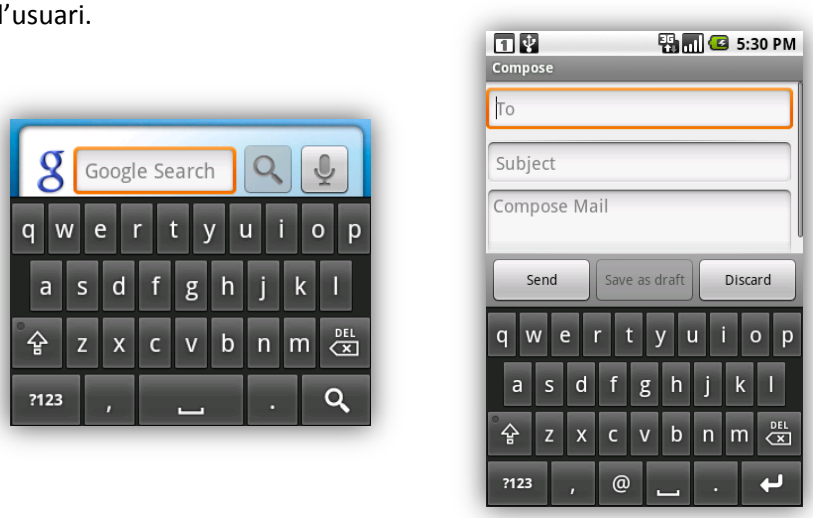
S'envia un intent, en el que es defineix el mode de reconeixement:

- *Language_model_form*: és la forma estàndard de reconeixement de veu.
- *Web_search*: realitza una cerca web amb el resultat obtingut, encara que es pot capturar la resposta per tal de fer una altra acció.

A més, també s'afegeix el nombre màxim de resultats possibles. La acció demanada pot retornar diferents codis de resposta, segons la situació que hagi succeït (error a la xarxa, no s'han obtingut resultats, ha sigut cancel·lat, resultat correcte, etc...). En el millor dels cassos, ens retorna un *ArrayList d'Strings*, que contindrà les possibles respostes, ordenades de més a menys puntuació.

5.6 Teclat

Segons el context en el que l'usuari hagi d'introduir text, *Android* pot configurar el teclat, que es posiciona a la zona inferior de la pantalla, en diferents modes, que milloren l'experiència d'usuari.



Modalitats de teclat

En aquest parell d'exemples, podem veure el mode de cerca i el mode estàndard de teclat, apreciant que en el primer d'ells, hi ha una lupa, per finalitzar la introducció de text. En canvi en el segon hi ha la tecla *enter* per fer salt de línia i permetre per exemple, la redacció de correus electrònics. A la nostra aplicació, hem fet servir el mode de cerca, per finalitzar la introducció del carrer de destí i tornar a la pantalla de la App.

5.7 Selecció des de Mapa

Com el seu nom indica, aquest mètode mostra una vista de mapa a la pantalla i l'usuari només ha de prémer sobre el carrer al que vol arribar. Es va haver d'implementar el *longTap* perquè no venia de sèrie.

6 ACCESSIBILITAT

Procés de disseny de la interfície accessible que hem fet server a la nostra aplicació per tal de donar funcionalitats equivalents a tots els perfils

6.1 Interfície Blind

Inicialment es va pensar en realitzar un disseny inicial únic que servís per totes les interfícies, amb 2 botons grans per pantalla, ubicats a les cantonades inferiors, o quatre botons, ubicats a cada cantonada, però després de fer un estudi de viabilitat, es va acabar descartant, per limitar massa la quantitat d'opcions per pantalla, agreujant la profunditat dels menús o limitant massa la aplicació i per altra banda, mostrant dades visuals a gent que depenent del perfil, podia necessitar accessibilitat visual.

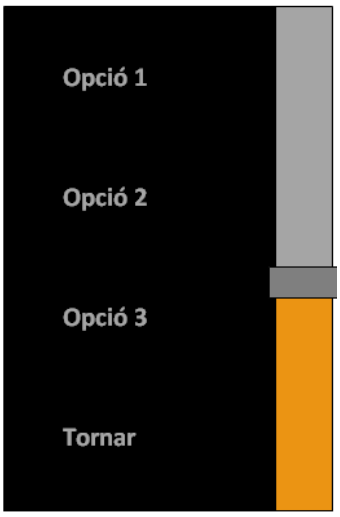
Pantalla d'opcions de viatge

Altra de les raons va ser la necessitat d'utilitzar mapa sempre que sigues possible, sent inútil en algun dels perfils.

Per aquestes raons es va decidir de crear una interfície alternativa per tal de cobrir les necessitats de tots els perfils, permetent-los funcionalitats equivalents, sense discriminar a cap col·lectiu, amb l'objectiu de mantenir les bases del disseny universal.

Amb aquest objectiu, es van consultar temes d'accessibilitat i altres projectes finals de carrera amb temàtiques d'accessibilitat i es van obtenir idees interessants que hem pogut aplicar per poder realitzar la nostra interfície accessible.

Amb la solució que s'ha fet servir a la nostra aplicació, es poden presentar fins a 8 o 9 opcions com a màxim per pantalla, sent més que suficient per la majoria de situacions. La nostra



Disseny Blind

implementació final també es dinàmica, només cal que rebi una llista d'opcions, de tipus *String*, i associar una acció determinada a cada opció.

La idea general es tractava d'una barra vertical a la dreta de la pantalla per on la qual el nostre usuari pogués desplaçar el dit a través de ella i obtenint la informació a mesura que la anés recorrent, tot això sense la necessitat d'haver d'utilitzar la visió.

Per tal objectiu es va haver d'adaptar el codi d'un component de la GUI *d'Android*, la barra de cerca (*SeekBar*), per tal de que fós vertical.

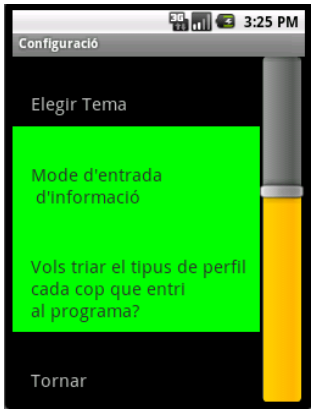
Un cop realitzada aquesta feina, lo següent va ser encarregar-se d'una correcta repartició de l'espai vertical per tal que les opcions s'expandissin per tota la pantalla de manera proporcional.

La primera idea va ser utilitzar un *View* anomenat *ListView* (per més informació sobre les Views, consultar Annex D), ja que aparentment bé preparada per rebre informació i repartir-la per la pantalla d'una manera semblant a la que interessava en aquest cas però va ser descartada aquesta opció degut a la impossibilitat de saber la mida total de cada fila i de modificar la seva alçada abans de mostrar-la per pantalla per tal de repartir les opcions de manera uniforme a la pantalla, a més de que per defecte és *Scrollable*, és a dir, s'expandeix quan la quantitat d'elements que conté és superior als que caben per pantalla, tenint en compte que la mida de la fila és fixa en aquell moment i apareix una barra de desplaçament invisible.

Problema amb ListView

Com a següent opció es va provar d'utilitzar com a layout un *TableLayout*, insertant a cada fila una sola columna amb un *TextView*, obtenint resultats molt millor, en el sentit que permetia modificar l'alçada de cada filera de manera dinàmica abans de dibuixar-se per pantalla. Els problemes que van sorgir llavors van ser dos:

1. Alguns textos ocupaven més d'una línia i això provocava que a la barra vertical, els píxels que s'havien dedicat per cada fila no fossin constants. Com a resultat negatiu, podíem desplaçar el selector de la barra vertical per tal de seleccionar una opció i



Problema múltiples línies

abans d'arribar, ja s'havia seleccionat perquè la *SeekBar* es pensava que cada opció era d'una fila.

2. Apareixien retallats els missatges segons la pantalla del telèfon on s'executés la aplicació.

TextViews retallats

Una primera aproximació per solucionar aquests problemes va ser forçar a que cada opció ocupés una fila, limitant la quantitat de lletres per opció i en el cas de que ocupés més, doncs es finalitzava el text amb "..." però si seleccionàvem la opció, el TTS pronunciava la informació sencera que pertanyia a aquesta fila.

Finalment, es va trobar una solució per tots aquests inconvenients, que va ser la utilització d'un *marquee*, semblant al concepte utilitzat a HTML, que permet utilitzar textos llargs per les opcions, encara que per pantalla només es mostra lo que cap, i amb l'avantatge que si es deixa la opció seleccionada, el text es va desplaçant verticalment de manera automàtica, permetent llegir tot el seu contingut sense cap complicació. Per la utilització d'aquesta funcionalitat, ha calgut eliminar la separació per columnes, que en aquest cas, al tractar-se de només un *TextView* per fila, no hi ha hagut problemes.

6.2 Millores d'accessibilitat aplicades a la interfície

A la solució explicada en l'apartat anterior se li han afegit aquestes millores per tal de fer-ho encara més òptim:

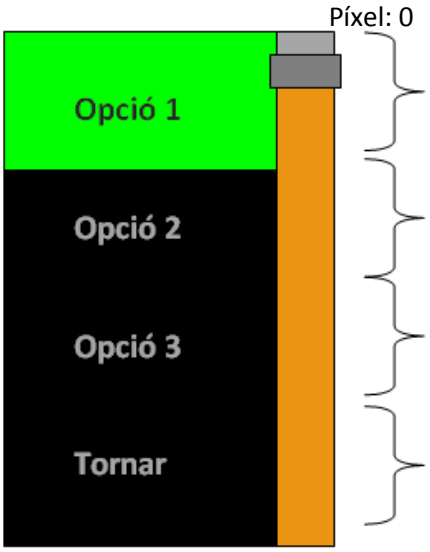
a. **Sistema d'Ajudes:** S'ha incorporat un sistema de reforç de la informació per tal que l'usuari no es pugui desorientar amb facilitat en cap dels menús. Quan l'usuari mou el selector de la *SeekBar* pot escoltar mitjançant el *TTS*, la opció que té seleccionada en cada moment, però si després d'escoltar la informació, encara li queda algun dubte, només ha de continuar prement el selector i la ajuda associada a aquella opció sonarà. Per el moment, les ajudes estan configurades per disparar-se als 4 segons d'estar premut, però si posteriorment es veiés que no és el temps més adequat, es pot canviar sense dificultats.
b. **Vibració** al desplaçar-se i canviar d'opció: per enriquir la informació proporcionada a l'usuari. Cada cop que es produeix un canvi d'opció degut al desplaçament del selector, es produeix una vibració molt lleu però significativa.

c. **Doble vibració:** per suggeriment dels tutors, s'ha acabat afegint una doble vibració quan el selector arriba a la primera o última opció de la pantalla.

d. Executar una opció fent **doble pulsació** al mig de la pantalla en comptes d'una sola.

e. Opció de **Tornar al final** sempre, per donar seguretat a l'usuari i que sàpiga que sempre pot cancel·lar.

f. Possibilitat de canviar de banda la barra vertical a les preferències, sent accessible per dretans i esquerrans.



Repartició de píxels

PÍXELS QUE PERTANYEN A CADA OPCIÓ:

Opció 1: De 0 a max/nº elements*1

Opció 2: De max/nº elements*1 a max/nº elements*2

Opció 3: De max/nº elements*2 a max/nº elements*3

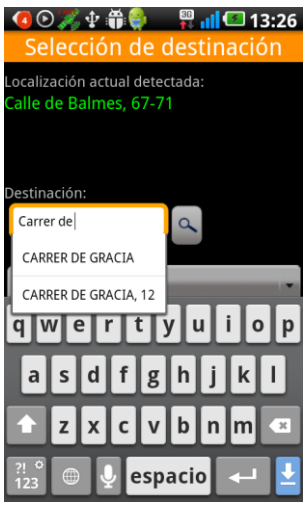
Tornar: De max/nº elements*3 a max

6.3 Millores en l'acceleració de la navegació

Com a principals millores en la navegació es pot nombrar principalment les que van relacionades amb la ràpida selecció de la destinació:

- Utilització **d'historial** a la interfície accessible per facilitar la feina a l'usuari i que hagi d'escriure el mínim possible. D'aquesta manera, sempre pot seleccionar molt ràpidament qualsevol dels últims destins que hagi triat.
- **Autocompletar**: és la versió anàloga a l'historial però a la interfície sense accessibilitat. En el mode per teclat, a mesura que se'n va introduint el carrer, apareix l'autocompletar, que es recupera des de la BD d'historial de cerques[11].
- **Favorits**: S'ha implementat la funcionalitat de poder salvar una posició determinada desde qualsevol lloc del guiatge per tal de poder tornar-hi en un altre moment.
- **Agenda**: També ha semblat oportú el fet de poder associar una destinació a un contacte de la teva agenda personal del telèfon.

[11] Mòdul de Dades

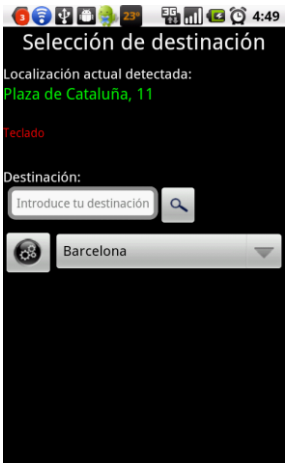


Favorits, Agenda i Autocompletar

7 Activities I

En aquesta secció es farà un breu repàs d'un conjunt de les classes que s'han fet servir a l'aplicació i del seu funcionament intern.

7.1 DestinationSelect



Canvi de mode de resposta

Aquesta *Activity* és una de les més importants, juntament amb la de guiatge. S'encarrega de mostrar la ubicació actual a l'usuari i permetre-li introduir el carrer i el número per tal de poder obtenir de l'algorisme de cerca, la millor ruta, a la *Activity RoutebyDefault*.

Per defecte, la pantalla consulta les preferències per saber quin és el mode favorit de resposta de l'usuari. Per si de cas, se li dóna la oportunitat de canviar ocasionalment aquest mode per un dels altres, però si torna a engegar la aplicació, es tornarà a agafar el de preferències. Entre les opcions disponibles, pot triar entre dir la seva destinació per veu, escriure mitjançant el teclat d'Android o els Gestures i si finalment, no coneix el nom del carrer però si que coneix la seva ubicació al mapa, pot seleccionar el punt allà directament, sempre que es trobi dintre

del domini de l'aplicació. En els tres primers modes, ha de seleccionar la ciutat on vol associar el carrer que ha introduït. Per defecte, la ciutat és l'actual.

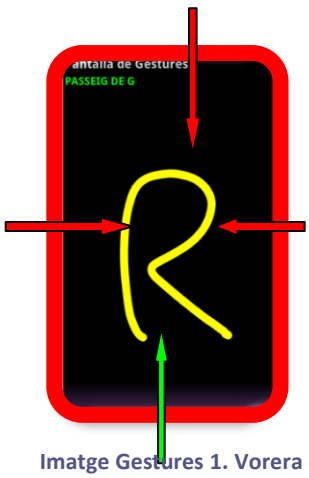
En el mode d'introducció de text per teclat, a més d'aprofitar l'historial a l'autocompletar, també estan afegits els noms dels tipus de vies.

7.2 GesturesTime

Pantalla de *Gestures* amb les opcións de la tecla menú

Aquesta és la pàgina de reconeixement de Gestures. Està realitzada de manera que li arriba la destinació anterior, que pot estar buida, i et permet afegir lletres amb espais o números, segons el mode que faci servir l'usuari en cada moment.

Un problema bastant comú quan s'han fet els testeigs amb el perfil d'accessibilitat visual, era que l'usuari, al no poder veure les dimensions del dispositiu exactes, contínuament s'estigués sortint de la pantalla, amb la corresponent fallada en el reconeixement del *Gestures*, que si la traçada no és continua, empitjora molt els resultats. Per aquesta raó, es va investigar un mètode que venia inclòs a la documentació *d'Android*, anomenat *getEdgeFlags*(), que teòricament et retorna, al finalitzar el *Gestures,* informació automàtica per saber des de quina banda ha ocorregut la fallada. Aquest mètode s'ha provat i no funciona de la manera esperada. Després de buscar més informació al respecte, es va trobar un llibre [12] on s'explicava que aquest mètode no té res implementat, que deixen el mètode *setEdgeFlags*(...) per tal de que ho programi el desenvolupador. A continuació, s'ha implementat els mètodes necessaris i afegit una vorera imaginària

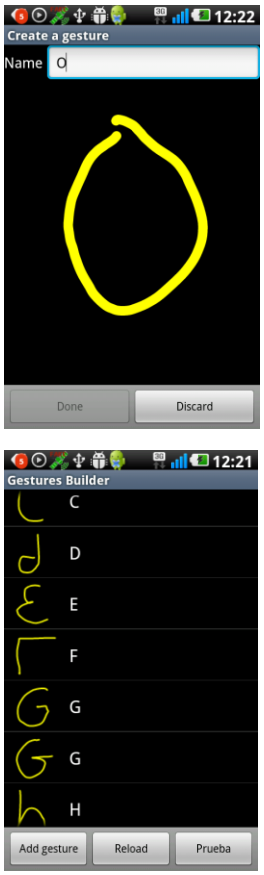


Imatge Gestures 1. Vorera de vibració en vermel

[12] Pro Android 3, (Pàgines 853 i 854)

(veure *Imatge Gestures 1. Vorera de vibració en vermell*) propera al final de pantalla, que cada cop que el dit s'apropa, fa que comenci a vibrar el mòbil. Un altre problema que ha sorgit posteriorment a això, era que si l'usuari començava a realitzar el *Gestures* des de fora de la pantalla cap a dins, i així no s'activa mai el reconeixement Gestures. Estranyament, des de sota si que s'activa correctament.

En els altres casos al no detectar inici de Gestures, no es podia agafar les coordenades d'incidència a la pantalla per lo qual no es podia avisar de que alguna cosa anava malament. Finalment, capturant les coordenades del *onTouchEvent*, es va poder iniciar una vibració que no para fins que no aixeca el dit, en senyal de que l'acció no es realitzarà correctament. També se li avisa per veu d'aquesta situació, indicant-li quina banda ha sigut la que ha produït l'error. D'aquesta manera, encara que inicialment l'usuari comenci des de fora, amb els avisos que va rebent, pot orientar-se i millorar la seva introducció de dades, fent-la molt més acurada.

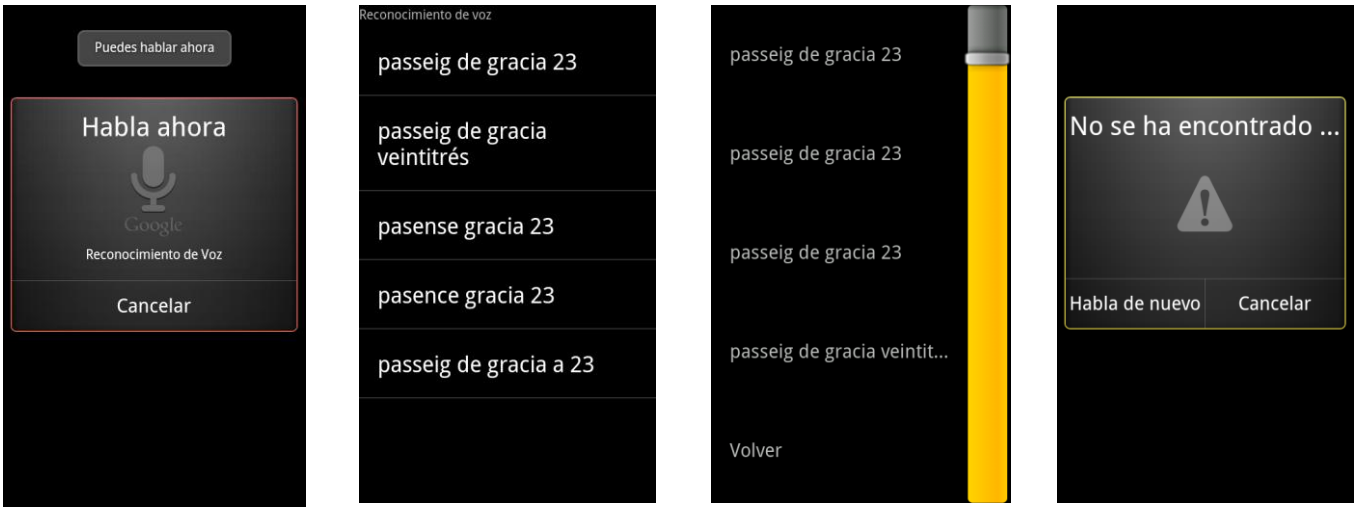


Pantalla de creació de Gestures

Amb totes aquestes millores, marges, alfabet simplificat, millor reconeixement, avisos de veu, etc... s'ha acabat aconseguint una bona interfície, 100% accessible i que dona uns bons resultats en un temps mínim.

Amb l'objectiu de poder testejar molt més ràpidament els símbols Gestures creats, es va adaptar un exemple de *l'API d'Android*:

7.3 VoiceRecognitionTime



Reconeixement de veu

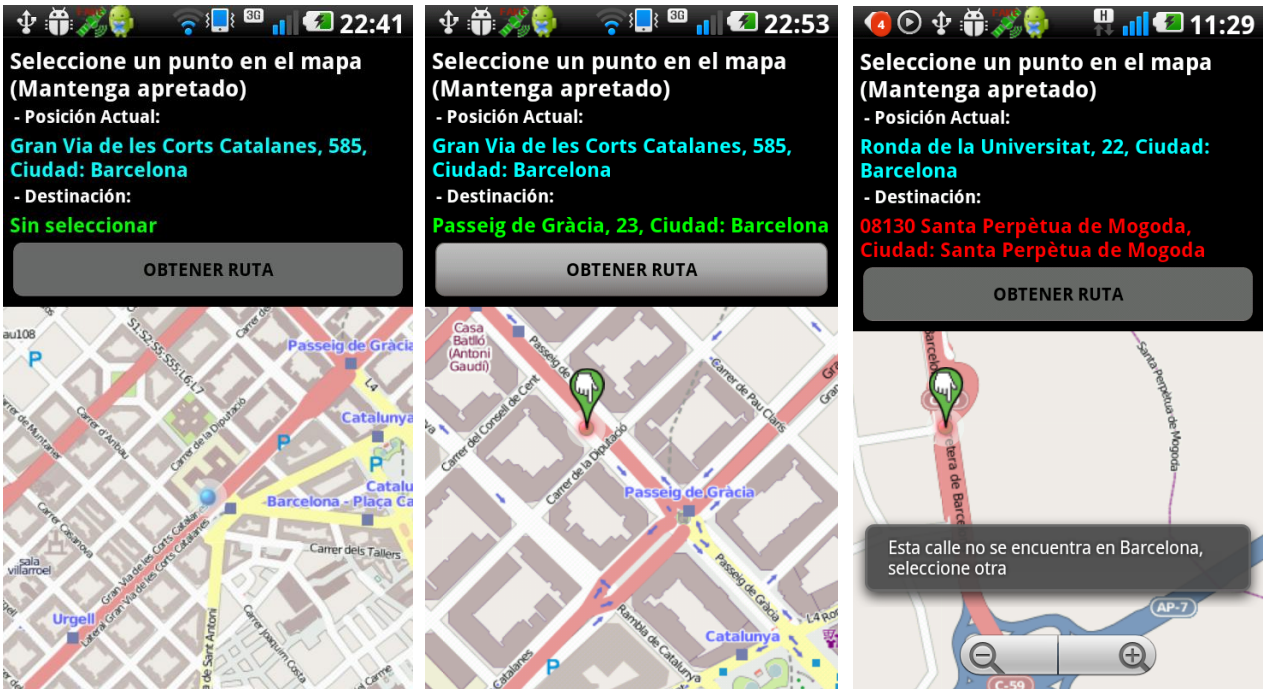
Com es va comentat en l'apartat d'interacció amb l'usuari, s'ha aprofitat la tecnologia que Google ha proporcionat als telèfons amb *Android*, de manera que es pot oferir al usuari la opció d'introduir carrer de destí i número amb veu, aconseguint uns bons resultats sempre i quan no hi hagi gaire soroll de fons.

S'ha aprofitat la *Activity* per què serveixi per tots dos tipus de perfils diferents, de manera que es carrega un *layout* o l'altre en rebre els resultats del reconeixement de veu.

A la darrera imatge es pot apreciar la pantalla d'error que surt quan els sons que es produeixen no han pogut ser reconeguts com a cap tipus de paraula i que no es pot capturar.

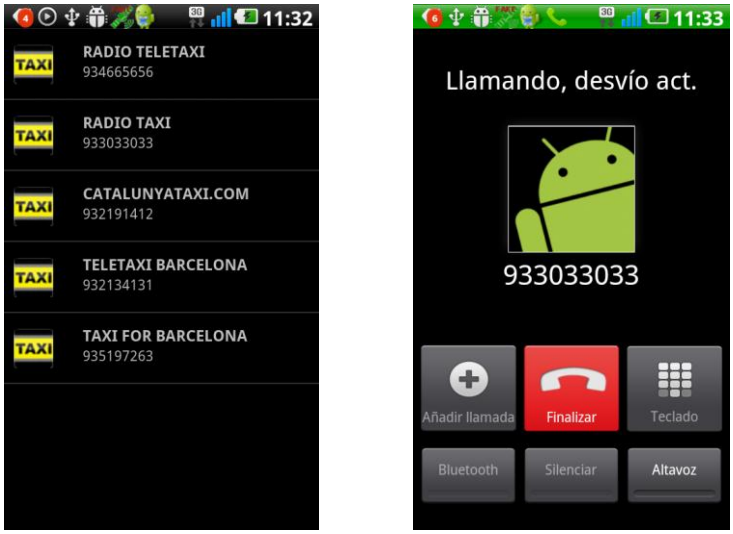
7.4 SelectMap

Aquesta ha sigut una de les darreres funcionalitats d'interacció de l'usuari implementades, però també és una de les més útils. Permet a l'usuari triar directament un punt qualsevol al mapa, amb la facilitat de mantenir polsat durant 3 segons. Al moment li apareix el nom del carrer i li permetrà continuar el procés només en cas que el carrer es trobi en un lloc permès per la aplicació.



Selecció des de Mapa

7.5 TaxisList



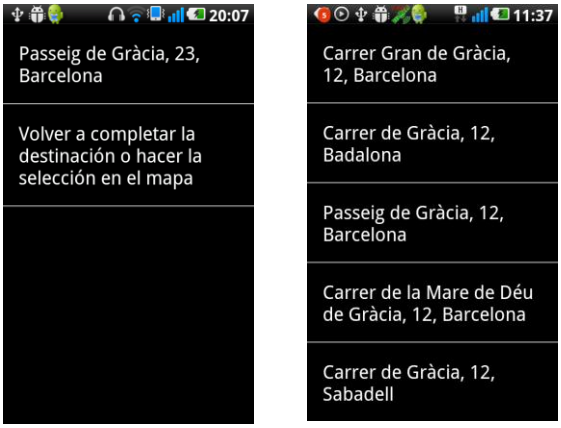
Imatge TaxisList1. Pantalla de companyies de taxi i trucada

S'ha afegit com alternativa al procés de guiatge, el poder trucar a alguna de les companyies de taxis de la ciutat (a cada ciutat de la qual tenim BD, hi ha 5 companyies de taxis per triar), havent obtingut la direcció on es troba l'usuari abans.

7.6 DestinationList

Aquesta *Activity* rep un *ArrayList* amb els resultats de cercar carrers que continguin el text que hagi introduït l'usuari a *DestinationSelect*, afegint-li la ciutat triada, per restringir els resultats.

El carrer que triï en aquesta pantalla, juntament amb l'origen detectat i les preferències del perfil seleccionat, definiran les restriccions de les rutes obtingudes a la pantalla de *RouteByDefault*. Es pot veure a les imatges que si per exemple introduïm "Carrer de Gràcia, 12" a Barcelona, ens retorna aquestes possibilitats, ja que hi han uns quants carrers en aquesta ciutat que contenen aquest nom.



Resultats obtinguts al cercar carrer

7.7 RouteByDefault

Un cop el mòdul de cerca ha calculat les millors rutes, envia els resultats a aquesta *Activity*, la qual ordena els resultats per temps total i mostra la informació de la ruta amb menys durada total.

Segons la ruta sigui a peu, amb autobús o amb transbord inclòs, la informació que es mostra varia. Des d'aquí, l'usuari té dos opcions (o només una en cas de que només es pugui arribar a la destinació a peu), començar la etapa de guiatge o si no està satisfet amb els resultats, pot modificar els criteris d'ordenació triant la segona opció.

Possibilitats de la ruta per defecte

Si el temps d'espera d'autobús es menor al temps d'arribada a la parada del primer bus (l'obtenim fent una consulta al mòdul de dades, que l'obté de la pàgina oficial de *iBus*[13]), llavors es descarta perquè se sap amb tota seguretat que no podrà arribar-hi i aleshores es comptabilitza un temps promig de 5 minuts que triguen els autobusos de Barcelona en arribar.

7.8 SelectRoute

Només es pot arribar a aquesta *Activity* si anem des de *RouteByDefault*, és a dir, que l'usuari arriba aquí volent canviar els criteris d'ordenació de rutes. Per fer això és molt senzill. Només ha de prémer qualsevol dels quatre botons que es troben amb les icones de transbord, a peu, rellotge (temps total) o accessibilitat i automàticament les rutes s'ordenaran segons lo triat. Aquesta taula està implementada de forma que en cas d'empat, segons el criteri que s'hagi triat, buscarà un desempat amb el criteri secundari.

[13] http://www.tmb.cat/es/ibus

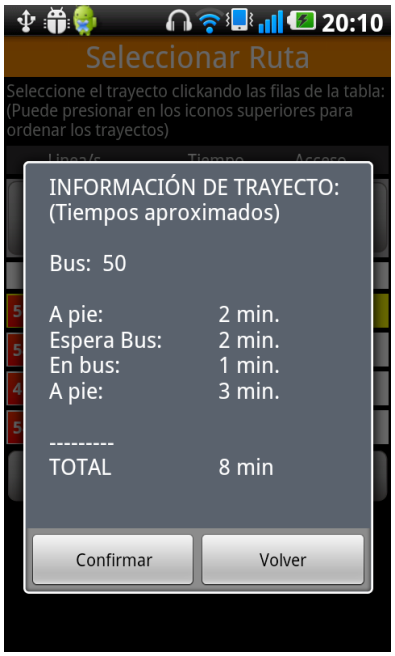
Casos d'empat	Desempat (criteri secundari)
Millor temps total	Millor temps a peu
Millor temps a peu	Millor temps total
Menor número de busos	Millor temps total
Millor accessibilitat	Millor temps a peu

L'usuari pot prémer qualsevol de les files per tal d'obtenir una informació molt més detallada dels temps del viatge. A les imatges següents es pot veure el desglossament de la informació. S'ha afegit un botó especial per poder actualitzar els temps d'espera dels autobusos.

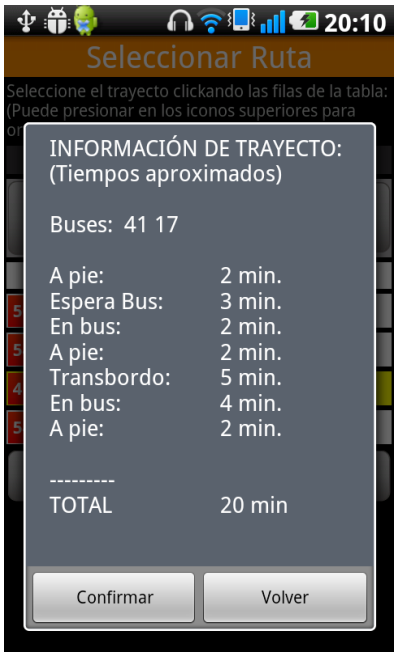
Rutes ordenades per temps total

Rutes ordenades per temps caminat

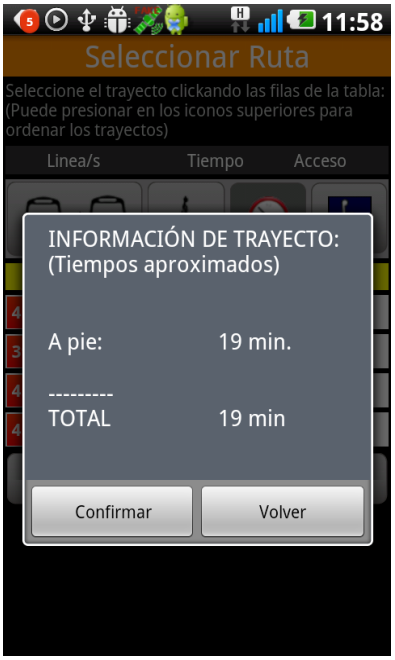
Recarregant els temps d'espera



Detalls de ruta amb autobús



Detalls de ruta amb transbordament



Detalls de ruta a peu

Hi han implementades les classes:

a) *AccessibilityComparator*
b) *LinesComparator*
c) *TimeTotalComparator*
d) *TimeWalkingComparator*

Són les que fan possible aquestes reordenacions segons els criteris que s'han comentat en l'apartat de SelectRoute.

7.9 Routing

Un cop es confirma la ruta, es contacta amb el nucli de l'aplicació, que ens dona pas a la pantalla de guiatge.

Trajecte a peu

És bastant interessant el fet de que es pot guardar la nostra posició en qualsevol moment, per poder tornar-hi en un altre moment.

Segons el trajecte que estiguem fent, hi poden haver 3 tipus de guiatge diferent, més el de la brúixola:

Guiatge a peu

En aquesta moment del guiatge, l'usuari pot estar sota aquestes condicions:

- Realitzant un trajecte sencer d'origen a destí a peu.
- En el cas que s'agafi un bus, pot trobar-se de camí a la parada o haver-se baixat i caminant cap al destí.
- Si es un trajecte amb transbordament, pot estar en qualsevol dels casos del guió anterior o pot estar caminant de la primera línia a la segona, fent transbordament.

Guiatge en el moment de l'espera de l'autobús

Aquest tipus de tram només es produeix sota dos condicions:

- En una ruta amb un sol bus, esperant a la parada
- Ruta amb transbord, en qualsevol de les dues esperes d'autobús.

S'activa automàticament quan l'usuari es troba dins d'un radi determinat de la parada on ha d'arribar el seu autobús. Obté el temps d'espera des de la pàgina oficial i l'actualitza periòdicament.

Trajecte d'espera de bus

Quan s'apropa la hora d'arribada del bus, la freqüència de les notificacions es van incrementant i en cas de tenir la App minimitzada, arribarà un avís per la barra de notificació. Com realment no hi ha manera de saber si l'usuari ha pujat al bus o no, s'ha decidit incorporar un botó de confirmació que disparará el següent tipus de guiatge.

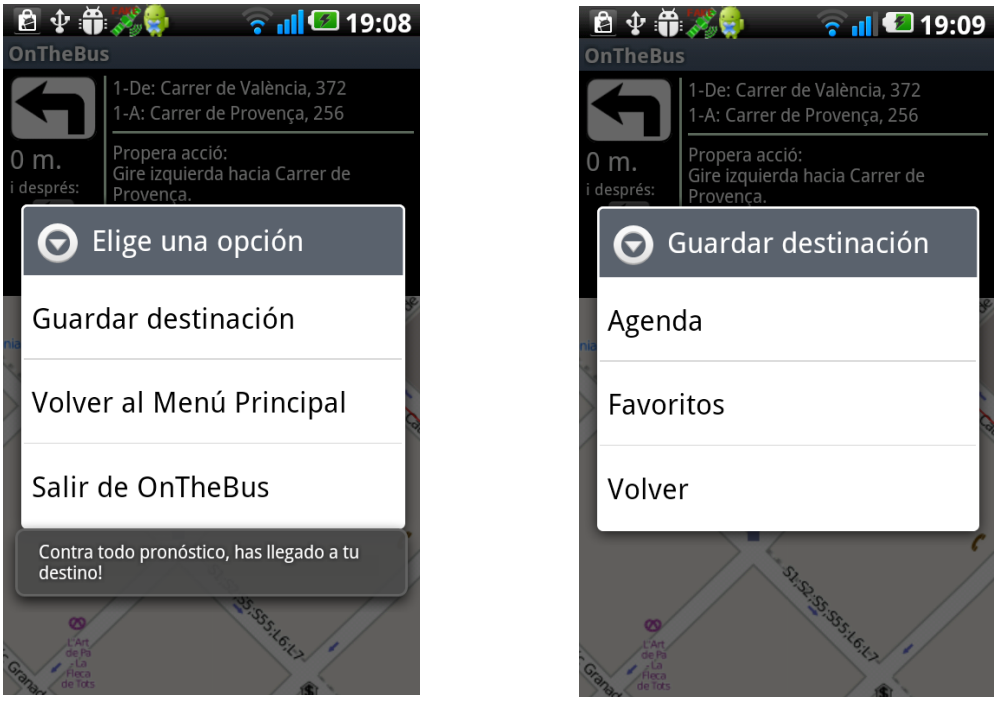
Guiatge a l'interior de l'autobús

En el moment en que fem servir aquest guiatge, l'usuari ja ha pujat al bus perquè ho ha notificat abans. Pot consultar tota la llista sencera de parades de la línia i veure remarcades les seves. A mesura que les va recorrent, van canviant de color per ser més intuïtiu.

Trajecte en autobús. A la dreta, botó de llista de parades premut

Com en el cas anterior, l'usuari ha de notificar a l'aplicació que ha baixat del bus perquè no podem tenir la certesa d'aquest fet, almenys de manera immediata.

Un cop l'usuari ha arribat al seu destí, automàticament apareix a la pantalla un diàleg que l'avisa d'aquest fet i li dona com opcions acabar l'aplicació, tornar al menú principal o guardar la seva posició actual, ja sigui associant-la a algun contacte de la seva agenda o als seus favorits.



Pantalla d'arribada a la destinació. Opcions disponibles.

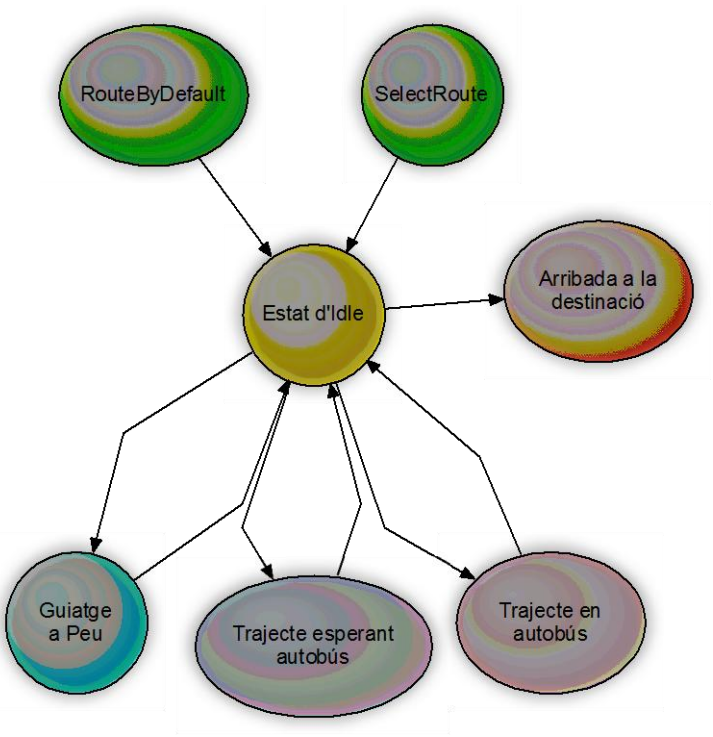


Diagrama OTB 1. Diagrama d'estats de Routing.java

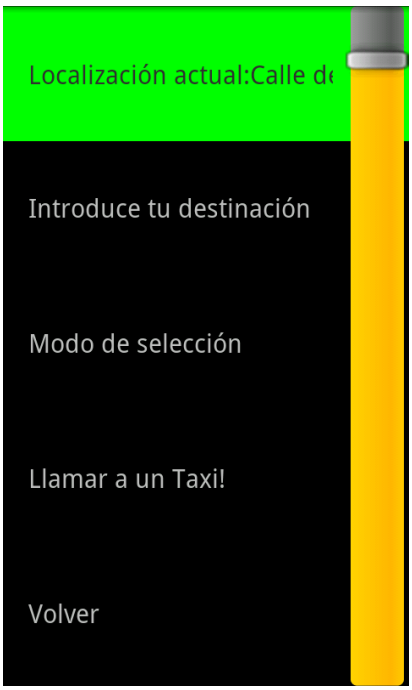
8 Activities II (Accessibilitat)

8.1 AbsVerticalSeekBar, VerticalProgressBar i VerticalSeekBar

Es va modificar l'exemple trobat a Internet i es van afegir mètodes a VerticalSeekBar per poder definir els dominis de cada fila d'opcions i per saber en cada moment en quina fila es troba el selector.

8.2 DestinationSelectBlind

Es van haver de realitzar remodelacions en l'estructura de menús, per aconseguir una millor fluïdesa en la navegació i que l'usuari sabés en cada moment en quina pantalla es trobava i quina era la seva funció. Finalment es va triar una estructura més complexa internament però que alliberava de prendre tantes decisions a l'usuari.

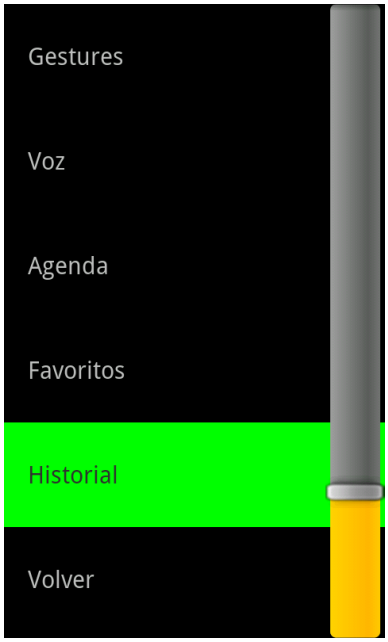


DestinationSelectBlind.java

8.3 ModifyConfBlind

En aquesta pantalla, l'usuari tria la forma en que vol introduir el carrer i el número de destí, com en la versió sense accessibilitat. En aquest cas, la particularitat és que l'autocompletar es transforma en la opció d'historial, aconseguint automatitzar el procés. La resta d'opcions són similars a les ja esmentades en *DestinationSelect.*

Si es tria un destí des de agenda, favorits o historial, l'aplicació arribarà directament a *DestinationListBlind*, perquè ja conté la informació de la ciutat de destí en canvi, si es tria *Gestures* o veu, és necessària una confirmació abans de començar a buscar.



Pantalla de selecció de mode

8.4 TaxisListBlind

Aquesta pantalla també és homòloga a la de selecció de companyies de taxis de la versió no accessible per aquesta raó no entrarem en més detall.

Pantalla de selecció de companyia de Taxis

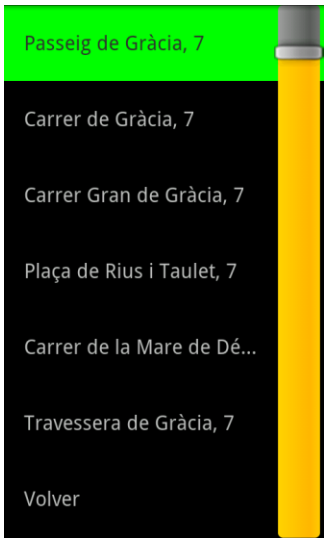
8.5 DestinationHistoryBlind (Disseny Final)

Com a l'historial de *DestinationSelect*, hem emmagatzemat les darreres destinacions cercades per part de l'usuari. En aquest cas hi ha un inconvenient que no apareixia en l'altre perfil, hi ha un número d'opcions limitat i per aquesta raó s'ha decidit deixar a triar les quatre destinacions més recents que es van fer servir, ordenades de més a menys noves.

Pantalla per triar carrer de l'historial

8.6 DestinationsListBlind

Aquesta *Activity* funciona de manera similar a l'altre perfil, internament fa les mateixes comunicacions amb el nucli de l'aplicació i obté com a resposta les opcions que més s'assemblen al carrer introduït.



Pantalla de resultats que contenen "gràcia"

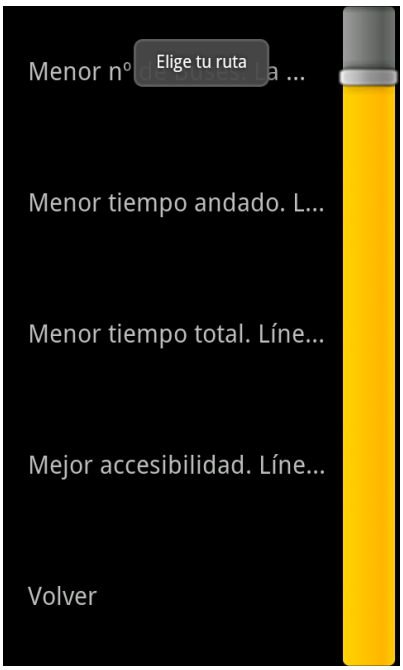
8.7 RouteByDefaultBlind

Després de realitzar implementacions alternatives, s'ha decidit que aquesta és una de les formes més adequades de presentar la informació detallada dels trajectes sense donar massa informació de cop. Per començar, es presenten com a molt, les 5 rutes més curtes fins al destí. Al seleccionar cada opció, el *TTS* informa de si la ruta és a peu o amb autobús (o transbordament). A continuació, es comunica el temps total de la ruta i el temps total a peu (només en cas de que siguin diferents). Si l'usuari vol rebre informació més detallada de la ruta, bastarà amb disparar l'ajuda i el *TTS* mostrarà el desglossament de temps.

Pantalla per triar la ruta

8.8 SelectRouteBlind

Aquesta *Activity* també ha canviat la seva funcionalitat original. Abans servia per seleccionar la ruta, però actualment només serveix per triar un criteri d'ordenació i retornar a la pantalla anterior, on s'actualitzarà la informació, ja reordenada.



Pantalla per triar el criteri d'ordenació

8.9 GuiConstants

Aquest objecte conté tot el conjunt d'elements constants que es requereixen des de la interfície. És molt preferible treballar d'aquesta manera per tal de centralitzar la informació. Un canvi en una d'aquestes constants, generarà un canvi en els dos perfils de la interfície.

8.10 GuiUtils

GuiUtils és una classe estàtica que proporciona uns mètodes que es re aprofiten en diverses *Activities*. Hi ha un mètode que s'encarrega de filtrar els noms de les parades, per mostrar una informació 'neta' a l'usuari (internament fem servir terminacions diferents per designar tipus de línies). Els altres mètodes són per encarregar-se de normalitzar els noms dels carrers, abans d'introduir-los a la BD de l'historial per aconseguir evitar insercions de carrers ja existents. Inicialment es va provar amb un *equalsIgnoreCase* però no va ser suficient, perquè això no evitava que s'inserissin erròniament casos com:

"carrer consell de cent, 265" i " carrer consell de cent, 265".

Amb els mètodes implementats, s'ha pogut arreglar aquesta fallada.

8.11 Parcelables

Aquests són els objectes que ha utilitzat principalment la interfície gràfica per comunicar-se amb la resta de mòduls de l'aplicació. Per tal de poder-los enviar i rebre, s'ha implementat la interfície *Parcelable*.

8.11.1 Objecte Preference

Conté totes les preferències de l'usuari. Inicialment s'anava enviant d'Activity en Actvity mitjançant els putExtra y getExtra però finalment s'ha implementat fent servir el patró de disseny Singleton.

8.11.2 Objecte Row

Classe que s'encarrega d'adaptar la informació rebuda en forma de *Tracks* per tal de poder modificar-la per poder mostrar-la per pantalla, vinculant-la amb els temps de iBus. Realment, és l'array de Rows el que es va ordenant cada cop que es selecciona un criteri, no el de *Tracks*, que és el que es rep inicialment. En el moment d'enviar l'origen i la destinació triats,

aprofitem la vinculació de l'id de *Rows* per tornar a seleccionar el *Track* original i començar el procés.

8.11.3 OBJECTES GUIDANCE

Guidance:

Classe base de la que hereten la resta de classes que es fan servir a *Routing*. Únicament conté un id, que es farà servir a les classes derivades segons sigui el seu context.

Derivades: Els objectes d'aquestes classes van arribant a Routing portant informació i produeixen actualitzacions en la pantalla.

Guidance_wlk:

Modifica la informació de passeig, mostrant la indicació actual, la propera indicació, els metres que falten per arribar al proper punt de control...

Guidance_wait:

Mostra el número de línia, juntament amb les característiques de l'autobús.

Guidance_bus:

En el primer paquet que ens arriba, vénen totes les parades, per poder consultar-les.

Guidance_compass:

Anirà actualitzant les dades de la brúixola.

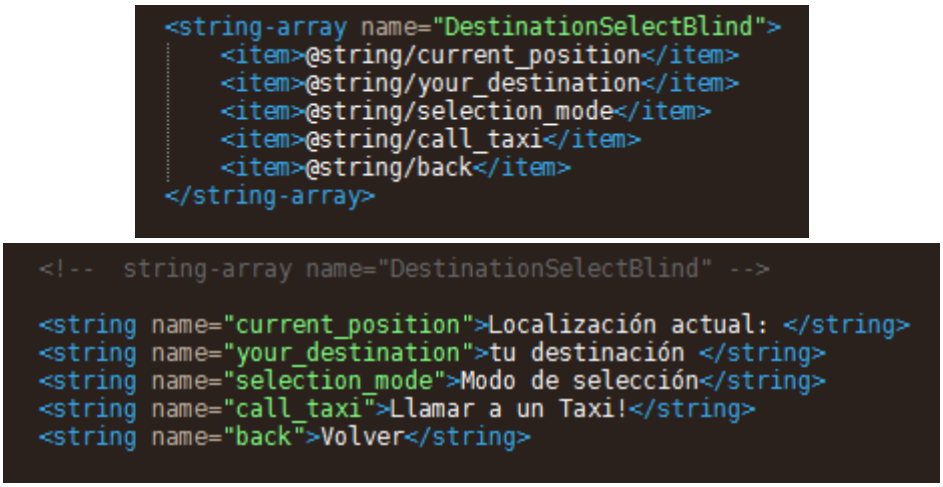
9 Multillenguatge

Afortunadament, Android utilitza una estructura interna que permet relacionar els fitxers de manera que facilita molt la traducció de la *App* a qualsevol altra llengua. La clau de tot això és el fitxer *strings.xml*. A la carpeta *res*/*values*/ es troba aquest fitxer, que conté tots els texts de l'aplicació. Si volem fer la traducció de la *App*, només hem de crear una carpeta "*values*-<qualificador>" paral·lelament a /*values*/ on <qualificador> es un codi regional que defineix l' idioma que conté. Llavors, segons l' idioma que tingui configurat el dispositiu, Android agafarà el *strings.xml* d'una ruta o de l'altra.

Quan associem un text a un component de la interfície, podem diferenciar en dos casos principals:

- Cas **estàtic**: aquest text no es modificarà durant tota l'execució de l'aplicació. Llavors la manera de fer-lo multillenguatge és ficant el seu contingut al fitxer *strings.xml* i fer servir l'identificador al que s'ha associat en aquest fitxer al *layout*.

- Cas **dinàmic**: si el text que volem traduir es pot modificar en qualsevol moment, podem associar-lo a un array *d'Strings* o a una variable:
  String[]menu = getResources().getStringArray(R.array.DestinationSelectBlind);
  I al fitxer *arrays*.xml, en comptes de posar el text directament, lo més adient es posar les referències al fitxer *strings*.xml, de manera que puguem concentrar tota la informació a un sol fitxer, que serà el que es traduirà.



Captura de pantalla on es veu la vinculació entre l'arxiu arrays.xml (a dalt) i el strings.xml (a baix)

9.1.1 AUTOMATITZACIÓ

Per tal de realitzar de manera més ràpida aquesta traducció, es van analitzar diverses possibilitats:

- Implementació d'un parser i un parser invers, per tal d'extreure per una banda els identificadors i per altre els texts, per realitzar còmodament la traducció.
- Cerca de parsers ja existents, per evitar possibles errors.
- *Plugin* per *Google Chrome* anomenat *Android Multilangual Translation Tool* que automatitza tota la feina, traduint el fitxer de *Strings* a més de 20 llengües diferents, mitjançant *Google Translator*, de forma que després s'ha de fer una revisió manual. Es pot fer servir sempre i quan tots els textos es trobin concentrats al fitxer *strings.xml*.

```
<string-array name="DestinationSelectBlind">
    <item>@string/current_position</item>
    <item>@string/your_destination</item>
    <item>@string/selection_mode</item>
    <item>@string/call_taxi</item>
    <item>@string/back</item>
</string-array>
```

Mètode per redireccionar els texts de l'arrays.xml al fitxer strings.xml

Depenent del grau de dinamització dels texts, poden aparèixer dificultats a l'hora de crear textos genèrics que es puguin traduir directament o amb el mínim nombre de canvis d'una llengua a una altra, degut a les diferències sintàctiques que es poden trobar.

10 Càlcul de número de clics màxims, mínims i mitjana per iniciar un recorregut en els 2 modes

Es pot veure com són gairebé equivalents en lo que respecta a velocitat de navegació:

	Mode sense accessibilitat	Mode amb accesibilitat
Millor dels casos	5 (amb Rec. Veu)	6 (amb Historial)
Pitjor dels casos	7,5 -> 8	8
Mitjana	10 (amb Favorits i canvi de ciutat)	10 (amb canvi de ciutat i Gestures)

11 Proves i resultats

Durant tot el projecte s'han anat fent múltiples proves per tal de poder testejar el correcte funcionament de l'aplicació. Per la realització de les proves s'ha utilitzat l'emulador que Android incorpora en el seu SDK (permet emular dispositius mòbils amb el SO d'Android incorporat sense necessitat d'utilitzar un dispositiu físic) i els diferents telèfons mòbils disponibles pels membres del projecte.

A continuació es detallen algunes de les proves més significatives realitzades i els resultats obtinguts a la ciutat de Barceona.

11.1 Proves i resultats de camp

*Proves realitzades a Barcelona. Per la resta de mòduls (ciutats) consultar l'*Annex B*. Les proves fan referència al perfil sense discapacitat. El sistema de guiatge és el mateix per tots els perfils, només varia la interfície, per tant, s'obtindran els mateixos resultats.*

11.1.1 PROVES I RESULTATS DE LOCALITZACIÓ

En aquest apartat es mostraran els diferents resultats obtinguts al obtenir la localització actual del usuari:

- Localització Precisa:

Figura 1 - Localització amb perfecte precisió.

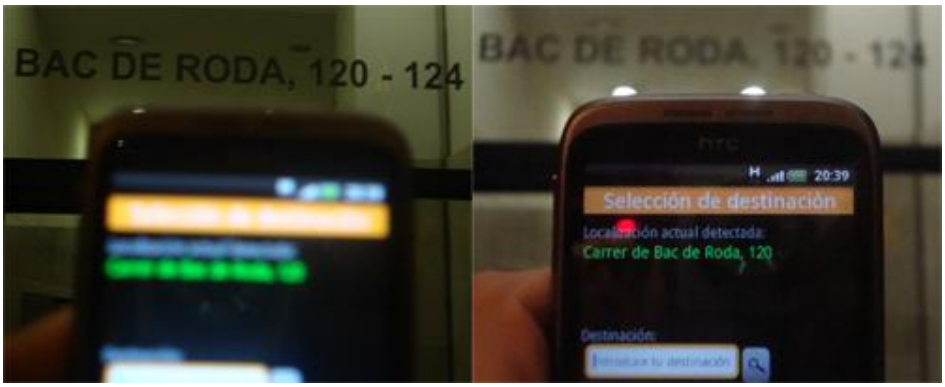


Figura 2 - Localització amb perfecte precisió

- Errors de localització:

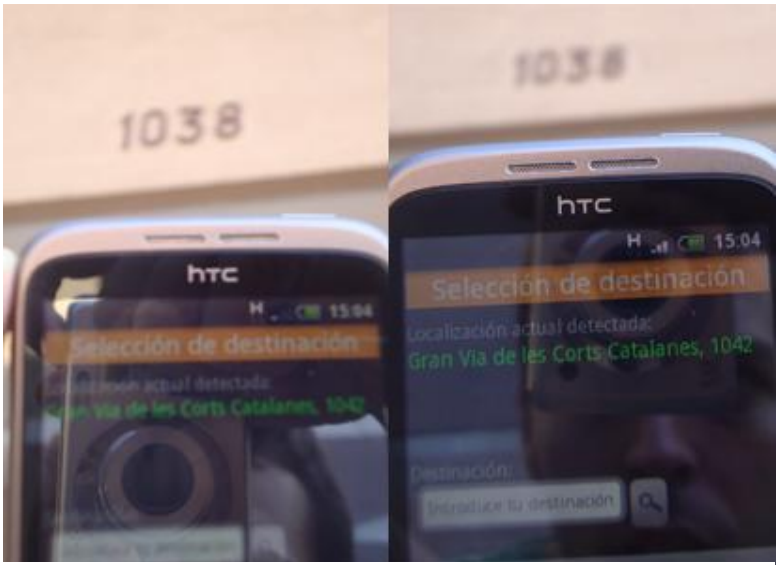


Figura 3 - Localització errònia

Es aquest cas es pot comprovar que hi ha un imprecisió. L'usuari està al carrer *Gran Via*, número *1038*, però la localització marca el número *1042* del mateix carrer. Això és degut a que el *Geocoder* no mostra tots els números d'un carrer, sinó que només mostra uns quants en un segment determinat i per tant treu precisió a l'hora d'obtenir i mostrar l'adreça actual del usuari.

Un error d'imprecisió semblant i derivat del *Geocoder* es troba en la *Figura 4.* En aquest cas es pot veure que l'usuari està una mica lluny de la seva posició actual i a més que el *Geocoder* retorna un conjunt de números en els quals pot estar l'usuari, per tant, la precisió en cas d'estar en el conjunt de números, tampoc serà total.

Figura 4 - Localització errònia, agrupació de números.

11.1.2 PROVES I RESULTATS DE GUIATGE

11.1.2.1 Guiatge WALK

- Guiatges

Figura 5 - Exemple simple de gir.

En la *Figura 5* es pot veure com avança l'usuari pel camí i la indicació marcada. Just en el punt en que l'usuari ha de girar per un carrer, li es mostrada la indicació de gir sense cap tipus d'error. En les següents il·lustracions (*Figura 6*) es mostrarà uns exemples de guiatge amb un bona precisió de girs.

Figura 6 - Guiatge amb un bona precisió de girs

- Recalcular la ruta

Figura 7 - Recalcular Ruta

Es pot apreciar en la primera imatge de la *Figura 7* que l'usuari ha iniciat el guiatge de caminar però que no li arriben indicacions, ja que mentres es calculava la ruta ha mogut la seva posició (es pot apreciar la diferència entre l'inici del camí vermell i el punt de localització). En les dues següents imatges es pot veure com l'aplicació detecta l'error i automàticament recalcula la ruta de caminar fins al proper punt de guiatge o destí de l'usuari.

- Errors
  - GPS

Figura 8 - Guiatge Walk. Error localització GPS.

En la *Figura 8* es pot apreciar que el punt de localització està lluny del camí que hauria de seguir l'usuari (marcat en vermell) i que a més, el punt marca que l'usuari està dintre d'un edifici. Aquest error de localització passa de tant en tant i és molt difícil de controlar, ja que deriva de l'error de GPS que tenen els dispositius mòbils.

- Errors Mapa

Hi ha vegades que el mapa que utilitzem de *OpenStreetMap* no s'acaba d'aproximar amb l'estat actual dels carrers, i això ens indueix principalment a dos tipus d'error:

- Cantonades

Figura 9 - Guiatge Walk. Cantonades

Sovint el mapa mostra cantonades amb girs de 90º (amb forma de "quadrat", com es pot veure a la *Figura 9*) que no s'aproximen amb la realitat, ja que moltes han sigut adaptades en forma de diagonal per aprofitar millor l'espai públic. Aquest fet indueix a imprecisions en el guiatge ja que l'aplicació s'ajusta al mapa proporcionat per poder implementar el servei. Per tal de solucionar-ho s'ha augmentat el radi de la *proximity alert* que avisa que l'usuari ha de girar, per a que aquest ho pugui fer correctament, com es pot veure en l'exemple.

- Voreres

Figura 10 - Guiatge Walk. Voreres

És el cas més complicat i de moment impossible de controlar degut a la seva complexitat i dificultat. El servei proporciona una ruta i indicacions que són mostrades a l'usuari en el format indicat. El problema deriva en que moltes vegades quan hi ha un canvi de vorera no es proporciona aquesta informació, i per tant, no es pot indicar al usuari que canviï de vorera quan es fa el gir. En el mapa de la *Figura 10* es pot veure com es realitza un canvi de vorera però no s'informa en cap moment en les indicacions.

11.1.2.2 Guiatge BUS

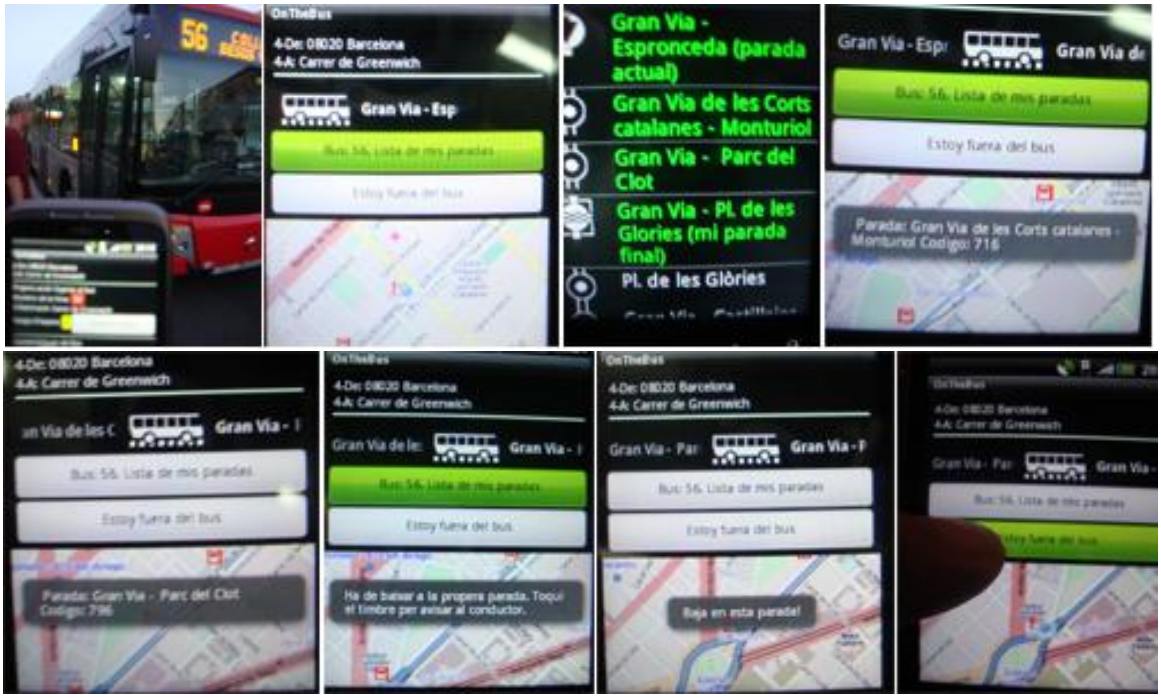


Figura 11 - Guiatge Bus. Etapes de Guiatge

Com es pot veure, un cop l'usuari puja a l'autobús pot consultar les parades que recorrerà, i cada cop que passi amb aquest per una parada rebrà un avís on l'indicarà la parada actual del recorregut. Durant el trajecte entre la penúltima i última parada, es mostra un missatge cada cert temps alertant a l'usuari de que ha de baixar en la següent parada i finalment al arribar a l'última parada, s'informa a l'usuari de que ha de baixar. Un cop ho hagi fet, l'usuari confirma a l'aplicació que ha deixa't l'autobús per continuar amb el següent guiatge.

- Problema: Passar-se de Parada

En aquest cas, l'aplicació avisa a l'usuari de que no ha baixat en la parada correcta i que cal recalcular la ruta. Un cop l'usuari baixa del autobús, es recalcula automàticament la ruta fins el destí (línia blava de l'última imatge) i comença de nou el guiatge.

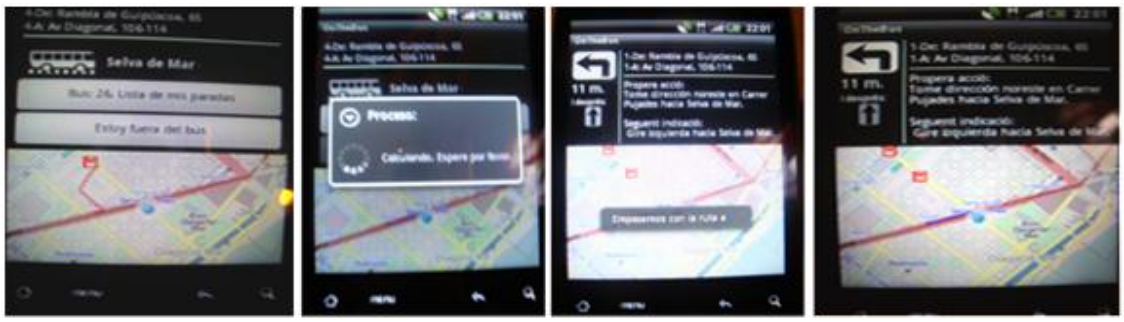


Figura 12 - Guiatge Bus. Exemple de passar-se de parada

11.1.3 PROVES TEMPS ESPERA

S'ha generat un conjunt de test per veure el nivell de precisió real del temps d'espera a la parada proporcionat per *TMB*.

- Test 1: Precisió Correcte.

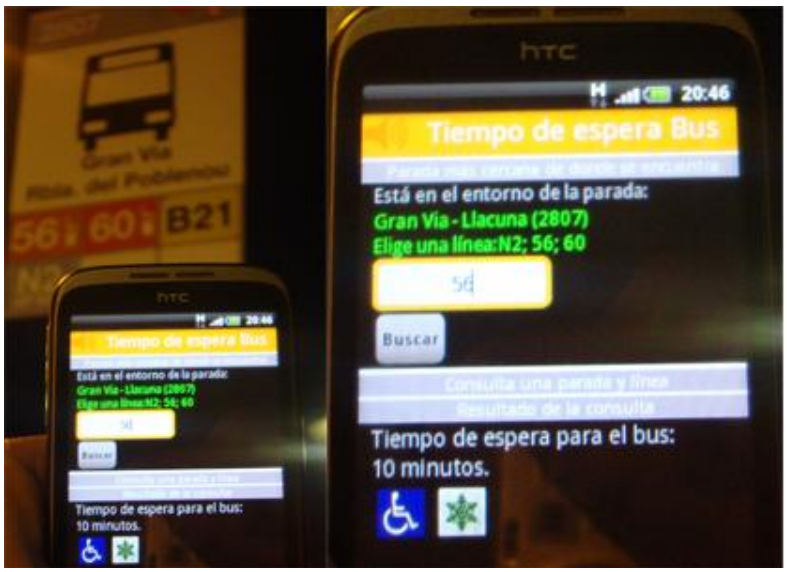


Figura 13 - Precisió correcte.

Com es pot veure en la *Figura 13*, l'aplicació detecta automàticament la parada en la que l'usuari està esperant el bus (codi *2807*). En aquest cas, al haver-hi diferents línies l'usuari ha d'introduir la línia que vol utilitzar.

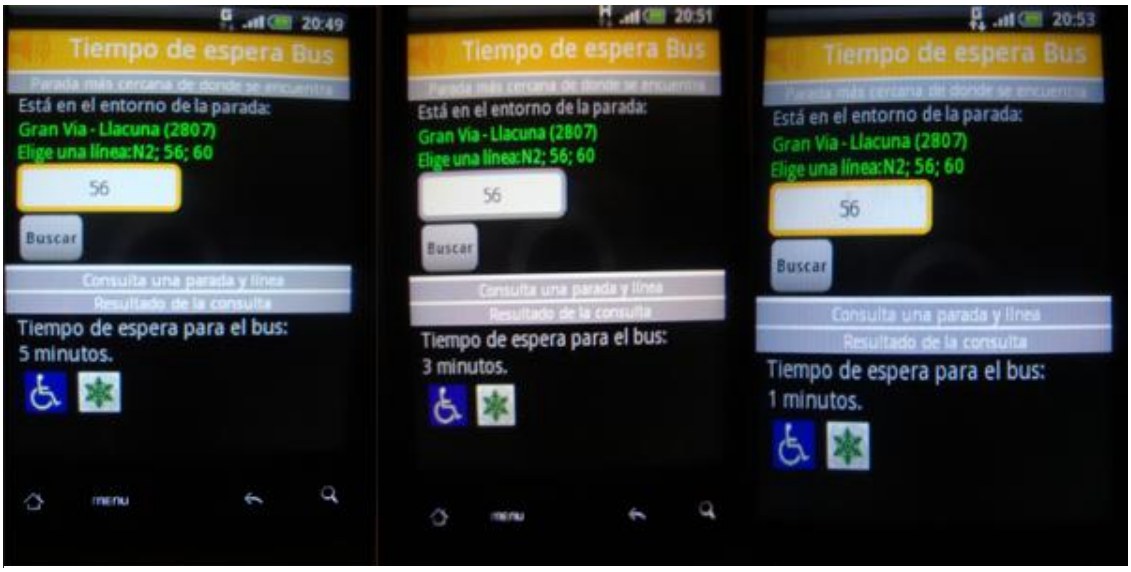


Figura 14 - Successió.

Si s'observa el temps que marca l'aplicació i l'hora que marca el telèfon mòbil, es pot veure que l'autobús sembla que arriba sense cap retard a la parada.

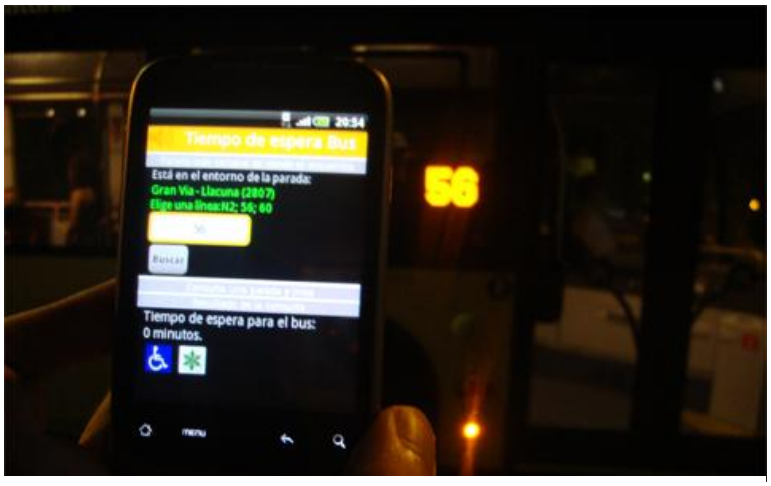


Figura 15 - Precisió correcte.

Finalment, es pot apreciar que l'autobús,ha arribat a l'hora prevista.

- Test 2: Error per falta d'informació.

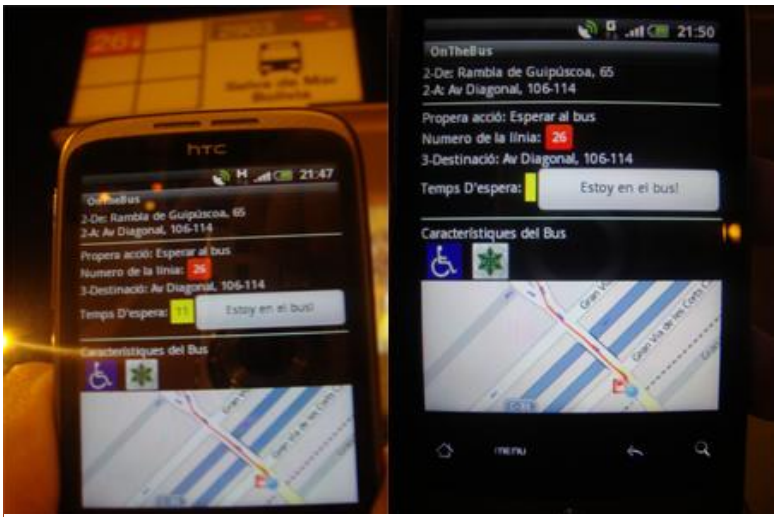


Figura 16 - Successió Temps Espera errònia. Falta d'informació.

Hi han casos en que la informació del temps d'espera no és disponible ja que falla la consulta a Internet per la connexió o perquè el proveïdor del servei no té les dades operatives, com es pot veure en la *Figura 16* anterior on passem de mostrar els minuts restants a no poder mostrar res.

- Test 3: Retard en l'arribada del autobús.

Figura 17 - Successió Temps Espera. Retard arribada.

A la *Figura 17* es pot observar que l'autobús ha petit un retard en el seu trajecte i trigarà més dels tres minuts que marcava la consulta inicial. Durant tres minuts de rellotge (*20:13 – 20:15*), el temps d'espera marcat per l'aplicació és el mateix, dos minuts. També s'hi produeix una imprecisió quant està a punt d'arribar a la parada, ja que es passa d'un minut restant a vint-i-tres, és a dir, hauria d'haver arribat i ja s'està esperant al següent.

En aquesta sèrie de testos es pot veure que de vegades el temps que falta pel proper autobús potser no concorda del tot amb la seva arribada, tot i tenir una precisió força alta en global. En el test 3 per exemple, es passa d'un minut restant a vint-i-tres, tot i que el bus no ha passat per la parada i pot estar parat en un semàfor o a uns metres. Els minuts que falten s'obtenen a través d'una consulta al servei web de la companyia en qüestió, per tant, depenem exclusivament d'aquest servei i no es pot generar un control en cas de diferències entre la informació que mostrem i l'arribada del bus.

12 Conclusions i possibles ampliacions

Reflexions al final d'aquest trajecte... millores per la propera ocasió.

12.1 Conclusions personals

Sincerament, puc afirmar que la realització d'aquest projecte ha sigut per mi molt profitosa, tant per l'àmbit professional com per el personal. A l'àmbit professional, perquè m'ha permès ampliar el meu currículum amb aquesta experiència acumulada durant aquests mesos, treballant en un projecte de mida mitjana, amb un equip de gent competent i amb una tecnologia que no coneixia prèviament però que ha marcat un abans i un després a la meva vida professional, afegint-me noves motivacions laborals. Per la part personal, perquè m'ha donat la oportunitat de continuar creixent com a persona, enfrontant-me a noves dificultats i trobant les seves solucions, com quan va ser necessari dissenyar i implementar la interfície amb accessibilitat visual.

Crec que he pogut assolir els objectius que se m'havien assignat en el projecte, aportant alternatives majoritàriament vàlides a les dificultats que han anat apareixent en el meu mòdul d'interfície gràfica al llarg del projecte, i col·laborant perquè els altres components de l'aplicació poguessin assolir els seus.

Per altra banda, el fet de descobrir un camp nou que tenia totalment desconegut, m'ha generat noves inquietuds de coneixement i alhora m'ha fet veure lo allunyat que estic d'arribar al nivell on vull arribar.

Aquest projecte també m'ha servit per adonar-me de la poca cobertura que actualment el mercat dona als grups minoritaris, com poden ser la gent que té cap necessitat especial d'accessibilitat. Amb la irrupció al mercat de les pantalles tàctils, el nombre de dispositius amb botons físics, que eren més fàcilment accessibles, ha caigut en picat. Encara continuen funcionant els revisors de pantalla però han perdut la seva eficàcia, degut a la dificultat d'establir un ordre en la navegació per el menú. A més, els dispositius amb *trackball* o *trackpad*, que funcionaven millor amb la combinació d'aquest software, estan desapareixent, com s'ha pogut veure a l'apartat de l'anàlisi de botons[14].

[14] Veure Annex A per complementar la informació.

Fa mesos que han arribat notícies[15] d'experiments que s'estan fent, on s'ha aconseguit fabricar pantalles de rugositat variable, això podria ser una possible solució per un futur mitjanament proper, però aquesta tecnologia encara no està estesa.

Apart d'aquesta aplicació, tampoc hem pogut trobar gaires aplicacions que tinguin en compte el disseny universal o que intentin millorar la integració dels diferents perfils que s'ha tingut en compte en aquest projecte, encara que paradoxalment, crec que ens trobem en el millor moment per fer-ho, aprofitant la quantitat de sensors que avui en dia porta un telèfon qualsevol. Espero per aquesta raó, que surtin endavant molts projectes oberts, que no estan tan lligats a uns resultats econòmics i que tenen un caire més social.

12.2 Conclusions generals

Els objectius principals del projecte s'han assolit. S'ha implementat una solució seguint el disseny universal i que funciona, realitzant correctament el guiatge des del punt inici on es troba l'usuari fins un punt destí fent ús de la xarxa de transport d'autobusos per a diferents ciutats d'arreu del món.

També s'ha aconseguit gestionar diferents grups homogenis en el desenvolupament d'un projecte de dimensions importants creant sinergies entre aquest grups per a assolir l'objectiu principal.

Alguns dels inconvenients que han aparegut han estat relacionats amb la precisió del dispositius GPS que incorporen els mòbils que s'han usat durant les proves de l'aplicació, donant diferents resultats cadascun d'ells. Això ha fet que s'allargués la duració del projecte per sobre del que s'havia previst inicialment. Aquesta desviació ha suposat aproximadament cinc setmanes de retràs, que han fet que l'entrega es realitzes en setembre.

12.3 Dificultats trobades pel camí del mòdul de la interfície gràfica

- Encara no és possible capturar l'error quan el reconeixement de veu falla, perquè actualment quan succeeix això, surt un *alert* amb dos botons que dóna la opció de tornar-ho a provar i cancel·lar, però malauradament aquest *alert* no és accessible[16], per aquesta raó es realitza una comprovació de que hi hagi connexió a Internet abans d'iniciar *l'Activity* de reconeixement de veu.

---

15 http://www.microsiervos.com/archivo/tecnologia/otra-pantalla-tacil-intenta-imitar-sensacion-teclado-fisico.html

16 http://code.google.com/p/android/issues/detail?id=4541

- No es pot variar la mida d'un *ListView* abans de mostrar el seu contingut, perquè dóna un *null pointer exeception.*
- Error important per part de *Google* el fet de que no es pugui detectar com a *Gestures* una línia vertical i una horitzontal a la vegada en un mateix *GestureOverlay*, degut a la orientació del *layout*.
- La falta d'implementació del mètode *getEdgeFlags*().
- Problemes inicials amb els marges dels botons, que produïen visualment voreres irregulars i ens van obligar a afegir un petit marge al voltant dels botons o en tot cas, a fer servir un *theme* com en el nostre cas.

Botons irregulars

12.4 Possibles ampliacions del mòdul de la interfície gràfica

En lo respectiu a futures millores de la interfície gràfica, es podria ampliar el grau de personalització de l'aplicació. Seria possible canviar els colors de més *Views* de forma dinàmica, com per exemple els *ListView*. A més, l'usuari podria variar la mida de la tipografia de forma dinàmica.

Un altre element que seria interessant personalitzar, seria la barra de progrés "*seekbar*" mitjançant la selecció de *skins* de forma estàtica o bé la selecció del seu color de forma dinàmica.

Altre punt a millorar, seria modificar el *layout* de guiatge fent-lo més atractiu, incorporant un sistema de finestres lliscants per a minimitzar i maximitzar la informació de guiatge.

Una alternativa a la millora anterior seria dissenyar una interfície de guiatge utilitzant conceptes de realitat augmentada.

Per millorar el tema de *Gestures*, es pot incorporar en properes versions, una funcionalitat per a inserir *Gestures* pròpies i associar-los als contactes de l'agenda i a favorits. Una altra millora per als usuaris que no necessiten un perfil d'accessibilitat visual, seria

millorar el sistema de detecció de símbols. Per tal de realitzar aquest canvi, a l'hora d'obtenir les diferents puntuacions en el *matching*, es descartarien tots els símbols menys els tres amb millor puntuació i es mostrarien aquestes alternatives a l'usuari com a suggeriment.

En la secció dels Taxis, seria interessant el tema d'enviar la posició actual a la companyia seleccionada en el moment de trucar.

Per aconseguir una experiència d'usuari encara millor, seria interessant explotar més el mòdul de reconeixement de veu i millorar la interfície d'accessibilitat visual, intentant homogeneïtzar les dues interfícies.

Com a darrera millora, no estaria malament aprofitar les eines de *Google Analytics* per tal de trobar els possibles colls d'ampolla del conjunt de menús.

12.5 Possibles ampliacions de l'aplicació

A continuació es presenten algunes de les possibles ampliacions a realitzar en l'aplicació. Es tracta de noves funcionalitats afegides, que resten pendents d'anàlisi i disseny.

Cerca per punts d'interès (POI): Es tractaria d'afegir la geolocalització de diferents punts d'interès per tipologia (comissaries, farmàcies, restaurants, etc.). Encara que queda pendent d'anàlisi aquesta modificació implicaria la modificació de la majoria del mòduls de l'aplicació.

Rutes turístiques: Es tractaria de tenir rutes complertes d'inici a fi amb un o més punts turístics. Es podria enllaçar amb l'ampliació anteriorment esmentada per tal d'emmagatzemar els punts turístics com a punts d'interès.

Recorregut Invers: Es tractaria de poder indicar-li a l'aplicació que emmagatzemi el punt inici i final de ruta per a en un breu espai de temps demanar que es guiï de tornada al punt inicial. És una ampliació interessant per a quan et desplaces puntualment a un destí.

Cercar un pla de negoci: Consistiria en l'addició d'algun tipus de publicitat en l'aplicació.

13 Bibliografia

1. *10 physical gestures that have been patented*

   http://io9.com/5808604/10-physical-gestures-that-have-been-patented
2. *Graffiti (Palm OS)*

   http://es.wikipedia.org/wiki/Graffiti_(Palm_OS)
3. Support: Using the Gesture Library

   http://gestureworks.com/support/supported-gestures/
4. Gestures. Problem getting inconsistent results

   http://stackoverflow.com/questions/6380908/android-gestures-single-stroke-multi-stroke-produce-inconsistent-results
5. Google Books – Android Pro 3

   http://books.google.es/books?id=_lV-jkAHLtsC&pg=PA853&lpg=PA853&dq=android+event.getEdgeFlags()+returns+0&source=bl&ots=3ZaqEdwYbH&sig=Igrm0EtacbRT2E8T6t2jDV-uUUc&hl=es&ei=OkleTsWQAcWh8QOR4ZisAw&sa=X&oi=book_result&ct=result#v=onepage&q&f=false
6. MarkMail – Reply from Dianne Hackborn (*Android framework engineer)*

   http://markmail.org/message/7l34vhzfbikteveh#query:+page:1+mid:iqlxi4jn5d4lvkyt+state:results
7. Does MotionEvent.getEdgeFlags() ever return anything other than 0 ?

   http://groups.google.com/group/android-developers/browse_thread/thread/a55db4c49b12248d?pli=1
8. Vibration Examples for Android Phone Development

   http://android.konreu.com/category/developer-how-to/
9. Key press volume is associated with media volume!

   http://www.google.com/support/forum/p/Google+Mobile/thread?tid=1d6a1d1940c7636c&hl=en
10. AudioManager

    http://www.smartandroidapps.com/faqs-and-tips-for-audiomanager/
11. Basic XML attributes for controlling IMEs

    http://developer.android.com/resources/articles/on-screen-inputs.html
12. ListView

    http://developer.android.com/reference/android/widget/ListView.html
13. Vertical SeekBar

    http://www.tutomobile.fr/realiser-une-progressbar-seekbar-personnalisee-sur-android-tutoriel-android-n%C2%B024/10/02/2011/
14. Multilanguage

    http://developer.android.com/guide/topics/resources/localization.html

15. Accessing Resources

    http://developer.android.com/guide/topics/resources/accessing-resources.html

16. Localization

    http://developer.android.com/guide/topics/resources/localization.html

Signatura: Eric Xavier Lara Amat y León

Bellaterra, Setembre 2011

Resum

El projecte OnTheBus pretén ser un sistema de guiatge gratuït, destinat per a la plataforma mòbil Android, útil com a eina d'ajuda pel desplaçament d'un punt d'origen a un punt de destí dins de la ciutat, utilitzant la xarxa de transport metropolità.

Aquesta memòria s'enllaça amb 6 memòries més, dels meus companys amb els qui he realitzat el projecte OnTheBus.

Per poder elaborar amb èxit l'aplicació OntheBus, cal donar importància al disseny i a la implementació de la interfície d'usuari. L'objectiu d'aquesta memòria és explicar com s'ha realitzat i dissenyat l'aplicació perquè sigui accessible per a tothom. Ja que pot tenir un rang molt ampli d'edats d'usuaris, la implementació de la interfície està basada en les directrius del Disseny Universal.

Resumen

El proyecto OnTheBus pretende ser un sistema de guiado gratuito, destinado a la plataforma móvil Android, útil como herramienta para desplazarse desde un punto de origen a un punto de destino dentro de la ciudad, utilizando la red de transporte metropolitano.

Esta memoria se enlaza con 6 memorias más, de mis compañeros con los cuales he realizado el proyecto OnTheBus.

Para poder elaborar con éxito la aplicación OnTheBus, hay que dar importancia al diseño y a la implementación de la interfaz de usuario. El objetivo de esta memoria es explicar cómo se ha realizado y diseñado la aplicación para que sea accesible para todo el mundo. Como puede tener un rango muy amplio de edades de usuarios, la implementación de la interfaz se ha basado en las directrices del Diseño Universal.

Abstract

The OnTheBus Project pretends to be a guiding free system, designed for the Android platform, with the target to go from one source point to a destination point in a chosen city, using the metropolitan transport network.

This study links with another six works from my partners who we have been developing this project.

To succeed with OnTheBus app, it is important to give relevance to the design of the application and to the user interface implementation.

The work's objective is to explain how we did the app to be accessible for everybody. Because the OnTheBus program can be used for people of different ages, this causes the interface implementation it is based into the Universal Design guidelines.