shoPOIng

Gestor de POIs per dispositius Android

Memòria del projecte

D'Enginyeria Tècnica en

Informàtica de Gestió

realitzat per

Carlos García Ferrer

i dirigit per

Marc Tallò Sendra

Escola Universitària d'Informàtica

Sabadell, Septembre de 2012

[El/La] sotasignat, ***Marc Talló Sendra***,

professor[/a] de l'Escola d'Enginyeria de la UAB,

CERTIFICA:

Que el treball al que correspon la present memòria
ha estat realitzat sota la seva direcció per

Carlos García Ferrer

I per a que consti firma la present.
Sabadell, ***Setembre*** de ***2012***

Signat: ***Marc Talló Sendra***,
director/a

FULL DE RESUM – PROJECTE FI DE CARRERA DE L'ESCOLA D'ENGINYERIA

Títol del projecte: Gestor de POIs per dispositius Android	
Autor[a]: Carlos García Ferrer	**Data:** Setembre de 2012
Tutor[a]/s[es]: Marc Tallò Sendra	
Titulació: Enginyeria Tècnica en Informàtica de Gestió	
Paraules clau • **Català:** Android, PDI, targeter, botigues, localització, navegador • **Castellà:** Android, PDI, tarjetero, tiendas, localización, navegador • **Anglès:** : Android, POI, address holder, shopping, geolocation, navigator	
Resum del projecte • **Català:** Aplicació per emmagatzemar les direccions dels llocs d'interès de l'usuari amb l'objectiu de substituir els targeters tradicionals. La seva funcionalitat inclou poder gestionar-los, compartir-los a través del correu electrònic i classificar-los, amb una sèrie de categories i paraules clau per després poder geoposicionar-los i traçar rutes entre els diferents punts des de la ubicació on es troba l'usuari. • **Castellà:** Aplicación para almacenar las direcciones de los lugares de interés del usuario con el objetio de substituir los tarjeteros tradicionales. Su funcionalidad incluye poder gestionarlos, compartirlos con otros usuario por medio del correo electrónico y clasificarlos, con una serie de categorías y palabras clave para posteriormente poder geoposicionarlos y trazar rutas entre los diferentes puntos desde la ubicación donde se encuentre el usuario. • **Anglès:** Aplication to store places wich may be relevant for the user in order to replace the traditional address holder.It's functionallity includes the managing of the POIs, sharing them with other users by eMail, and catalog them with a category and keywords system, so the user can trace routes between these points from his/her actual location at any time.	

ÌNDEX

1. INTRODUCCIÓ

1.1. PRESENTACIÓ

DE QUE TRACTA EL PROJECTE?

Les motivacions per dur a terme aquest projecte van ser principalment dues.

Per una banda, aprendre un llenguatge nou basat en dispositius mòbils, que en aquests moments es un dels camps dins de l'àmbit del desenvolupament d'aplicacions que més importància està adquirint.

I d'una altre banda, la intenció de cobrir una necessitat personal a l'hora de proporcionar-me una eina amb la que pogués tenir emmagatzemada i disponible tota una sèrie d'informació dels llocs on podia trobar uns articles determinats o que tenien un interès especial per mi, prescindint de les incòmodes targetes de visita que sempre acabava perdent.

Des del inici de l'idea inicial, tenia clar que el projecte seria en una tecnologia per *smartphones* donada la seva naturalesa. La idea del targeter virtual perdia gran part del seu atractiu i utilitat si es feia en un suport web o com a aplicació d'escriptori per PC, ja que es perdria la immediatesa de poder consultar en qualsevol moment la informació i la possibilitat d'afegir un nou punt d'interès (POI) en el mateix moment que tenim les dades a mà, o si estem en el mateix POI i el poden posicionar directament mitjançant el GPS i emmagatzemar-lo.

Amb la implantació de *Google Maps* en els telèfons mòbils juntament amb l'accés assequible a la xarxa i la fiabilitat del *GPS*, cada cop s'utilitza més el navegador dels *smartphones* com a guia per moure's per la ciutat (jo mateix en faig un ús intensiu quan tinc dubtes sobre la ubicació d'un carrer o si hi puc arribar d'una forma més curta o quin transport em deixa més a prop). Per aquesta raó també em semblava interessant la idea de poder traçar una ruta fins a aquest punt des de la ubicació de l'usuari i, anant més enllà, que el programa fos capaç de crear una ruta entre diferents punts amb l'objectiu que, si has d'anar a diferents llocs que tens registrats, l'aplicació fos capaç de donar una ruta optima entre tots ells des de la posició de l'usuari.

1.2. OBJECTIUS

APRENDRE UN LLENGUATGE

Des de la sortida d'*Android* m'ha interessat con funcionava el llenguatge i la interacció amb l'usuari. Ja havia programat alguna cosa petita en *Windows Mobile*, però la diferència de tecnologies i l'experiència final que proporciona a l'usuari (més propera a la interacció amb una versió portàtil d'un PC amb Windows que no pas un *smartphone*) no té res a veure amb les possibilitats que ofereix *Android*. Així que el projecte em va donar la possibilitat d'entrar en un món que fa temps que volia explorar, però que per falta de temps i potser també motivació no havia tingut oportunitat.

Tanmateix, aprendre un llenguatge per a dispositius mòbils es avui dia obrir les portes a un mercat laboral amb moltes més possibilitats i en creixement constant.

RESOLDRE UNA NECESSITAT PERSONAL

Com he comentat en l'apartat anterior, el projecte neix bàsicament de la necessitat que tenia jo mateix com a usuari d'un *smartphone* . Per mi era molt important poder tenir tots els llocs que m'interessaven guardats i catalogats en una mateixa aplicació i poder prescindir de les targetes de visites, els *post-it,* o apuntar a qualsevol lloc una direcció per després perdre-la o trobar-la passat un temps i no saber per que l'havia apuntat. Després de investigar, cap de les aplicacions per *Android* del mercat em donava la solució concreta que anava buscant, ja que per norma general estan més orientades a donar-se a conèixer a altres usuaris, es a dir pujar dades a aplicacions per que puguin ser consultades per la resta d'usuaris i no puguin modificar-les. Mentre que la meva idea era que fos més semblant a una llista de contactes com els existents en els telèfons, on pogués editar, crear, esborrar, filtrar i en definitiva explotar la informació sense cap tipus de restricció. Tot això s'explica més detingudament a l'estudi de viabilitat, on s'examina una mostra de les diferents opcions del mercat per donar una visió concreta de la situació.

Un altre dels aspectes que em va fer decantar per aquesta opció es que la resta de programes que gestionen aquest tipus d'informació, al final no acaben sent més que *spam* i publicitat de llocs que realment poden no ser interessants a l'usuari, es a dir tens sobre-informació de llocs que ha pujat altre gent, o dels establiments han pagat per sortir i al final realment no tens el control per escollir que vols i no vols veure a la teva aplicació.

1.3. ESTRUCTURA DE LA MEMÒRIA

Aquest document es compon principalment de 7 parts importants, les quals s'expliquen breument a continuació.

L'ESTUDI DE VIABILITAT

En aquest apartat es farà un estudi del programa, la funcionalitat que s'implementarà i la viabilitat de dur a terme l projecte. Examinarem diferents solucions que ofereix el mercat en la actualitat per poder oferir una vista més detallada de la realitat i contrastarem els beneficis/inconvenients de les diferents solucions . A partir d'aquest estudi, escollirem la opció més adient i continuarem analitzant els requisits funcionals i no funcionals de l'aplicació, els riscos i els recursos necessaris per dur-lo a terme per tal d'establir una planificació i un pressupost final del cost de l'aplicació.

FONAMENTS TEÒRICS

Aquí s'analitzarà l'estat de l'art de les tecnologies emprades al projecte. Explicarem una mica la història de cada tecnologia i les característiques per les quals s'ha escollit envers les altres opcions disponibles del mercat.

ANÀLISI

En aquest capítol presentarem dures parts diferenciades. Per una banda, els recursos que participen en el projecte, els tipus d'usuaris que podran accedir a l'aplicació i els seus rols. Per altra banda, definirem els casos d'us del projecte explicant cada acció que por fer cada usuari. Adjuntarem també una petita llegenda del significat de cada camp del document per a poder comprendre millor la seva estructura.

Disseny i Implementació

El disseny està dedicat a l'estudi de la implantació de l'aplicació. Es definirà l'estructura de la Base de Dades a nivell de taules i camps i es farà un petit anàlisi del perquè d'aquesta solució. També es definiran les classes de l'aplicació que conformarà l'esquelet en el qual es sostindrà tota la lògica del programa. Tot seguit es comentarà com funciona el patró de disseny MVC (model-vista-controlador) i es mostrarà un exemple de cadascuna de les parts d'aquest patró.

La implementació mostrarà una visió global del projecte. Explicarem com s'ha estructurat i com s'han definit els processos a partir del patró MVC per tal de dur a terme el projecte. També aprofitarem per exposar alguns dels problemes més significatius que han sorgit mentre es desenvolupava l'aplicació.

Conclusions, Desviacions i Millores

Aquest apartat està dedicat a l'autoavaluació del projecte, a l'anàlisi de tot el procés que s'ha dut a terme d'una manera més personal i subjectiva. Reflexionarem sobre com es va idear l'aplicació en un principi i com ha anat evolucionant fins al producte final que va lligat a aquesta memòria.

En l'últim punt comentarem les possibles millores que es poden fer en el programa tant funcionals com estètiques. Mirarem com es podria ampliar el projecte amb noves funcionalitats en un futur, tant si s'han quedat pendents per qualsevol motiu, com si en el procés de creació han aparegut noves idees per millorar l'aplicació.

2. ESTUDI DE VIABILITAT

2.1. INTRODUCCIÓ

En aquest apartat s'estudiarà la viabilitat i implementació d'un aplicació per a la plataforma Android que gestioni, a mode d'agenda personal, els punts d'interès - PDI en català o POI en anglès - d'un usuari.

Es presentarà una petita descripció sobre que tractarà el projecte, les paraules clau associades a aquest, els objectius a complir i els beneficis obtinguts una vegada finalitzada l'aplicació.

2.2. OBJECTIUS

L'objectiu del projecte es ben senzill. La seva finalitat és donar una eina a l'usuari d'un *smartphone* amb sistema operatiu Android, la possibilitat d'emmagatzemar i gestionar els Punts d'Interès que desitgi.

Per Punt D'Interès (d'ara endavant POI) entenem un punt que algú pot trobar útil o interessant, es representa en una ubicació de la Terra mitjançant unes coordenades donades per una latitud i longitud concretes.

En el nostre cas pot ser la direcció d'una botiga o la d'un restaurant que ens cridi l'atenció al passar per davant.

Amb la creixent implantació dels *smartphones* en la societat degut als preus cada cop més accessibles per part de les companyies de telecomunicacions, neix una nova forma per unificar i gestionar tot el contingut personal en un únic dispositiu. Cada cop més es tornen obsolets els mitjans tradicionals com els targeters o les targes de visita i requerim d'aplicacions que puguin substituir aquestes necessitats.

El programa neix de la necessitat de poder tenir emmagatzemats, classificats i a la mà tots els llocs d'importància que l'usuari necessiti, a l'estil d'una agenda de contactes.

En qualsevol moment, l'usuari pot accedir a ells, tenir una informació detallada del lloc i si es necessari, mitjançant la interacció amb l'aplicació *Google Maps*, poder geoposicionar el lloc i traçar una ruta fins el POI.

DESCRIPCIÓ DELS OBJECTIUS

- **Guardar la fitxa d'un POI:**
  - nom, direcció, coordenades, telèfon, observacions, per visitar, que tenen/venen...

- **Catalogar els POIs per tipus i característiques:**
  - Segons el tipus d'establiment: botiga, restaurant...

- **Poder afegir un POI:**
  - mitjançant la posició: En qualsevol punt del carrer poder fixar-lo amb GPS (+precís,-ràpid) o la Xarxa (-precís,+ràpid).
  - mitjançant la direcció: Poder geoposicionar l'establiment.

- **Cerca els llocs per paràmetre:**
  - Poder cercar per tipus de establiment, localització...
    Per exemple, filtrar 'per visitar' mostra on no has anat encara.

- **Rastrejador de establiments per tipus:**
  - Quins establiments hi ha del tipus X a N metres d'on soc.

- **Poder traçar una ruta des d'on es troba l'usuari fins el POI**

- **Llista de compres:** Que vols comprar?
  - Mostrar una llista de llocs que ofereixen el que es busca.

- **Enviar i compartir els POIs a una direcció de correu**

- **Poder copiar el POI com a contacte de Gmail**

- **Constructor de rutes:**
  - Trobar el camí mes curt entre N establiments a visitar (problema del viatjant).

- **Escaneig de targetes de visita:**
  - Poder fer una lectura de les dades d'una tarja amb la càmera del dispositiu i emmagatzemar-la com a POI.

- **Lector de codis QR amb la informació del POI:**
  - Poder escanejar un codi QR a la porta d'un establiment i obtenir tota la informació necessària per ubicar el POI.

TAULA DE PRIORITATS

A continuació es detalla una taula amb els objectius catalogats segons la prioritat que s'estableix per al projecte.

	Objectius	**Prioritat**
1	Guardar la fitxa d'un POI	Crític
2	Catalogar els POIs per tipus	Crític
3	Afegir un POI	Crític
4	Cerca els llocs per paràmetre	Prioritari
5	Traçar una ruta des d'on es troba l'usuari fins el POI	Prioritari
6	Llista de compres. Que vols comprar?	Secundari
7	Enviar els POIs per correu	Secundari
8	Copiar el POI com a contacte de Gmail	Opcional
9	Constructor de rutes	Secundari
10	Escaneig de targetes de visita	Opcional
11	Lector de codis QR	Opcional

2.3. ESTAT DE L'ART

ALTERNATIVES I SELECCIÓ DE LA SOLUCIÓ

En aquest apartat estudiarem les possibles solucions que actualment ofereix el mercat i avaluarem la idoneïtat d'aquestes, si s'ajusten als nostres requisits i si són viables.

Es van estudiar vàries aplicacions, però per motius d'espai i de l'objectiu de la memòria s'han retallat i s'han deixat dues aplicacions que representen els dos grans grups que hi ha al mercat: les aplicacions per gestió personal i les col·laboratives que ofereixen informació de la comunitat d'usuaris i patrocinadors.

ALTERNATIVA 1: ADQUISICIÓ I ADAPTACIÓ D'UN PROGRAMA SIMILAR

- **Aplicació *MyPoi* for Android**

(http://www.androlib.com/android.application.com-scoobler-*MyPoi*-jBmn.aspx)

Funcionalitats:

- *Guardar POIs*
- *Gestionar POIs*
- *Catalogar POIs per un tipus comú*
- *Ruta fins al POI*

Costos:

- *Android Open License*
  - *Versió gratuïta:*
    - *Limitació a només 5 POIs*
  - *Versió de pagament:*
    - *1,5€*

Aquesta va ser una de les primeres opcions que es van contemplar com a solució per al nostre cas. En un principi semblava que complia les condicions, però després d'un estudi més en detall es va veure que les característiques eren insuficients. En tant que cobrien les funcionalitats bàsiques (manteniments dels POIs) es queda curt en la resta de possibilitats. No hi ha opció a catalogar per més d'una característica i tret del nom, la descripció i la posició no es pot informar cap altre detall. La representació del POI en aquesta aplicació és massa genèrica i no ofereix pràcticament cap opció de detallar el punt. Tampoc es possible traçar rutes entre diferents POIs ni fer cerques específiques.

ALTERNATIVA 2: GESTOR DE POIS MITJANÇANT LES EINES *GOOGLE MAPS*

Aplicació *Google Maps* for Android

(http://www.*Google*.com/mobile/maps/)

Funcionalitats:

- *Guardar POIs*

- *Gestionar POIs*
- *Repositori de POIs*
- *Ruta fins al POI*
- *Opció d'afegir múltiples capes cartogràfiques sobre els POIs*
- *Interacció amb altres aplicacions Google (Buzz,Latitude,Navigator)*

Costos:

- Gratuït
- *Google Maps* Android Open License

Google Maps es l'opció que a simple vista semblaria més adient respecte la resta, però després d'un anàlisi podem comprovar que té un ús molt concret. Es poden crear POIs, però no es poden esborrar ni modificar lliurement, consta d'un sistema de 'capes' que superposa els teus POIs als que hi ha per defecte en *Google Maps*, per la qual cosa sempre veus punts que no t'interessen. Això es degut al seu caràcter publicitari i per tant no es pot eliminar les empreses que han 'pagat' per sortir al mapa.

ALTERNATIVA 3: DESENVOLUPAMENT D'UN GESTOR DE POIS

Funcionalitats:

- S'ajusta als requisits del client.
- Necessitat de preveure la possible evolució i introducció de futures millores.
- Ajustable als recursos disponibles per la entitat

Crear una aplicació a mida del client no té cap inconvenient, ja que els criteris i la funcionalitat son els que el client estableix. L'únic impediment a la seva acceptació pot ser el cost elevat, que analitzarem més endavant, en vers les altres opcions.

SOLUCIÓ PROPOSADA

Avaluació de les alternatives proposades:

	Costos Adquisició	*Costos Adaptació*	*Nous Recursos*	*Suport*	*Nivell Integració*	*Complexitat*	*Formació*
MyPoi	1,50 €	Alt	No cal	No hi ha	Alt	Baixa	No hi ha
Google Maps	0,00 €	Baix	No cal	Fòrum Web	Alt	Mitja	Es desconeix
Gestor de POI	0,00 €	Segons projecte/ Pressupost	No cal	Inclòs al projecte	Alt	Alta	Inclosa en el projecte

Justificació de la solució:

De tots el candidats que es van cercar a *l'Android Market* hem presentat dues opcions que defineixen els dos grans grups en que es podrien dividir. Per una banda les aplicacions que tenen un suport físic al dispositiu, com es l'exemple de l'aplicació *MyPoi* i el grup d'aplicacions que estan gestionades a la xarxa (*Cloud Computing*) com es el cas de *Google Maps*.

Cap de les altres dues opcions ofereixen tots el requisits que demana el client.

Per part de l'aplicació *MyPoi*, malgrat que el seu cost es baix, trobem insuficient les característiques que ofereix, com per exemple no poder catalogar un POI amb més d'una paraula clau. Per tant, com que la modificació d'aquesta aplicació es inviable, es descarta.

En el cas de *Google Maps*, trobem que la integració es immediata (no és necessari cap software auxiliar), però tampoc ofereix les opcions que l'usuari demana, ja que aquest tipus d'aplicació està més destinat a ser una espècie de *pàgines grogues* interactives per publicitar empreses i llocs d'interès, sense permetre a l'usuari tenir total control del que pot veure i filtrar. Mentre que els seus punts forts, com la integració amb altres aplicacions de la mateixa empresa, no son necessaris per l'usuari i per tant no son avaluables respecte a la utilitat de l'aplicació en vers el client.

Un cop comparades les 3 opcions, podem concloure que la alternativa número 3, la creació d'una aplicació pròpia per gestionar i mantenir una llista de POIs, es la opció més adient.

2.4. ESPECIFICACIONS I REQUISITS

En aquest capítol s'exposaran els requisits del sistema, tant els funcionals com el no funcionals i s'estudiarà la seva importància dins del projecte juntament amb els objectius.

TIPOLOGIA I PARAULES CLAU

Tipologia: Desenvolupament.

Paraules clau: Android, POI, PDI, targeter virtual, GPS, geoposicionament.

Descripció: Aprofitant les noves capacitats dels telèfons mòbils actuals, es pretén donar solució al problema de tenir sempre a mà la llista dels llocs més habituals o amb un interès especial per a l'usuari, proporcionant un targeter virtual i alliberant-lo de la obligació de portar a sobre un targeter tradicional.

Al mateix temps, gràcies a la tecnologia *GPS* actual, es podrà geoposicionar i traçar una ruta des d'on es trobi l'usuari fins el POI de destí.

ESTUDI DE LA SITUACIÓ ACTUAL

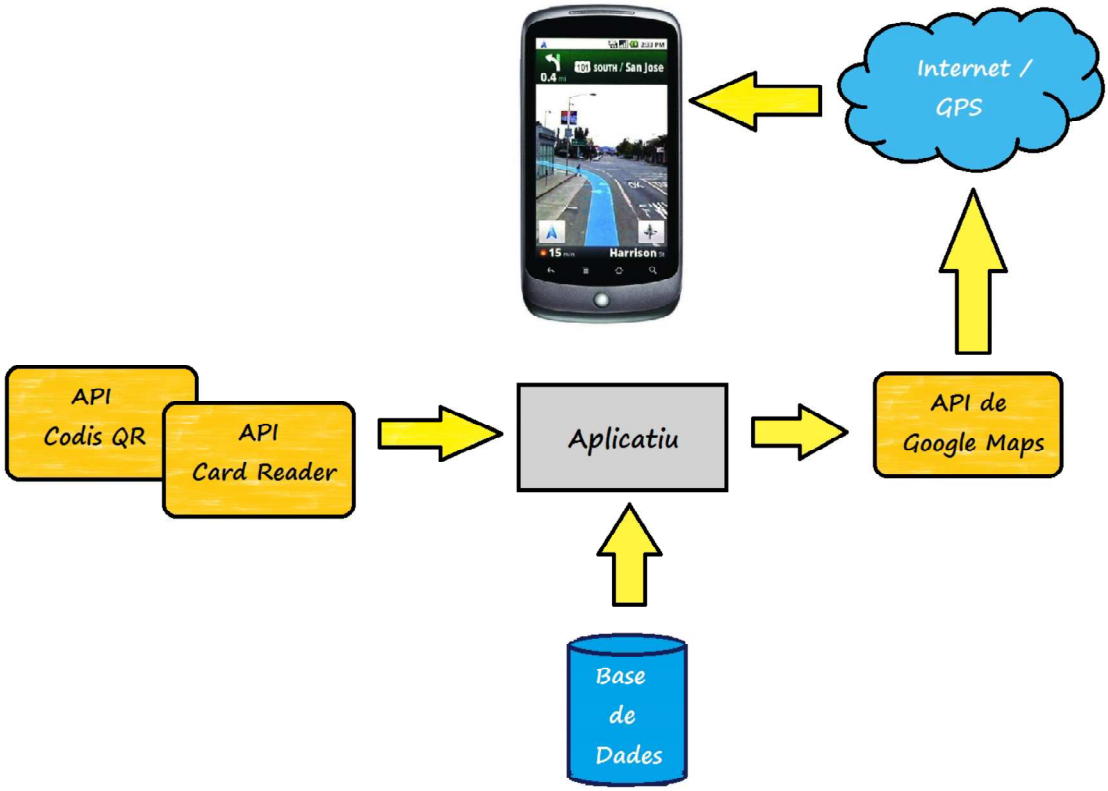
Actualment no hi ha cap aplicació prèvia i per tant, aquest projecte no parteix de cap altre projecte anterior.

L'aplicació es començarà des de zero i el que avaluarem en aquest apartat es si la proposta es por dur a terme o si la tecnologia actual no permet implantar alguns dels requisits de l'aplicació.

CONTEXT

S'ha escollit *Google Maps* com a servei de en línia entre tota la oferta actual, tant de pagament con gratuïta, degut a que en la seva versió gratuïta ja incorpora totes les eines necessàries per que la nostra aplicació pugui funcionar. També s'ha escollit pel fet que és de la mateixa companyia que dona suport al sistema operatiu del nostre terminal i per tant, la interacció i funcionament entre *l'API* i el SO està més integrat.

DIAGRAMA QUE DESCRIU LA LÒGICA DE LA SITUACIÓ



2.4.1. ESPECIFICACIONS FUNCIONALS

LLISTA DE REQUISITS FUNCIONALS DE L'APLICACIÓ:

- Manteniment (altes, baixes, modificacions) de dades dels POIs.
- Manteniment de les dades de l'usuari de l'aplicació.
- Manteniment de la catalogació de tipus i característiques dels POIs.
- Manteniment de les llistes de POIs per rutes.
- Gestió (altes, baixes, modificacions) de dades dels POIs.
- Gestió de les dades de l'usuari de l'aplicació.
- Gestió de la catalogació de tipus i característiques dels POIs.
- Gestió de les llistes de POIs per rutes.
- Control d'accés a l'aplicació.
- Interacció amb el gestor de correu per enviar POIs.
- Interacció amb *Google Maps*.
- Interacció amb el lector de codis QR i imatges.
- Sistema d'ajuda en línia.
- Còpies de seguretat i recuperació de dades.

2.4.2. ESPECIFICACIONS NO FUNCIONALS

LLISTA DE REQUISITS NO FUNCIONALS:

- Compliment de la LOPD pel que fa referència als fitxers de dades i als drets dels clients.
- Normalització de la base de dades i accés segons l'estàndard SQL 99 (ISO/IEC 9075:1999).
- Els recursos utilitzats per l'aplicació han d'estar ajustats a la capacitat del terminal mòbil *Samsung Galaxy Nexus.*
- Tolerància a errades i a accions incorrectes.
- Seguretat i consistència de les dades: la base de dades no es podrà accedir físicament.

2.4.3. CATALOGACIÓ I PRIORITZACIÓ DELS REQUISITS

REQUISITS FUNCIONALS

	RF 1	RF 2	RF 3	RF 4	RF 5	RF 6	RF 7	RF 8	RF 9	RF 10	RF 11	RF 12	RF 13	RF 14
Crític	X	X	X	X	X	X	X	X	X		X			
Prioritari										X			X	X
Opcional												X		

REQUISITS NO FUNCIONALS

	RNF 1	RNF 2	RNF 3	RNF 4	RNF 5
Crític	X	X			X
Prioritari			X	X	
Opcional					

RELACIÓ ENTRE ELS REQUISITS I ELS OBJECTIUS

	RF 1	RF 2	RF 3	RF 4	RF 5	RF 6	RF 7	RF 8	RF 9	RF 10	RF 11	RF 12	RF 13	RF 14	RN F1	RN F 2	RN F 3	RN F 4	RN F 5
O1	X		X	X	X		X	X						X	X	X		X	X
O2	X		X	X	X		X	X					X		X	X			X
O3	X		X	X	X		X	X					X		X	X		X	X
O4	X		X		X		X											X	X
O5								X			X				X		X		
O6	X		X		X	X	X							X					
O7		X			X				X		X		X		X				
O8		X			X									X	X				
O9	X		X		X		X	X	X		X		X	X			X	X	
O10	X		X		X		X					X	X				X		
O11	X		X		X		X					X	X				X		

2.5. PLANIFICACIÓ

Aquest document recull el Pla del projecte, és a dir, el conjunt d'activitats que permeten desenvolupar, executar i controlar el projecte.

El document inclou les tasques i punts de control del projecte, els recursos del projecte, el calendari del projecte, l'avaluació de riscos i el pressupost del projecte.

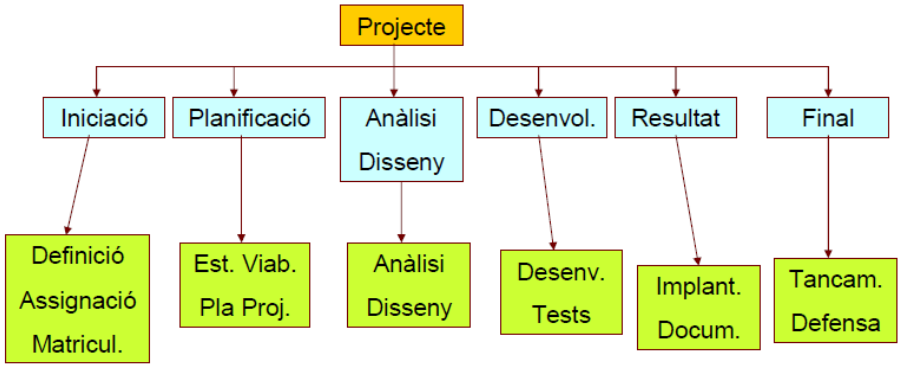
2.5.1. WBS (WORK BREAKDOWN STRUCTURE)

FASES I ACTIVITATS DEL PROJECTE

Llista de totes les fases, activitats i tasques del projecte:

Fases	***Descripció***
Iniciació	Fase d'iniciació. Inclou les activitats definició del projecte, assignació i matriculació
Planificació	Inclou l'Estudi de Viabilitat i el Pla del Projecte.
Anàlisi	Anàlisi de requisits funcionals i no funcionals juntament amb l'arquitectura del sistema.
Disseny	Inclou el disseny de la capa de dades, de control i l' interfície amb el disseny dels tests.
Desenvolupament	Fase de desenvolupament de l'aplicació.
Test i proves	Fase de prova del sistema. Inclou tests unitaris i d'integració.
Implantació	L'aplicació s'instal·la en el seu entorn real. Inclou la formació d'usuaris.
Generació de documents	Fase de documentació del projecte. Inclou manuals i memòria del projecte.
Tancament del projecte	Fase de tancament. El director del projecte signa l'acceptació i tancament del projecte.
Defensa del projecte	Defensa del projecte davant la comissió.

DIAGRAMA WBS



MILESTONES

Nom	***Descripció***	***Data***
Iniciació	Matriculació	01/11/11
Est. Viabilitat	Aprovació	13/12/11
Pla del Projecte	Aprovació	13/07/12
Anàlisi	Aprovació	24/07/12
Disseny	Aprovació	27/07/12
Tancament	Acceptació	20/09/12
Defensa	Avaluació	25/09/12

2.5.2. RECURSOS DEL PROJECTE

A continuació es detallarà la llista de recursos assignats al projecte, tant els humans, com els d'equipament, com els materials i la valoració d'aquests recursos.

Recursos humans

	Valoració
Cap de projecte	65 €/h
Analista	50 €/h
Programador	20 €/h
Tècnic proves	10 €/h

Recursos materials

Els recursos materials seran els disponibles pel client.

El desenvolupament es farà mitjançant programari amb llicència lliure, per tant no hi haurà costos addicionals per utilitzar les eines detallades al punt 1, tant les pròpies de desenvolupament com les de *'third-parties'.*

CALENDARI DELS RECURSOS

Recursos humans que s'utilitzaran en tot el projecte

- **Cap de projecte:** Iniciació, Planificació, Generació de documents, tancament i defensa. Punts de control.
- **Analista:** Anàlisi i disseny, Implantació i Punts de control d'anàlisi, disseny i desenvolupament.
- **Programador:** Disseny, Desenvolupament i Test. Parcialment en l' implantació.
- **Tècnic de proves:** Fase de test.

Recursos materials

S'utilitzaran principalment durant les fases de desenvolupament, test i implantació.

CALENDARI DEL PROJECTE

- **Descripció general del calendari del projecte:**

El projecte es desenvoluparà del Juny de 2012 a l'Agost de 2011 amb una dedicació de 20 hores setmanals aproximadament.

El total de dies dedicats al projecte serà de 109 dies.

- Data començament: 25 de Juny de 2012
- Data finalització: 21 d'Agost de 2012

- **Eines de planificació i control:**
    - Microsoft Project
    - Mercurial

DEPENDÈNCIES

Totes les fases es desenvolupen utilitzant un model lineal. Per tant, cada fase no es comença fins que no s'ha completat la fase anterior.

En la fase de desenvolupament es preveu un model àgil de tal manera que el disseny, el desenvolupament i el test segueixin un model iteratiu.

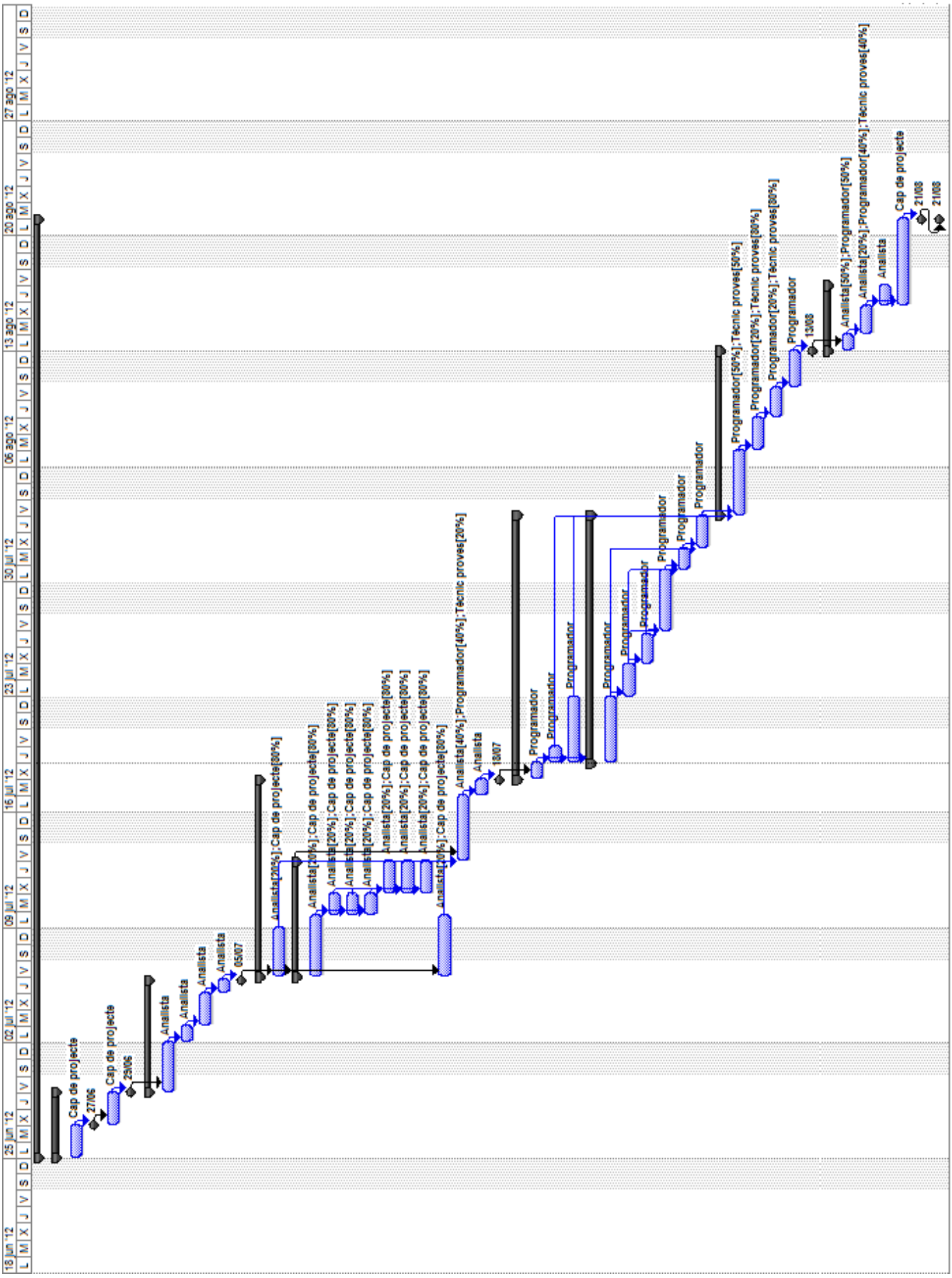
La fase de generació de documents es preveu al final perquè inclourà els documents elaborats durant el desenvolupament del projecte: inici, estudi de viabilitat, pla de projecte,...

2.5.3. QUADRE DE TASQUES DEL PROJECTE

ID	Nom de la tasca	Duració	Recursos	Predecessores
1	**-Inici del projecte: assignació i matriculació del projecte**	109 dies		
2	**-Planificació**	9 dies		
3	Estudi de viabilitat	5 dies	-Cap de projecte	
4	Aprovació Estudi Viabilitat (Punt de control)	0 dies	-Cap de projecte	3
5	Pla del projecte	4 dies	-Cap de projecte	4
6	Aprovació Pla del Projecte (Punt de control)	0 dies	-Cap de projecte	5
7	**-Anàlisi de l'aplicació**	13 dies		
8	Anàlisi de requisits (cassos d'ús)	4 dies	-Analista	6
9	Anàlisi de dades (base de dades)	3 dies	-Analista	8
10	Anàlisi de la seguretat i legalitat	4 dies	-Analista	9
11	Documentació de l'anàlisi	2 dies	-Analista	10
12	Aprovació de l'anàlisi (Punt de control)	0 dies	-Cap de projecte -Analista	11
13	**-Disseny de l'aplicació**	22 dies		
14	Disseny de la base de dades	3 dies	-Analista[20%] -Cap de projecte[80%]	12
15	**-Disseny modular de l'aplicació**	15 dies		12
16	Manteniment (altes, baixes, modificacions)	6 dies	-Analista[20%] -Cap de projecte[80%]	
17	Gestió de la catalogació de tipus i característiques dels POIs	6 dies	-Analista[20%] -Cap de projecte[80%]	16
18	Gestió de les dades de l'usuari de l'aplicació	6 dies	-Analista[20%] -Cap de projecte[80%]	16
19	Gestió de les llistes de POIs per rutes	6 dies	-Analista[20%] -Cap de projecte[80%]	16
20	Interacció amb el gestor de correu	6 dies	-Analista[20%] -Cap de projecte[80%]	18
21	Interacció amb *Google Maps*	6 dies	-Analista[20%] -Cap de projecte[80%]	18;19
22	Disseny de l'interfície	6 dies	-Analista[20%] -Cap de projecte[80%]	12
23	Disseny de les proves (test)	4 dies	-Analista[40%] -Programador[40%] -Tècnic proves[20%]	14;15;22
24	Documentació del disseny	3 dies	-Analista	23
25	Aprovació del disseny (Punt de control)	0 dies	-Cap de projecte -Analista	24
26	**-Desenvolupament de l'aplicació**	33 dies		
27	Preparació entorn de desenvolupament	3 dies	-Programador	25
28	Configuració base de dades	2 dies	-Programador	27
29	Mòdul d'adquisició de dades i funcionalitats de l'aplicació	5 dies	-Programador	27
30	**-Desenvolupament de l' d'usuaris**	30 dies		27
31	Gestió de la catalogació de tipus i característiques dels POIs	5 dies	-Programador	
32	Gestió de les dades de l'usuari de l'aplicació	5 dies	-Programador	31
33	Gestió de les llistes de POIs per rutes	5 dies	-Programador	32
34	Interacció amb el gestor de correu	5 dies	-Programador	32;33
35	Interacció amb *Google Maps*	5 dies	-Programador	32;32;34
36	**-Test i proves**	16 dies		

37	Proves unitàries	5 dies	-Programador[50%] -Tècnic proves[50%]	28;29
38	Proves d'integració	5 dies	-Programador[20%] -Tècnic proves[80%]	37
39	Proves d'estrès (incidències, riscos)	4 dies	-Programador[20%] -Tècnic proves[80%]	38
40	Documentació de desenvolupament i test	2 dies	-Programador	39
41	Aprovació del desenvolupament i proves (Punt de control)	0 dies	-Analista -Cap de projecte -Programador	40
42	**-Implantació**	11 dies		
43	Instal·lació	2 dies	-Analista[50%] -Programador[50%]	41
44	Proves reals	4 dies	-Analista[20%] -Programador[40%] -Tècnic proves[40%]	43
45	Formació d'usuaris	5 dies	-Analista	44
46	Generació de documents (memòria del projecte)	10 dies	-Cap de projecte	44
47	Tancament del projecte	0 dies	-Cap de projecte	46
48	Defensa del projecte	0 dies	-Cap de projecte	47

2.5.4. DIAGRAMA DE GANTT



2.6. AVALUACIÓ DE RISCOS

En aquest apartat, indicarem, per a cada risc, el sistema o subsistema que origina del risc, una descripció d'aquest i l'efecte que tindria sobre el projecte amb la seva possible solució.

Es tindran en compte tant els riscos inherents al problema com els derivats de la solució proposada.

LLISTA DE RISCOS

ID	Descripció:
1	**Planificació temporal optimista:** Pla de projecte. No s'acaba en la data prevista, augmenten els recursos.
2	**Manca alguna tasca necessària:** Pla de projecte. No es compleixen els objectius del projecte.
3	**Pressupost poc ajustat:** Pla de projecte. Menys qualitat, pèrdues econòmiques.
4	**Canvi de requisits:** estudi de viabilitat, anàlisi. Endarreriment en els desenvolupament i resultat.
5	**Equip del projecte massa reduït:** Pla de projecte. Endarreriment en la finalització del projecte, no es compleixen els objectius del projecte.
6	**Eines de desenvolupament inadequades:** desenvolupament. Endarreriment en la finalització del projecte, menys qualitat, ...
7	**Dificultat per accedir als *stakeholders*:** estudi de viabilitat, anàlisi, proves, formació. Manquen requisits o són inadequats, endarreriments, insatisfacció usuaris.
8	**No es fa correctament la fase de test:** desenvolupament, implantació. Manca de qualitat, deficiències en l'operativa, insatisfacció usuaris, pèrdua econòmica.
9	**Incompliment d'alguna norma, reglament o legislació**: en qualsevol fase. No es compleixen els objectius, repercussions legals.
10	**Manca d'adopció de mesures de seguretat:** estudi de viabilitat, anàlisi, desenvolupament. Pèrdua d'informació, incompliment legal, pèrdues econòmiques.
11	**Abandonament del projecte abans de la finalització:** en qualsevol fase. Pèrdues econòmiques, frustració.

CATALOGACIÓ DE RISCOS

Risc	Probabilitat	Impacte
R1	Alta	Crític
R2	Alta	Crític
R3	Alta	Crític
R4	Alta	Marginal
R5	Alta	Crític
R6	Baixa	Crític
R7	Baixa	Crític
R8	Alta	Crític
R9	Mitjana	Crític
R10	Alta	Crític
R11	Mitjana	Catastròfic

PLA DE CONTINGÈNCIA

	Solució que cal adoptar
R1	Ajornar alguna funcionalitat, afrontar possibles pèrdues, fer una assegurança.
R2	Revisar el Pla de Projecte, modificar la planificació.
R3	Renegociar amb el client, afrontar possibles pèrdues, fer una assegurança.
R4	Renegociar amb el client, ajornar funcionalitat, modificar planificació i pressupost.
R5	Demanar un ajornament, negociar amb el client, afrontar pèrdues.
R6	Millorar la formació de l'equip, preveure eines alternatives, millorar la qualitat.
R7	Fixar un calendari de reunions, millorar el contacte amb el client.
R8	Dissenyar els test amb antel·lació, realitzar tests automàtics, negociar contracte de manteniment, donar garanties, afrontar pèrdues econòmiques.
R9	Revisar les normes i legislació, consultar un expert, afrontar possibles repercussions penals.
R10	Revisar la seguretat en cada fase, aplicar polítiques de seguretat actives.
R11	No té solució.

2.7. PRESSUPOST

ESTIMACIÓ COST DE PERSONAL

Costos de personal imputables directament al projecte:

	Hores	Cost
Cap de projecte	160,2h	10.413,00 €
Analista	99h	4.950,00 €
Programador	151,5h	3030,00 €
Tècnic de proves	36,3h	363,00 €
	Total:	**18.756,00 €**

ESTIMACIÓ COST DELS RECURSOS

Amortització dels recursos propis del projecte:

	Cost amortització	Cost unitari	Període amortització	Període utilització
PC programador	300	1200	24 mesos	6 mesos
Dispositiu Galaxy Nexus per proves	95	380	24 mesos	6 mesos
MSProject	115	460	24 mesos	6 mesos
Total:	**510,00 €**			

ESTIMACIÓ COST DE LES ACTIVITATS

ID	Nom de la tasca	Hores	Cost
1	**-Inici del projecte: assignació i matriculació del projecte**	447 hores	18.756,00 €
2	**-Planificació**	27 hores	1.755,00 €
3	Estudi de viabilitat	15 hores	975,00 €
4	Aprovació Estudi Viabilitat (Punt de control)	0 hores	0,00 €
5	Pla del projecte	12 hores	780,00 €
6	Aprovació Pla del Projecte (Punt de control)	0 hores	0,00 €
7	**-Anàlisi de l'aplicació**	39 hores	1.950,00 €
8	Anàlisi de requisits (cassos d'ús)	12 hores	600,00 €
9	Anàlisi de dades (base de dades)	9 hores	450,00 €
10	Anàlisi de la seguretat i legalitat	12 hores	600,00 €
11	Documentació de l'anàlisi	6 hores	300,00 €
12	Aprovació de l'anàlisi (Punt de control)	0 hores	0,00 €
13	**-Disseny de l'aplicació**	150 hores	8.808,00 €
14	Disseny de la base de dades	9 hores	558,00 €
15	**-Disseny modular de l'aplicació**	105 hores	6.510,00 €
16	Manteniment (altes, baixes, modificacions)	15 hores	930,00 €
17	Gestió de la catalogació de tipus i característiques dels POIs	15 hores	930,00 €
18	Gestió de les dades de l'usuari de l'aplicació	15 hores	930,00 €
19	Gestió de les llistes de POIs per rutes	15 hores	930,00 €
20	Interacció amb el gestor de correu	15 hores	930,00 €
21	Interacció amb *Google Maps*	15 hores	930,00 €
22	Disseny de l'interfície	15 hores	930,00 €
23	Disseny de les proves (test)	15 hores	930,00 €
24	Documentació del disseny	12 hores	360,00 €
25	Aprovació del disseny (Punt de control)	9 hores	450,00 €
26	**-Desenvolupament de l'aplicació**	0 hores	0,00 €
27	Preparació entorn de desenvolupament	120 hores	2.400,00 €
28	Configuració base de dades	9 hores	180,00 €
29	Mòdul d'adquisició de dades i funcionalitats de l'aplicació	6 hores	120,00 €
30	**-Desenvolupament de l' d'usuaris**	15 hores	300,00 €
31	Gestió de la catalogació de tipus i característiques dels POIs	90 hores	1.800,00 €
32	Gestió de les dades de l'usuari de l'aplicació	15 hores	300,00 €
33	Gestió de les llistes de POIs per rutes	15 hores	300,00 €
34	Interacció amb el gestor de correu	15 hores	300,00 €
35	Interacció amb *Google Maps*	15 hores	300,00 €
36	**-Test i proves**	15 hores	300,00 €
37	Proves unitàries	15 hores	300,00 €
38	Proves d'integració	48 hores	669,00 €
39	Proves d'estrès (incidències, riscos)	15 hores	225,00 €
40	Documentació de desenvolupament i test	15 hores	180,00 €
41	Aprovació del desenvolupament i proves (Punt de control)	12 hores	144,00 €

42	**-Implantació**	6 hores	120,00 €
43	Instal·lació	0 hores	0,00 €
44	Proves reals	33 hores	1.224,00 €
45	Formació d'usuaris	6 hores	210,00 €
46	Generació de documents (memòria del projecte)	12 hores	264,00 €
47	Tancament del projecte	15 hores	750,00 €
48	Defensa del projecte	30 hores	1.950,00 €

RESUM I ANÀLISI COST BENEFICI

Encara que el cost total del projecte és alt, aquesta es l'única solució viable, ja que les altres propostes no complien els requisits que l'usuari desitjava que tingués l'aplicació. Més que els beneficis econòmics s'han de considerar els beneficis no econòmics derivats de la centralització de tota la informació sobre els punts d'interès d'un usuari en el seu dispositiu mòbil i la possibilitat de poder gestionar-la com decideixi, així com la possibilitat de poder intercanviar amb altres usuaris els POIs de forma immediata .

OFERTA DE LA SOLUCIÓ PROPOSADA

- *Cost de desenvolupament del projecte* *18.756 €*
- *Cost d'amortització del material* *510 €*

Total: ***19.266 €***

2.8. VALORACIÓ

Com a deficiència trobada mentre es feia l'estudi de viabilitat, podem assenyalar que el mòdul pel qual l'aplicació podria escanejar codis QR i targetes de visita mitjançant un sistema de reconeixement d'imatge es totalment inviable en aquests moments.

La tecnologia per poder dur a terme un reconeixement amb èxit es troba encara en fase de creació. Per part de *Google* es pretén treure una API per desenvolupadors del seu escàner d'imatge *Google Google*s en breu. I per part d'un altre grup, *FutureAPI*, també hi ha previst un llançament.

Ara mateix només hi ha una companyia que té una API en el mercat, IQEngines, però només funciona amb adreces en format americà.

Per tant podem dir que de moment aquesta fase quedaria fora del projecte inicial, mentre no hi hagi un mitjà fiable per extreure una direcció d'una targeta de visita, ja que la possibilitat de crear una API pel nostre compte es descarta totalment degut a l'elevat cost econòmic y de temps.

Beneficis:

- Disposar de tots els llocs d'interès per l'usuari en forma de POIs.
- Afegir un POI en qualsevol moment i incorporar-lo a la base de dades.
- Catalogació i ordenació de POIs per tipus i característiques.
- Compartir POIs amb altres usuaris mitjançant correu.
- Possibilitat de traçar una ruta des d'on es trobi l'usuari fins el POI de destí.
- Crear llistes.

Inconvenients:

- Inversió important.
- Recel per part dels usuaris.
- Necessitat d'un període de formació.

2.9. CONCLUSIONS

Com ja s'ha comentat, el benefici més clar es el poder tenir y gestionar totes les adreces dels llocs que son importants per l'usuari en comptes de tenir la cartera plena de papers i targetes de locals, o anar carregant una llibreta o targeter per poder disposar d'aquesta informació.

Poder fer cerques ràpides i complexes per característiques concretes mitjançant la inclusió de les 'característiques' i les 'paraules clau'.

El fet de poder traçar una ruta des d'on es troba fins el seu POI facilita molt el saber com arribar al lloc sense tenir que consultar en cap mapa el numero i carrer on es troba.

Poder guardar el teu POI en qualsevol moment sense perdre temps amb la captura de la ubicació actual i poder compartir-lo amb altres usuaris. Com a exemple, hi ha dos casos molt comuns que ens podem trobar:

Quan, caminant pel carrer, passes per davant d'un local de copes/restaurant que et crida l'atenció i que pot estar be, però no tens res on apuntar-ho i penses "ja me'n recordaré després" i resulta que quan arribes a casa ja no saps ni el carrer on era.

Un altre cas seria quan veus una botiga que ven articles que li poden interessar a un amic, doncs amb l'aplicació podries, en pocs segons, guardar la direcció, què és el que has vist i enviar-ho al teu amic.

Un altre aspecte interessant es el fet de poder planificar una ruta entre els diferents POIs (mitjançant el problema del viatjant) i poder calcular el camí més curt entre ells per estalviar temps i recursos.

3. FONAMENTS TEÒRICS

3.1. ANDROID

BREU HISTÒRIA DEL LLENGUATGE

Android és un sistema operatiu mòbil basat en Linux enfocat per ser utilitzat en dispositius mòbils com *tablets* i *smartphones*. Inicialment va ser desenvolupat per **Android Inc**. Però *Google* va comprar la firma l'any 2005 i va passar a ser desenvolupat per ***Google*** mitjançant la ***Open Handset Alliance***.

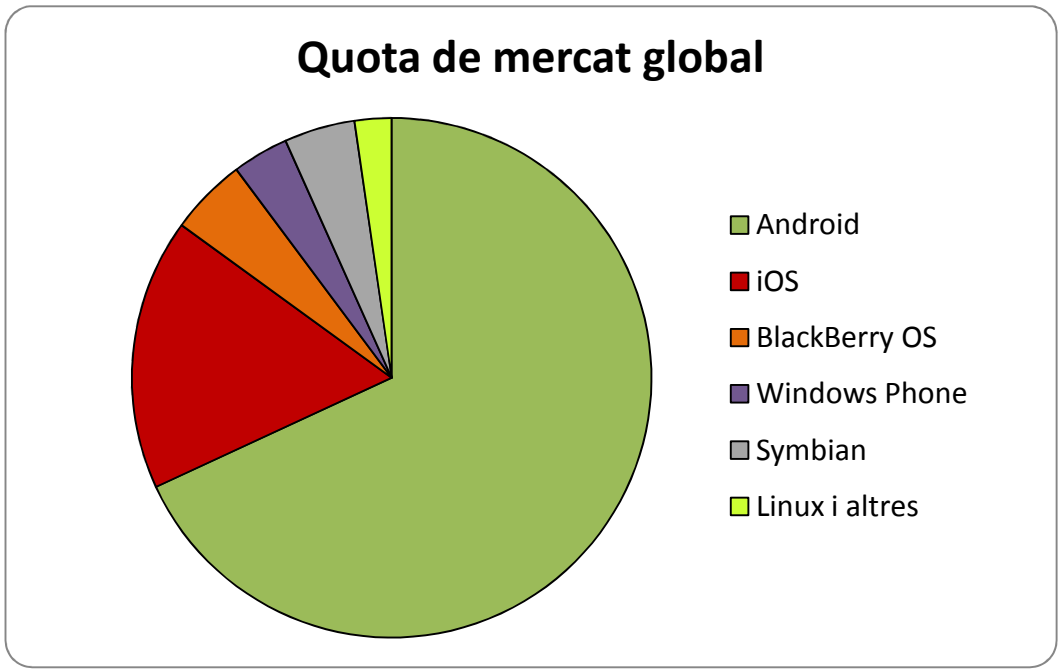
L'anunci es va realitzar el 5 de novembre de 2007 juntament amb la creació de la *Open Handset*, un consorci de 78 companyies de hardware, software i telecomunicacions dedicades al desenvolupament d'estàndards oberts per dispositius mòbils. *Google* va alliberar la majoria del codi font *d'Android* sota la llicencia *d'Apache*, una llicencia lliure i de codi obert.

El gran potencial d'aquest sistema es que té una gran comunitat de desenvolupadors que creen aplicacions per crear i estendre noves funcionalitats. Aquests aplicacions es poden descarregar des del portal oficial de *Google* –**Android Market** - o bé des de un portal creat per una altre empresa, com per exemple la *App Store d'Amazon* o el portal de *Samsung*, *Samsung Apps*.

Els programes estan escrits en llenguatge Java, avui dia un dels llenguatges de programació més instaurats i amb més seguidors i recolzament de la comunitat.

Ingressos Totals de *Google* Mobile (en bilions de dòlars)

Evolució de la implantació terminals Android *



*Quota de mercat a mitjans de 2012

Dades concretes: **Android** 68,1%, **iOS** 16,9%, **BlackBerry** OS 4,8%, **Symbian** OS 4,4%, **Windows Phone i Windows Mobile** 3,5 %, **Linux i altres** 2,3%

Android te la major quota, des de gener de 2011, amb més de la meitat del mercat, ha experimentat un gran creixement i en tan sols dos anys (del 2009 a començaments del 2011) ha passat a ser el SO mòbil més utilitzat.

INTRODUCCIÓ A LA TECNOLOGIA

Estructura del Sistema Operatiu (SO)

L'estructura del S.O. es compon d'aplicacions que s'executen en un ***framework*** Java d'aplicacions orientades a objectes sobre el nucli de les biblioteques de Java es una màquina virtual ***Dalvik*** amb compilació en temps d'execució. Les biblioteques escrites en el llenguatge C inclouen un administrador d'interfície gràfica (*surface manager*), un *framework* ***OpenCore***, una base de dades relacional *SQLite*, una interfície de programació *d'API* gràfica *OpenGL* ES 2.0 3D, un motor de renderitzat ***Webkit***, un motor gràfic *SGL*, SSL i una biblioteca estàndard de C *Bionic*.

JUSTIFICACIÓ DE L'ELECCIÓ

A mesura que els telèfons mòbils creixen en popularitat, els sistemes operatius amb els que funcionen adquireixen més importància. Per tant, escollir per quin decantar-se es una elecció que s'ha de fer amb cura i investigant totes les possibilitat i implicacions. En aquests moments hi ha 4 opcions a tenir en compte per volum d'usuaris: Android, *iOS*, *WindowsPhone* i *Blackberry OS*.

D'aquestes quatre opcions podem descartar el SO de *Blackberry* com a candidat, ja que no ofereix una plataforma prou sòlida i avançada per poder desenvolupar una aplicació d'aquestes característiques. El seu *framework* no proveeix eines per poder gestionar amb facilitat l'us de *Google Maps* i la ubicació de punts i rutes, per tant, com que això es la pedra angular de l'aplicació, no té sentit desenvolupar-la amb limitacions. Per altra banda, l'us de *Windows Phone* no està gaire implantat encara, semblea que amb la nova versió (*Mango*) i la unió amb Nokia podria donar una empenta i començar a expandir-se, però de moment es un sistema minoritari. Això també comporta beneficis, ja que hi ha poques aplicacions d'aquest tipus per aquesta plataforma, però la obligació de comprar una llicència del seu entorn de desenvolupament per poder desenvolupar en el seu llenguatge propi i el fet que no tingui cap altre benefici substanciós, fa que aquesta opció també sigui descartada. Anem doncs a l'últim competidor, *iOS d'Apple*. En aquest cas, el sistema es troba molt més implantat que la resta d'opcions i té una comunitat de seguidors fidels. A més els usuaris estan més predisposats a pagar per una aplicació que en la resta, però com que la nostra aplicació neix amb la idea de ser gratuïta, això no es un factor a tenir en compte. Per altre banda, es desenvolupa en *Objetive-C* (un llenguatge propi i corporatiu orientat a objectes) però que malgrat això no s'assembla a cap altre llenguatge. Per tant apareix un nou *handycap*, aprendre un nou llenguatge a part d'una nova forma d'estructurar el codi. També afegir que realment no es pot desenvolupar aplicacions sense pagar una llicència i sense tenir un *Mac*, es a dir, si es vol desenvolupar una aplicació per *iOS* s'ha de fer en un *Mac*, ja que es *l'unica* manera de que es pugui validar una aplicació amb el seu entorn propi de desenvolupament *XCode*. A més a més, aquest aplicació no es podrà instal·lar en un dispositiu si no es paga una llicència, amb la qual cosa ni tal sols es pot traspassar del Mac al telèfon propi de l'usuari per fer-la servir. Per totes aquestes raons i també per que no estic d'acord amb la política elitista i hipòcrita *d'Apple*, *iOS* no es una opció.

Android en canvi, ofereix totes les característiques que busquem, te absoluta interacció amb el seu servei *Google Maps*, el seu llenguatge està basat en Java, es gratuït i es por desenvolupar aplicacions de forma il·limitada i amb una llicència gratuïta. Per tot això s'ha escollit Android com a entorn de desenvolupament.

3.2. BASE DE DADES SQLITE

INTRODUCCIÓ A LA TECNOLOGIA

SQLite és una base de dades *Open Source* compatible amb *ACID* continguda en una petita biblioteca escrita en C. Té una sèrie de diferencies respecte als sistemes de gestió de base de dades client-servidor tradicionals que el fa especialment adient per dispositius mòbils. Primer de tot, es un procés separat, es a dir, no té un servidor de base de dades executant-se per que no es un procés independent amb el que el programa es comunica. En comptes d'això, *SQLite* s'enllaça amb el programa convertint-se en una part integral del programa. El programa utilitza la funcionalitat mitjançant crides simples a subrutines i funcions. Això redueix la latència en l'accés a la base de dades (degut a que les crides a funcions són més eficients que la comunicació entre processos). EL conjunt de la base de dades (definicions, taules, índexs y els registres amb les dades) es guarda en un únic fitxer estàndard en el telèfon. Aquest disseny simple s'aconsegueix bloquejant tot el fitxer de base de dades a l'inici de cada transacció. A més, degut a que es el sistema escollit per Android con a suport oficial, és relativament fàcil d'utilitzar gràcies al a implementació nativa y les funcions que inclou Android per accedir a les dades.

JUSTIFICACIÓ DE L'ELECCIÓ

Android proporciona 3 formes diferents de gestionar les dades: mitjançant la classe '*SharedPreferences*', mitjançant arxius de text i en una base de dades *SQLite*.

La classe '*SharedPreferences*' serveix per emmagatzemar petites configuracions o dades úniques. Aquesta informació és accessible des de qualsevol activitat de l'aplicació i continua emmagatzemada malgrat que es surti o es tanqui l'aplicació i fins i tot, en cas que es reinicïi el telèfon. En el nostre cas, aquesta eina es insuficient, ja que està pensada per informa dades simples del tipus ("nom","valor"), per tant no seria possible emmagatzemar objectes tan complexos con un POI. Per altre banda seria pràcticament inviable guardar i extreure totes les relacions entre els diferents objectes que composen el model de l'aplicació.

La opció d'utilitzar fitxers de text com a suport per mantenir les dades malgrat que pot ser més fàcil a primera vista d'implementar, genera problemes posteriors que no tenen una fàcil resolució. El cost de serialitzar/deserialitzar els objectes no es menor que el de llegir-los d'una base de dades, tot al contrari, si l'objecte es molt complex, pot perjudicar el rendiment. Per altre banda, buscar un registre en concret comporta la lectura de tots els registres i transformar-los per poder extreure després un en concret, per tant, les cerques complexes tampoc es veuen millorades amb aquest sistema per persistir les dades. Per tot això i per experiència pròpia, no soc partidari de fer servir els fitxers de text com a base de dades per que no crec que sigui un sistema eficaç i segur.

Veiem doncs, que la solució al problema de com es desarà la informació és bastant fàcil de resoldre. La opció del '*SharedPreferences*', malgrat que es una forma ràpida per accedir al a informació i no s'ha de desenvolupar cap mòdul específic per tractar les dades, es queda curta en tant a la capacitat d'emmagatzematge com els objectes complexos que formen el nucli de la capa del model.

Podem concloure doncs, que l'opció més adient es utilitzar un sistema més complex i estable, com es una base de dades relacional, per poder reflectir la complexitat de les classes del programa requereix.

Per altre banda, l'elecció de quina base de dades hem d'escollir per administrar la informació no està al es nostres mans, ja que ve donada per el propi Android. Per tant persistirem les dades amb una base de dades *SQLite*

.

4. ANÀLISI

4.1. INTRODUCCIÓ

En aquest apartat il·lustrarem les parts interessades en el projecte. Farem una mirada ràpida als recursos que intervenen a nivell de desenvolupament de l'aplicació, des de la part mes 'alta' de la cadena de creació del projecte, com es el cas del cap de projecte fins al tècnic de proves, que es l'última part que afecta al desenvolupament de l'aplicació. Tot seguit, analitzarem i definirem els possibles casos d'us que es poden generar per cada acció de l'usuari.

4.2. PARTS INTERESSADES

EQUIP DE DESENVOLUPAMENT DEL PROJECTE

Són les persones responsables del desenvolupament de l'aplicació. Engloben totes les fases i contemplen des de la presa de requisits del client i elaboració dels documents funcionals, passant per tota la fase de programació i direcció de l'equip fins la creació del joc de proves i 'testeig' de l'aplicació per comprovar la seva integritat abans d'entregar el producte final.

	Responsabilitat
Cap del projecte (CP)	Defineix, gestiona, planifica i controla el projecte.
Analista (A)	Col·labora amb el cap de projectes en l'estudi de viabilitat i la planificació. Analitza l'aplicació: arquitectura, metodologia, especificacions, estàndards,... Participa en el disseny i validació.
Programador (P)	Dissenya y desenvolupa l'aplicació d'acord amb l'anàlisi i planificació prevista. Participa en el procés de validació i implantació.
Tècnic de proves (TP)	Participa en el disseny de les proves internes i externes. Realitza les proves i participa en el procés de control de qualitat.
Director del projecte (DP)	Supervisa la feina de l'alumne, en alguns casos també pot actuar com *stakeholder*.

PERFILS D'USUARIS

Els usuaris que intervenen en l'aplicació a nivell d'usabilitat amb les seves funcions i limitacions en vers aquesta.

Nota: En aquest aplicació no hi ha distinció d'usuaris, ja que l'usuari final es el que tindrà el control total sobre el programa.

	Responsabilitat
Usuari Expert	Gestió i control del sistema, gestió dels manteniments, recollida i consulta d'informació.

STAKEHOLDER

Els '*stakeholders*' es podrien definir com les parts interessades del projecte. En el nostre cas, tant el client, com el cap de projecte, l'analista i resta de recursos/usuaris s'engloben en l'usuari expert. Excepte el cas del director de projecte, que es la figura del tutor que avalua i fa el seguiment del projecte i no forma part de la figura de l'usuari.

	Responsabilitat
Usuari Expert	Participa en la definició de requisits, subministrament d'informació, representa a l'usuari tipus. Participa en la validació del projecte.
Director del projecte (DP)	Supervisa la feina de l'alumne i avalua el projecte.

4.3. CASOS D'US DE L'APLICACIÓ

Com a part de l'estudi del projecte s'ha fet una avaluació de cada funcionalitat i s'ha reflectit en un cas d'ús específic.

Com a primera definició, podríem dir que un cas d'ús és una descripció dels passos o activitats que han de realitzar-se per portar a terme un procés. Els personatges o entitats que hi participen s'anomenen 'actors'. Més en profunditat, el diagrama serveix per aclarir quina és la seqüència interacció entre els actors i un sistema (aplicació) en resposta a un esdeveniment que inicia un actor principal sobre el propi sistema. Amb això podem especificar quina es la comunicació o el comportament de l'aplicació amb la interacció dels usuaris (actors) i altres sistemes.

En el cas que ocupa aquest document es veurà que la relació d'actors es trivial, ja que com s'ha comentat en l'apartat anterior, aquest aplicació només consta d'un actor (l'usuari final de l'aplicació). Això es déu a que en les aplicacions per dispositius mòbils no té sentit parlar de rols. No existeixen diferencies de permisos, ja que només portarien confusió a l'usuari, incomoditat a l'hora d'haver d'iniciar una sessió a l'aplicació segons l'acció que volgués dur a terme i es perdria rapidesa i immediatesa a l'hora de treballar amb el programa. Per tot això diem que l'usuari d'una aplicació destinada als 'smartphones' es un usuari final i únic que adopta el rol de tots els perfils possibles i ha de poder tenir control total sobre tots els aspectes de l'aplicació. Per tant els perfils s'han simplificat en un només anomenat usuari general.

A continuació es detalla un diagrama del casos d'ús del projecte amb la relació entre actor i acció, juntament amb una petita introducció al significat de cada camp del document.

DEFINICIÓ DELS CAMPS DEL CAS D'ÚS

Especificació de cas d'ús

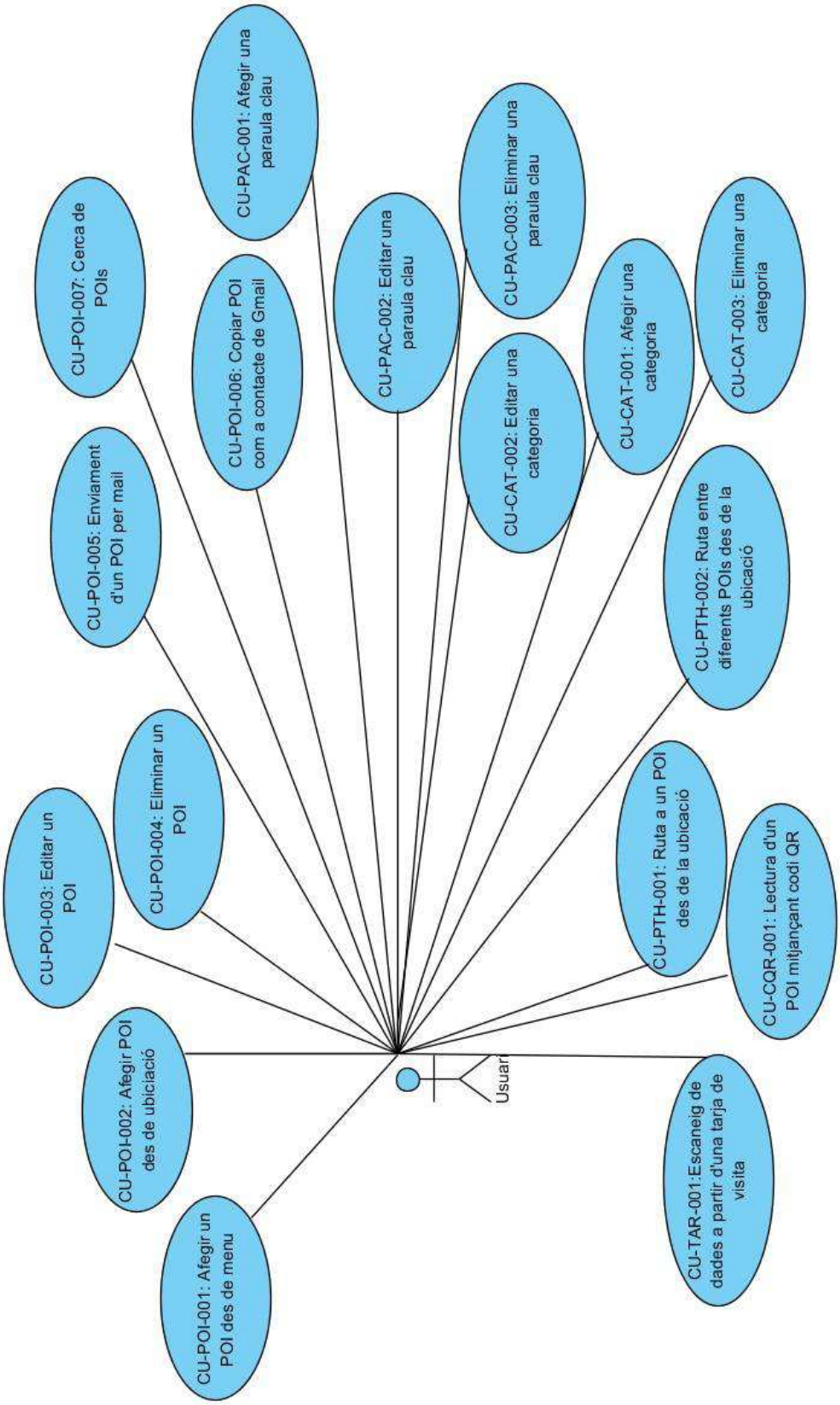
- **Identificador**: Codi d'identificació del cas d'ús. Es composa per les sigles: Cas Us – Descriptiu de 3 lletres del grup al que pertany – auto numèric del grup.
- **Nom del cas d'ús:** Descripció del cas d'ús.
- **Diagrama:** Gràfic que mostra les accions i els actors que hi intervenen.
- **Versió i data:** Versió del cas d'ús i la data de creació.

Aspectes funcionals

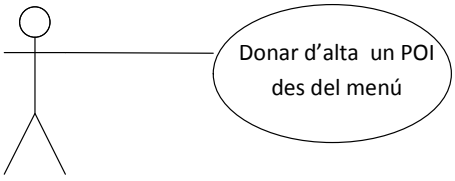
- **Resum de la funcionalitat:** Breu explicació del procés que es durà a terme.
- **Funcions primàries:** Llista dels passos que executarà l'usuari.
- **Relació d'actors:** Llistat d'actors que intervenen.

Aspectes formals

- **Casos d'ús relacionats:** Casos d'ús que tenen connexió en el cicle d'execució.
- **Precondicions:** Condicions prèvies que s'han de complir per tal que el cas es pugui dur a terme satisfactòriament.
- **Postcondicions:** Condicions que succeiran en el cas que el flux d'execució s'hagi completat.
- **Etapes:** Descripció en detall dels passos que ha de seguir l'usuari.
- **Alternatives de procés:** Processos paral·lels que poden interrompre el recorregut natural del projecte i afecten al resultat final del cas.
- **Excepcions:** Missatges d'error que es poden donar si l'execució del cas d'ús falla. Format: sigles EXcepció – grup del cas d'ús – auto numèric del cas d'ús.
- **Necessitats no funcionals:** Són requisits que especifiquen la necessitat de satisfer requisits legals, normatius, restriccions de disseny de l'entorn o problemes de compatibilitat. És a dir, qualsevol requisit que no permeti més d'una opció de disseny s'ha de considerar con a necessitat no funcional.


CU-PAC-001: Afegir una paraula clau
CU-PAC-002: Editar una paraula clau
CU-PAC-003: Eliminar una paraula clau
CU-POI-007: Cerca de POIs
CU-POI-006: Copiar POI com a contacte de Gmail
CU-CAT-001: Afegir una categoria
CU-CAT-002: Editar una categoria
CU-CAT-003: Eliminar una categoria
CU-POI-005: Enviament d'un POI per mail
CU-PTH-002: Ruta entre diferents POIs des de la ubicació
CU-POI-004: Eliminar un POI
CU-POI-003: Editar un POI
CU-PTH-001: Ruta a un POI des de la ubicació
CU-CQR-001: Lectura d'un POI mitjançant codi QR
CU-POI-002: Afegir POI des de ubiciació
CU-TAR-001:Escaneig de dades a partir d'una tarja de visita
CU-POI-001: Afegir un POI des de menu
Usuari

4.3.1. CU-POI-001: DONAR D'ALTA UN POI DES DEL MENÚ PRINCIPAL

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-POI-001	**Donar d'alta un POI des del menú principal**
Diagrama	
	
Versió	**Data**
1	11/06/2012
Aspectes funcionals	
Resum de la funcionalitat	
L'usuari crea un nou POI a través del botó 'Nou POI' del menú principal.	
Funcions primàries	
1. Crea un POI	
Relació d'actors	
Usuari	
Aspectes formals	
Casos d'ús relacionats	
CU-POI-002. Donar d'alta un POI des d'una ubicació. CU-POI-003. Editar un POI. CU-POI-004. Elimina el POI. CU-CAT-001: Afegir una categoria. CU-PAC-001: Afegir una paraula clau.	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i el botó per crear 'nou POI'.
Postcondicions	Un cop ha creat el POI l'usuari veu la pantalla amb el detall modificat del POI.
Etapes	
Informa les dades del POI 1. L'usuari selecciona el botó 'Nou POI'	

2. Apareix una finestra amb les dades i estructures a omplir:
    a. ***Nom***: Nom del POI a elecció de l'usuari
    b. ***Descripció***: Breu descripció del lloc
    c. ***Adreça***: Direcció física del POI
    d. ***Ubicació***: Coordenades pel geoposicionament (captades automàticament).
    e. ***Imatge***: Fotografia del POI
    f. ***Paraules*** clau: Una o vàries paraules clau per definir/filtrar el POI
    g. ***Categories***: Una o vàries categories clau per definir/filtrar el POI
    h. ***Puntuació***: Escala d'un a cinc de l'interés del POI.

Alternatives de procés

- Si l'usuari vol classificar el POI en una categoria que no existeix:
    - L'usuari obre la selecció de categoria.
    - Selecciona el botó 'Nova categoria'.
    - S'executa el procediment del cas d'us CU-CAT-001
    - L'usuari torna a la pantalla de creació del POI i continua l'execució normal.
- Si l'usuari vol escollir una paraula clau que no existeix:
    - L'usuari obre la selecció de categoria.
    - Selecciona el botó 'Nova paraula clau'.
    - S'executa el procediment del cas d'us CU-PAC-001
    - L'usuari torna a la pantalla de creació del POI i continua l'execució normal.

Excepcions

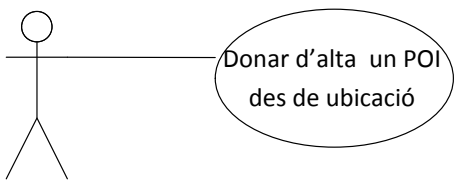
EX_POI_001: Error al inserir el POI

Necessitats no funcionals

- Usabilitat:
- Rendiment:
- Fiabilitat:

4.3.2. CU-POI-002: DONAR D'ALTA UN POI DES D'UNA UBICACIÓ

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
UC-POI-002	**Donar d'alta un POI des d'una ubicació**
Diagrama	



Versió	**Data**

1	11/06/2012

Aspectes funcionals

Resum de la funcionalitat

L'usuari crea un nou POI a través del botó per donar d'alta un POI semi automàticament a través de l' ubicació actual.

Funcions primàries

Informa dades del POI

Relació d'actors

Usuari

Aspectes formals

Casos d'ús relacionats

CU-POI-001. Donar d'alta un POI des del menú.

CU-POI-003. Editar un POI.

CU-POI-004. Elimina el POI.

CU-CAT-001: Afegir una categoria.

CU-PAC-001: Afegir una paraula clau.

Precondicions	L'usuari tècnic ha iniciat sessió a l'aplicació. Veu la llista de POIs i selecciona un per editar.
Postcondicions	Un cop feta qualsevol modificació sobre el POI, l'usuari retorna a la llista de POIs

Etapes

Informa les dades del POI

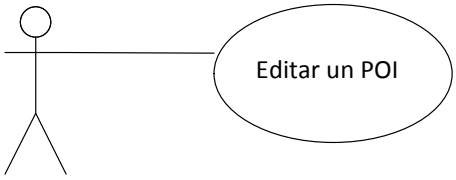
3. L'usuari selecciona el botó 'Nou POI'
4. Apareix una finestra amb les dades i estructures a omplir:
    a. ***Nom***: Nom del POI a elecció de l'usuari
    b. ***Descripció***: Breu descripció del lloc
    c. ***Adreça***: Direcció física del POI
    d. ***Ubicació***: Coordenades pel geoposicionament (captades automàticament).
    e. ***Imatge***: Fotografia del POI
    f. ***Paraules*** clau: Una o vàries paraules clau per definir/filtrar el POI
    g. ***Categories***: Una o vàries categories clau per definir/filtrar el POI
    h. ***Puntuació***: Escala d'un a cinc de l'interés del POI.

Alternatives de procés

- Si l'usuari vol classificar el POI en una categoria que no existeix:
    - L'usuari obre la selecció de categoria.
    - Selecciona el botó 'Nova categoria'.
    - S'executa el procediment del cas d'us CU-CAT-001
    - L'usuari torna a la pantalla de creació del POI i continua l'execució normal.

• Si l'usuari vol escollir una paraula clau que no existeix: o L'usuari obre la selecció de categoria. o Selecciona el botó 'Nova paraula clau'. o S'executa el procediment del cas d'us CU-PAC-001 o L'usuari torna a la pantalla de creació del POI i continua l'execució normal.
Excepcions
EX_POI_001: Error al inserir el POI
Necessitats no funcionals
No aplica.

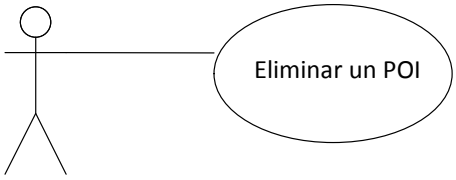
4.3.3. CU-POI-003: EDITAR UN POI

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
UC-POI-003	**Editar un POI**
Diagrama	
	
Versió	**Data**
1	11/06/2012
Aspectes funcionals	
Resum de la funcionalitat	
L'usuari informa sobre algunes dades de configuració del POI	
Funcions primàries	
Informa dades del POI	
Relació d'actors	
Usuari	
Aspectes formals	
Casos d'ús relacionats	
CU-POI-001. Donar d'alta un POI des del menú.	

CU-POI-002. Donar d'alta un POI des d'una ubicació. CU-POI-004. Elimina el POI. CU-CAT-001: Afegir una categoria. CU-PAC-001: Afegir una paraula clau.	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la llista de POIs i selecciona un per editar.
Postcondicions	Un cop feta qualsevol modificació sobre el POI, l'usuari retorna a la llista de POIs
Etapes	
1. L'usuari veu la llista de POIs. La llista conté el nom del POI. 2. Selecciona el POI i apareix una finestra amb les dades a editar: a. ***Nom***: Nom del POI a elecció de l'usuari b. ***Descripció***: Breu descripció del lloc c. ***Adreça***: Direcció física del POI d. ***Ubicació***: Coordenades pel geoposicionament (captades automàticament). e. ***Imatge***: Fotografia del POI f. ***Paraules*** clau: Una o vàries paraules clau per definir/filtrar el POI g. ***Categories***: Una o vàries categories clau per definir/filtrar el POI h. ***Puntuació***: Escala d'un a cinc de l'interés del POI. 3. Un cop finalitzats el canvis, l'usuari prem el botó 'Guardar' i desa els canvis.	
Alternatives de procés	
• Si l'usuari vol classificar el POI en una categoria que no existeix: o L'usuari obre la selecció de categoria. o Selecciona el botó 'Nova categoria'. o S'executa el procediment del cas d'us CU-CAT-001 o L'usuari torna a la pantalla de creació del POI i continua l'execució normal. • Si l'usuari vol escollir una paraula clau que no existeix: o L'usuari obre la selecció de categoria. o Selecciona el botó 'Nova paraula clau'. o S'executa el procediment del cas d'us CU-PAC-001 o L'usuari torna a la pantalla de creació del POI i continua l'execució normal.	
Excepcions	
EX_POI_002: Error al modificar el POI	
Necessitats no funcionals	
No aplica.	

4.3.4. CU-POI-004: ELIMINAR UN POI

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-POI-004	**Eliminar un POI**
Diagrama	



Versió	Data
1	11/06/2012

Aspectes funcionals

Resum de la funcionalitat

L'usuari elimina un POI mitjançant la selecció d'aquest en la llista de POIs

Funcions primàries

1. Elimina un POI

Relació d'actors

Usuari

Aspectes formals

Casos d'ús relacionats

CU-POI-001. Donar d'alta un POI des del menú.

CU-POI-002. Donar d'alta un POI des d'una ubicació.

CU-POI-003. Editar un POI.

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la llista de POIs emmagatzemats. Selecciona un d'ells i escull l'opció 'Eliminar'
Postcondicions	Un cop ha eliminat el POI, l'usuari veu la pantalla amb la llista de la resta POIs

Etapes

Informa dades del pla

1. L'usuari selecciona el botó 'Eliminar'
2. Apareix una finestra de confirmació per eliminar el POI

Alternatives de procés

No aplica

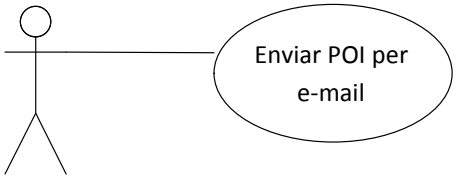
Excepcions

EX_POI_003: Error al eliminar el POI.

Necessitats no funcionals

No aplica.

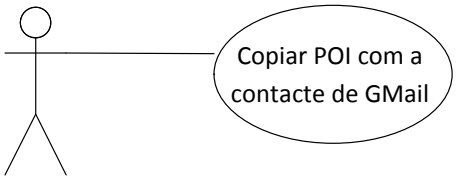
4.3.5. CU-POI-005: ENVIAMENT D'UN POI PER E-MAIL

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-POI-005	**Enviament d'un POI per e-mail**
Diagrama	
	
Versió	**Data**
1	11/06/2012
Aspectes funcionals	
Resum de la funcionalitat	
L'usuari escull un POI de la llista i mitjançant el menú contextual, l'envia per e-mail.	
Funcions primàries	
1. Envia un POI per e-mail	
Relació d'actors	
Usuari	
Aspectes formals	
Casos d'ús relacionats	
No hi ha.	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la llista de POIs emmagatzemats. Selecciona un d'ells i escull l'opció 'Enviar POI per e-mail'.
Postcondicions	Un cop s'ha enviat el POI, l'usuari torna a la pantalla amb la llista de POIs.
Etapes	
Informa dades del pla 1. L'usuari selecciona el botó 'Enviar POI per e-mail' 2. Apareix una finestra amb un camp de text per establir la direcció d'enviament. 3. Es mostra un missatge de confirmació al finalitzar l'enviament del POI.	
Alternatives de procés	
No aplica	
Excepcions	

EX_POI_003: Error al enviar el mail al remitent seleccionat
Necessitats no funcionals
No aplica.

4.3.6. CU-POI-006: COPIAR POI COM A CONTACTE EN GMAIL

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-POI-006	**Copiar POI com a contacte en Gmail**
Diagrama	



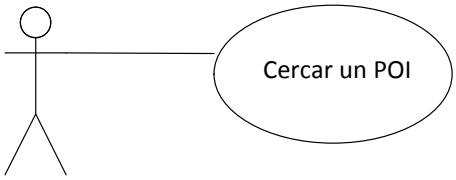
Versió	**Data**
1	11/06/2012

Aspectes funcionals
Resum de la funcionalitat
L'usuari escull un POI de la llista i mitjançant el menú contextual, el copia a la llibreta de contactes de Gmail.
Funcions primàries
1. Copiar un POI a la llibreta de contactes de Gmail
Relació d'actors
Usuari

Aspectes formals	
Casos d'ús relacionats	
No hi ha.	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la llista de POIs emmagatzemats. Selecciona un d'ells i escull l'opció 'Copiar POI com a contacte'.
Postcondicions	Un cop s'ha enviat el POI, l'usuari torna a la pantalla amb la llista de POIs.
Etapes	

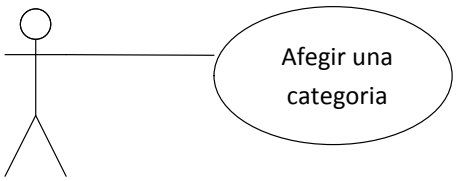
Informa dades del pla 1. L'usuari selecciona el botó 'Copiar POI a Gmail'. 2. Es mostra un missatge de confirmació al finalitzar el procés.
Alternatives de procés
No aplica
Excepcions
EX_POI_004: Error al crear el contacte
Necessitats no funcionals
No aplica.

4.3.7. CU-POI-007: CERCA DE POIS

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-POI-007	**Cerca de POIs**
Diagrama	
	
Versió	**Data**
1	11/06/2012
Aspectes funcionals	
Resum de la funcionalitat	
L'usuari escull un POI de la llista i mitjançant el menú contextual, el copia a la llibreta de contactes de Gmail.	
Funcions primàries	
1. Cercar un POI que compleixi les característiques aplicades la filtre.	
Relació d'actors	
Usuari	
Aspectes formals	
Casos d'ús relacionats	

No hi ha.	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la cerca de POIs. Selecciona els criteris de cerca i accepta.
Postcondicions	Es mostra una llista amb els POIs que s'han retornat amb la cerca establerta.
Etapes	
Informa dades del pla 1. L'usuari selecciona el botó 'Cercar POIs'. 2. Els camps a emplenar son: a. Paraules clau b. Categories c. Nom d. Descripció e. Adreça 3. Es mostra un missatge de confirmació al finalitzar el procés.	
Alternatives de procés	
No aplica	
Excepcions	
No hi ha.	
Necessitats no funcionals	
No aplica.	

4.3.8. CU-CAT-001: AFEGIR UNA CATEGORIA

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-CAT-001	**Afegir una categoria**
Diagrama	
	
Versió	**Data**
1	11/06/2012
Aspectes funcionals	
Resum de la funcionalitat	

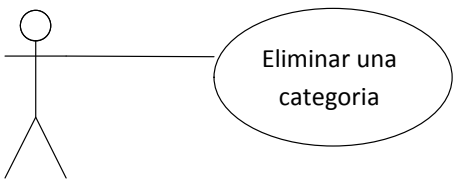
L'usuari crea un nova categoria mitjançant el boto 'Nova' de la pantalla Categories
Funcions primàries
1. Crear una categoria
Relació d'actors
Usuari
Aspectes formals
Casos d'ús relacionats
CU-CAT-002. Eliminar una categoria

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix les categories. Selecciona el botó 'Nova' i veu la pantalla de creació.
Postcondicions	Un cop s'ha creat la categoria, l'usuari torna a la pantalla amb la llista de categories.

Etapes
Informa dades del pla 1. L'usuari selecciona el botó 'Nova' 2. Apareix una finestra amb mes dades a omplir:
Alternatives de procés
No aplica
Excepcions
EX_CAT_001: Error a l'inserir la categoria.
Necessitats no funcionals
No aplica.

4.3.9. CU-CAT-002: ELIMINAR UNA CATEGORIA

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-CAT-002	**Eliminar una categoria**
Diagrama	



Versió	**Data**

1	11/06/2012

Aspectes funcionals

Resum de la funcionalitat

L'usuari elimina una categoria mitjançant la selecció d'aquesta en la llista de categories

Funcions primàries

1. Elimina una categoria

Relació d'actors

Usuari

Aspectes formals

Casos d'ús relacionats

CU-POI-001. Donar d'alta una categoria

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la llista de categories. Selecciona una d'ells i escull l'opció 'Eliminar'
Postcondicions	Un cop ha eliminat la categoria, l'usuari veu la pantalla amb la llista de la resta de categories

Etapes

Informa dades del pla

1. L'usuari selecciona el botó 'Eliminar'
2. Apareix una finestra de confirmació per eliminar la categoria

Alternatives de procés

Si l'usuari elimina una categoria amb categories filles, aquestes s'eliminaran en cascada juntament amb la categoria mare.

Excepcions

EX_CAT_002: Error al esborrar la categoria.

Necessitats no funcionals

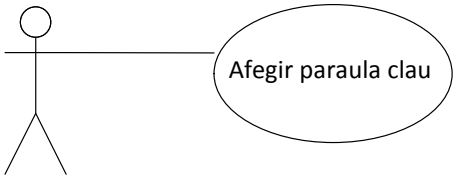
No aplica.

4.3.10. CU-PAC-001: AFEGIR UNA PARAULA CLAU

Especificació de cas d'ús

Identificador	**Nom del cas d'ús**
CU-PAC-001	**Afegir una paraula clau**

Diagrama



Versió	Data
1	11/06/2012

Aspectes funcionals

Resum de la funcionalitat

L'usuari afegeix una paraula clau mitjançant el boto 'Nova' de la pantalla 'Paraules clau'.

Funcions primàries

1. Elimina una paraula clau

Relació d'actors

Usuari

Aspectes formals

Casos d'ús relacionats

CU-POI-002. Eliminar una paraula clau

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a les paraules clau. Selecciona el botó 'Nova' i veu la pantalla de creació.
Postcondicions	Un cop s'ha creat la paraula clau, l'usuari torna a la pantalla amb la llista de categories on es veurà inserida la nova paraula clau.

Etapes

Informa dades del pla

1. L'usuari selecciona el botó 'Nova'
2. Apareix una finestra amb mes dades a omplir:

Alternatives de procés

No hi ha.

Excepcions

EX_PAC_001: Error al crear la paraula clau

Necessitats no funcionals

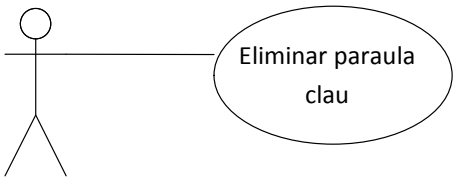
No aplica.

4.3.11. CU-PAC-002: ELIMINAR UNA PARAULA CLAU

Especificació de cas d'ús

Identificador	Nom del cas d'ús
CU-PAC-002	Eliminar una paraula clau

Diagrama



Versió	Data
1	11/06/2012

Aspectes funcionals

Resum de la funcionalitat

L'usuari elimina una paraula clau mitjançant la selecció d'aquesta en la llista de paraules clau

Funcions primàries

1. Elimina una paraula clau

Relació d'actors

Usuari

Aspectes formals

Casos d'ús relacionats

CU-PAC-001. Donar d'alta una paraula clau

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la llista de paraules clau. Selecciona una d'elles i escull l'opció 'Eliminar'
Postcondicions	Un cop ha eliminat la paraula clau, l'usuari veu la pantalla amb la llista de la resta de paraules clau

Etapes

Informa dades del pla

1. L'usuari selecciona el botó 'Eliminar'
2. Apareix una finestra de confirmació per eliminar la paraula clau

Alternatives de procés

No aplica.

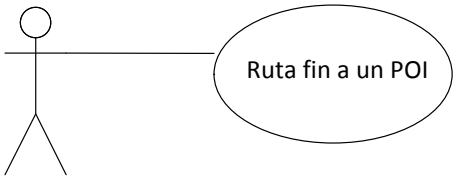
Excepcions

EX_PAC_002: Error al eliminar la paraula clau

Necessitats no funcionals

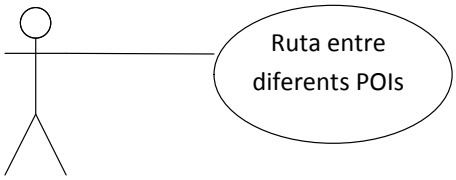
No aplica.

4.3.12. CU-PTH-001: RUTA A UN POI

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-PTH-001	**Ruta fins a un POI**
Diagrama	
	
Versió	**Data**
1	11/06/2012
Aspectes funcionals	
Resum de la funcionalitat	
L'usuari selecciona un POI des de la llista de POIs i escull l'opció al menú contextual "Traçar ruta".	
Funcions primàries	
1. Traçar la ruta des de l' ubicació actual de l'usuari fins al punt de destí, que en aquest cas es la direcció del POI seleccionat.	
Relació d'actors	
Usuari	
Aspectes formals	
Casos d'ús relacionats	
CU-PTH-002. Recorregut entre més d'un POI	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la llista de POIs. Obre el menú contextual i selecciona l'opció "Traçar ruta".
Postcondicions	Un cop s'ha seleccionat l'opció "Traçar ruta", l'aplicació obrirà un enllaç a *Google*Maps amb el POI com a punt de destí ja seleccionat i traçarà la ruta automàticament.
Etapes	
Informa dades del pla 1. L'usuari escull un POI i obre el menú contextual 2. Selecciona el botó "Traçar ruta" 3. S'obre una nova finestra de *Google*Maps amb un enllaç al POI ja especificat 4. L'usuari por iniciar la navegació a través de *Google*Maps	

Alternatives de procés
No hi ha.
Excepcions
EX_PTH_001: Error al obrir l'enllaç a *Google Maps* **EX_PTH_002: Error al posicionar tots els POIs**
Necessitats no funcionals
No aplica.

4.3.13. CU-PTH-002: RECORREGUT ENTRE MÉS D'UN POI

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-PTH-002	**Recorregut entre més d'un POI**
Diagrama	
	
Versió	**Data**
1	11/06/2012

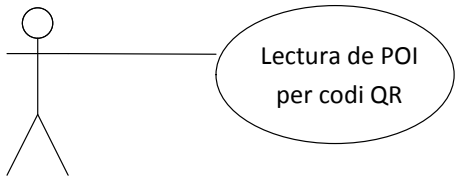
Aspectes funcionals
Resum de la funcionalitat
L'usuari selecciona el botó "ruta" de la finestra principal i escull una sèrie de POIs per traçar una ruta entre tots ells.
Funcions primàries
1. Escollir una sèrie de POIs de la llista de POIs de l'usuari. 2. Traçar la ruta des de l' actual de l'usuari fins al punt de destí, que en aquest cas es la direcció del POI seleccionat
Relació d'actors
Usuari

Aspectes formals
Casos d'ús relacionats
CU-PTH-001. Ruta a un POI

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la opció "ruta entre POIs". Obre una finestra per escollir els punts i accepta.
Postcondicions	Un cop s'ha acceptat, l'aplicació obrirà un enllaç a *Google*Maps amb tots els POIs informats a la pantalla i traçarà la ruta automàticament entre tots ells.
Etapes	
Informa dades del pla 1. L'usuari selecciona el botó "ruta entre POIs" 2. A la finestra resultant, escull una llista de POIs i accepta. 3. S'obre una nova finestra de *Google*Maps amb els POI escollits. 4. Es traça una ruta automàticament entre tots ells 5. L'usuari inicia la navegació a través de *Google*Maps.	
Alternatives de procés	
No hi ha.	
Excepcions	
EX_PTH_001: Error al obrir l'enllaç a *Google Maps* **EX_PTH_002: Error al establir tots els POIs**	
Necessitats no funcionals	
No aplica.	

4.3.14. CU-CQR-001: Lectura d'un POI mitjançant codi QR

Especificació de cas d'ús	
Identificador	**Nom del cas d'ús**
CU-CQR-001	**Lectura d'un POI mitjançant un codi QR**
Diagrama	



Versió	**Data**
1	11/06/2012

Aspectes funcionals
Resum de la funcionalitat
L'usuari selecciona el botó "llegir QR" de la finestra principal i enfoca amb la càmera un codi QR per extreure les dades d'un POI.
Funcions primàries

1. Obrir el lector de codis QR
2. Enfocar amb la càmera un codi QR i extreure les dades per desar-les a l'aplicació.

Relació d'actors
Usuari

Aspectes formals

Casos d'ús relacionats	
CU-POI-001: Afegir un POI	
Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la opció "escanejar codi QR".
Postcondicions	Quan el programa aconsegueix escanejar la informació de forma satisfactòria, l'usuari veu la finestra de edició d'un POI on pot confirmar els resultats.

Etapes

Informa dades del pla

1. L'usuari selecciona el botó "escanejar codi QR".
2. El programa inicia la càmera i intenta capturar el codi QR.
3. Si el codi s'ha capturat satisfactòriament, es mostra la pantalla d'edició del POI.
4. L'usuari confirma que les dades son correctes i desa el POI.

Alternatives de procés

No hi ha.

Excepcions

EX_CQR_001: Error al escanejar el codi QR

Necessitats no funcionals

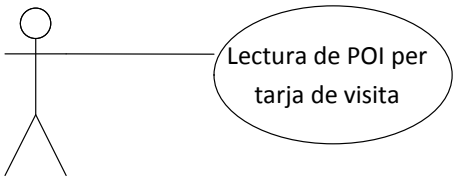
No aplica.

4.3.15. CU-TAR-001: Escaneig una Tarja de Visita per obtenir un POI

Especificació de cas d'ús

Identificador	**Nom del cas d'ús**
CU-TAR-001	**Lectura d'un POI mitjançant una targeta de visita**

Diagrama



Versió	**Data**

1	11/06/2012

Aspectes funcionals

Resum de la funcionalitat

L'usuari selecciona el botó "escanejar T.V." de la finestra principal i enfoca amb la càmera una targeta de visita comú

Funcions primàries

1. Obrir el lector de targes de visita
2. Enfocar amb la càmera la tarja i extreure les dades per desar-les a l'aplicació.

Relació d'actors

Usuari

Aspectes formals

Casos d'ús relacionats

CU-POI-001: Afegir un POI

Precondicions	L'usuari ha iniciat sessió a l'aplicació. Veu la pantalla principal i accedeix a la opció "escanejar T.V".
Postcondicions	Quan el programa aconsegueix escanejar la informació de forma satisfactòria, l'usuari veu la finestra de edició d'un POI on pot confirmar els resultats.

Etapes

Informa dades del pla

1. L'usuari selecciona el botó "escanejar T.V".
2. El programa inicia la càmera i intenta capturar les dades de la tarja.
3. Si les dades s'han capturat satisfactòriament, es mostra la pantalla d'edició del POI.
4. L'usuari confirma que les dades son correctes i desa el POI.

Alternatives de procés

No hi ha.

Excepcions

EX_TAR_001: Error al escanejar la targeta de visita

Necessitats no funcionals

No aplica.

5. DISSENY I IMPLEMENTACIÓ

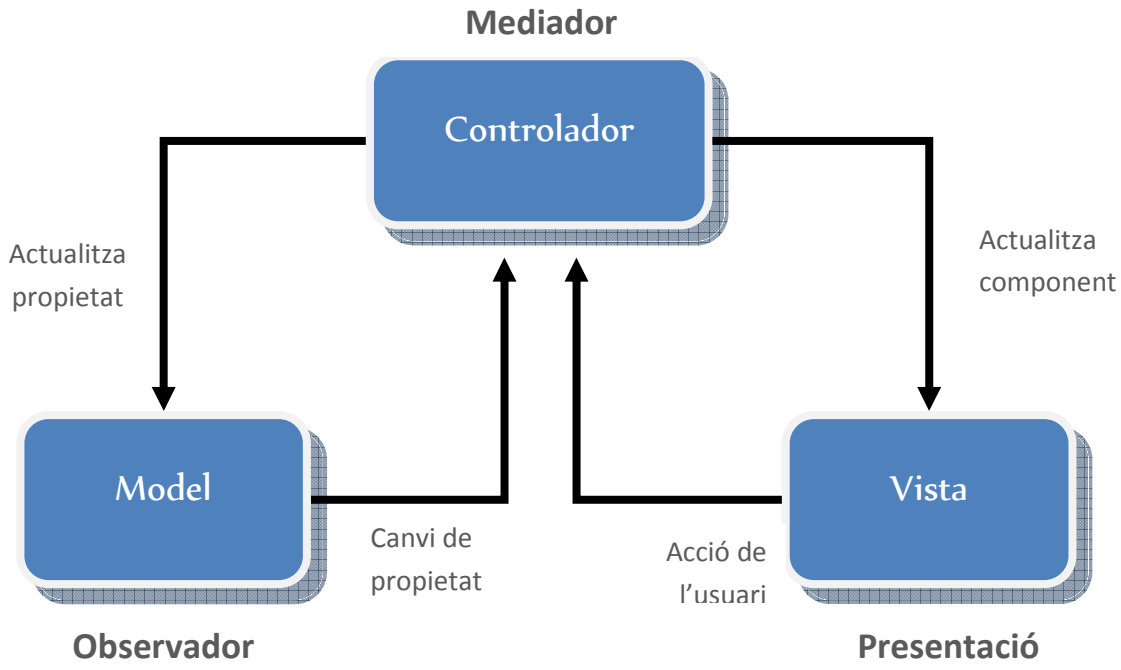
5.1. ESTRUCTURA INTERNA DE LA APLICACIÓ

L'arquitectura **Model–View–Controller** (MVC) és un patró de disseny per al desenvolupament de programari que separa el model de dades, la interfície d'usuari i la lògica de control. El patró MVC es veu freqüentment en aplicacions web, on es pot visualitzar una pàgina HTML i el codi que proveeix de dades dinàmiques a la pàgina, el controlador és el sistema de gestió de bases de dades i el model és el model de dades.

Model: El model representa les dades i les regles que controlen l'accés i l'actualització d'aquestes dades. Normalment son classes que modelen o son una aproximació a un procés o concepte del món real.

Vista: La vista renderitza el contingut d'un model. Especifica com s'ha de presentar les dades. Si el model canvia, la vista ha d'actualitzarse per representar els canvis en el model. Això hi ha dues formes d'aconseguir-ho: fent servir un model 'push', on la vista es registra en si mateixa al model per recollir les notificacions, o un model pull, on la vista es responsable de cridar el model quan necessita mostrar aquesta informació.

Controlador: El controlador tradueix les accions de l'usuari amb la vista en accions que el model executarà. Per exemple quan l' usuari interacciona amb un clic de botó o una selecció en un menú. Depenent del context, el controlador també pot seleccionar una nova vista per presentar la nova informació per presentar-la a l'usuari.



5.1.1. ESTRUCTURA DEL MODEL

El model es potser la part més trivial de l'aplicació. Només conté una representació objectual de les taules de la base de dades i la seva relació entre elles. Els camps estan modelitzats amb valors primitius (cadenes de text, números enters, decimals, etc.) mentre que les relacions entre les taules es fan declarant (en la classe que té la visibilitat de l'altre objecte) col·leccions o llistes de les classes que representen aquestes taules. Per exemple en el cas de la classe Poi, tenim els atributs 'simples' com por ser el nom, la descripció, puntuació, etc. I tenim objectes que fan referència a un registre d'altre taula (con la ubicació) i per últim tenim llistes d'objectes per reflectir la possibilitat que hi hagi més d'un registre associat al POI com el es cas de la llista d'horaris o les categories. En canvi, com que la navegabilitat només és en un sentit en el cas de la ubicació, l'objecte ubicació no serà capaç de referenciar l'objecte Poi al que pertany.

IMPLEMENTACIÓ

La conceptualització del diagrama de classes va ser el primer pas a l'hora de començar el projecte. El procés de modelitzar la idea inicial va ser un procés de depuració, en un primer moment vaig fer un esborrany de les classes principals, entre els quals destaca el POI que es el centre de l'aplicació i després vaig desenvolupar les idees de com distribuir la informació opcional que necessitaria per mostrar de manera fidel i amb la flexibilitat necessària per si s'havien d'ajustar després les dades.

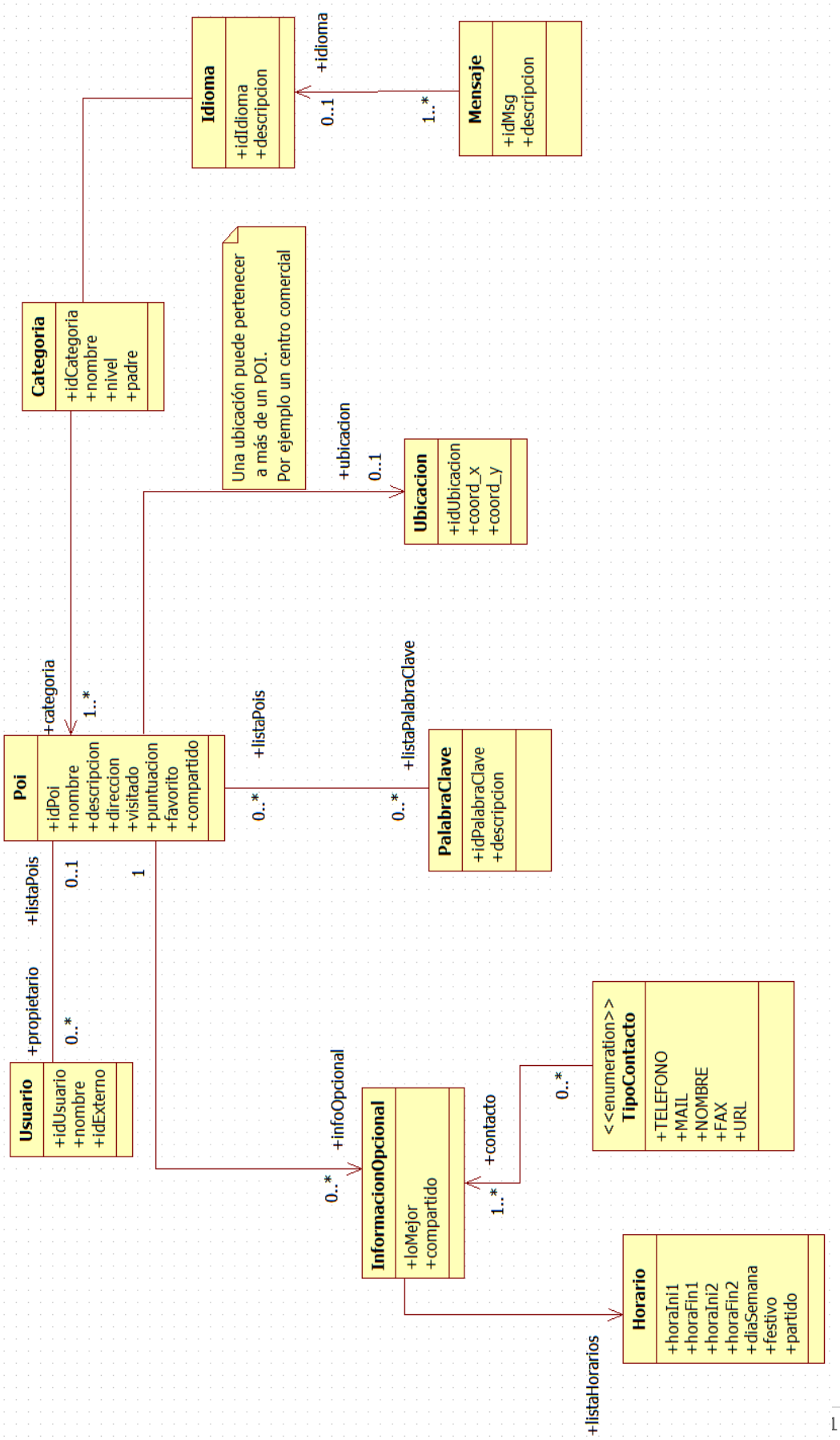
EL procés incloïa preguntes del tipus: Quants nivells ha de tenir la llista de categories? Es deixarà crear i esborrar nodes? O amb les paraules clau: Quan l'usuari entri una paraula clau es guardarà amb la resta de paraules clau o tindrà les seves pròpies? Que es un tipus d'informació opcional? Creem llistes predefinides amb el contacte, el telèfon, el fax, etc. O deixem que l'usuari introdueixi les que vulgui amb el seu criteri? A través d'aquestes preguntes es va anar formant les relacions finals entre les classes, es va decidir quines serien entitat i quines atributs. Tot això no obstant no vol dir que aquets diagrama fos vàlid per transformar-lo en taules, es bastant probable que no serveixin les mateixes relacions (com va ser al meu cas i es veurà més endavant) i que la base de dades i les classes no estiguin relacionades de la mateixa forma. Com a exemple posaré el cas de la classe informació opcional, en un primer moment es va decidir que un POI tindria una llista d'informació opcional, i que d'aquesta penjarien els horaris i els tipus de dades. Conceptualment es correcte, però després al portar-lo a la pràctica, es va refer a la base de dades i es va trencar aquesta associació. La classe Informació Opcional va desaparèixer per que feia més nosa que servei, i

finalment els tipus de dades i els horaris es van relacionar directament amb la taula dels POIs.

Exemple de codi d'una classe en JAVA

```
public class Poi {

	private long idPoi;
	private String nombre;
	private String descripcion;
	private String direccion;
	private boolean visitado;
	private float puntuacion;
	private boolean favorito;
	private boolean compartido;
	private Usuario propietario;
	private Ubicacion ubicacion;
	private List<Categoria> listaCategorias;
	private List<PalabraClave> listaPalabraClave;
	private List<PoiTipoDato> listaInfoOpcional;
	private List<Horario> listaHorarios;


	public Poi (long idPoi, String nombre, String descripcion, float puntuacion,
	  String direccion, boolean favorito, boolean visitado, boolean compartido) {

		this.idPoi = idPoi;
		this.nombre = nombre;
		this.descripcion = descripcion;
		this.direccion = direccion;
		this.visitado = visitado;
		this.puntuacion = puntuacion;
		this.favorito = favorito;
		this.compartido = compartido;

		this.propietario = null;
		this.ubicacion = null;
		this.listaCategorias = null;
		this.listaPalabraClave = null;
		this.listaInfoOpcional = null;
		this.listaHorarios = null;
	}
.
.
	public long getIdPoi() {
		return idPoi;
	}
	public void setIdPoi(long idPoi) {
		this.idPoi = idPoi;
	}
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public Ubicacion getUbicacion() {
		return ubicacion;
	}
	public void setUbicacion(Ubicacion ubicacion) {
		this.ubicacion = ubicacion;
	}
.
.
}
```

DIAGRAMA DE CLASSES



5.1.2. ESTRUCTURA DEL CONTROLADOR

La forma en la que el Controlador es comunica amb el model i les vistes es única en *Android*. A continuació definirem breument els controls i procediments que fa servir:

***Activity*:** Representa el component principal de la interfície gràfica d'una aplicació Android. Es por pensar en una *Activity* con l'element equivalent a una finestra en qualsevol altre llenguatge visual.

***Service*:** Son components sense interfície gràfica que s'executen en segon pla. En concepte, son exactament iguals als serveis presents en qualsevol altre sistema operatiu. Els serveis poden realitzar qualsevol tipus d'acció, per exemple, actualitzar dades, llençar notificacions, o mostrar elements visuals si es necessita la interacció de l'usuari en u moment donat.

***Content Provider*:** Es el mecanisme que s'ha definit a Android per compartir dades entre aplicacions. Mitjançant aquests components es possible compartir determinades dades de la nostra aplicació sense mostrar detalls sobre el seu emmagatzematge intern, la seva estructura, o implementació. De la mateixa forma, la nostra aplicació podrà accedir a les dades d'altre a través dels *content provider* que s'hagin definit.

***Intent*:** Un intent es l'element bàsic de comunicació entre els distints components Android que hem descrit anteriorment. Es pot entendre com els missatges o peticions que son enviats entre els distints components d'una aplicació o entre distintes aplicacions. Mitjançant un Intent es pot mostrar una *Activity* des de qualsevol altre, iniciar un servei, enviar un missatge de *broadcast*, iniciar altre aplicació, etc.

Ajuntant totes aquestes definicions tenim el següent cicle: Tots el fitxers de vistes *XML* (que veurem més endavant) tenen al darrere un arxiu *.java* que es el controlador associat a aquesta vista. Gràcies a això, quan declarem el controlador i li especifiquem quina es la seva vista associada, podem recollir els controls a través del seu identificador i interaccionar amb ells. A aquesta classe *java* se li anomena *Activity*, i pot rebre informació addicional a través dels Intents, que no son més que informació encapsulada que s'envien a les *Activities*. Un exemple seria el cas que, estant en la llista de POIs es volgués veure el detall d'un POI en concret. En aquest cas, *l'Activity llistaPoi* iniciaria *l'Activity detallPoi* i li enviaria un Intent amb la informació del POI que es vol mostrar.

IMPLEMENTACIÓ

En la nostra aplicació com es lògic, hem necessitat per força tots aquests objectes i conceptes per comunicar les diferents vistes. Tots el fitxers XML tenen darrere un *Activity* associat, i entre ells es criden a través dels *Intents*. Com a detall comentar que es obligatori declarar tots els controladors a l'arxiu de configuració general de l'aplicació (anomenat *AndroidManifest.xml*) i especificar una línia com aquesta:

```
<activity android:name="shopoing.controller.VerPoiMapa"></activity>
```

Un altre detall que em va retenir bastant temps es el fet que un Intent no pot enviar informació complexa (objectes) a un *Activity*, es a dir només pot enviar tipus de dades primitives com *strings*, *integers*, long, etc.. per tant tenim el *handycap* de la impossibilitat d'enviar classes pròpiament dit. Per fer això existeix en Java el procés de *serialització/deserialització*, que consisteix en enviar un objecte codificat i descodificar-lo a l'altra banda. Aquesta metodologia existeix també a *Android* però es infinitament més lenta, per això té la seva pròpia implementació anomenada *Parcelable*. Amb aquest sistema l'objecte que es vol enviar en un Intent ha d'heretar d'aquesta interfície i implementar una sèrie de mètodes que descomponguin l'objecte per enviar-lo i el tornin a reconstruir al instanciar-lo. Un exemple d'aquesta implantació el trobem a la classe *Horario.java*

Com a últim punt comentar que la localització i posicionament d'un POI a *Google Maps* no es un procés trivial. S'ha de crear una passarel·la de comunicació entre el *Content Provider* que gestiona la comunicació (*LocationManager*) per gestionar el *GPS*, *wi-fi* o qualsevol tipus de connexió que tingui el telèfon disponible en aquest moment. Registrar el nostre event en aquest *Manager*, informar de l'ordre de les preferències (si ens interessa més la rapidesa o la fiabilitat per exemple) i tot seguit informar en el manifest els permisos necessaris per habilitar la localització. Després de tot això, i si en última instancia l'usuari dona permís per que aquest servei s'activi, es podrà començar a rebre actualitzacions de la posició de l'usuari. No és un procés immediat, si no que depèn de la qualitat del senyal dels satèl·lits, les xarxes *wi-fi* que estiguin a l'abast i el temps que es deixi funcionant el posicionador. Per totes aquestes variables, la captura del punt actual on es troba l'usuari no es un procés que es cridi amb una funció de la llibreria pròpia *d'Android* tipus '*donamPosicioActual()*' i per tant no sempre es fiable o acurat.

Exemple de codi de la capa del controlador

```
public class EditPoi extends Activity {

        private Poi poi;
        private LocationManager locManager;
        private LocationListener locListener;

    //Al crear la vista
    @Override
    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.edit_poi);

        long idPoi = getIntent().getExtras().getLong("idPoi");

        DBAdapter_Poi DBAPoi = new DBAdapter_Poi(this);
        DBAPoi.open();
        poi = DBAPoi.getPoi(idPoi);
        DBAPoi.close();

    }

    public void btnSelCategoriaClick(View arg0) {

        Intent intent = new Intent(EditPoi.this, EditCategoria.class);

        //Creamos la información a pasar entre actividades
        Bundle b = new Bundle();
        b.putString("categoriasIDs", categoriasIDs);

        //Añadimos la información al intent
        intent.putExtras(b);

        startActivityForResult(intent, 1);
    }

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data) {

        super.onActivityResult(requestCode, resultCode, data);

        switch(requestCode) {

            case (1) : { // VENIMOS DE LA SELECCION DE CATEGORIAS

              if (resultCode == Activity.RESULT_OK) {
                  Button btnCat = (Button)findViewById(R.id.btnSelCategoria);
                  //Info del bundle que trae los parametros del activity que ha hecho la llamada
                  Bundle bundle = data.getExtras();

                  if (bundle.getString("categoriasNombre") != null) {
                    btnCat.setText(bundle.getString("categoriasNombre"));
                    categoriasIDs = bundle.getString("categoriasID");
                  }
                  else {
                    btnCat.setText(getString(R.string.txtSelCategoria));
                    categoriasIDs = "";
                  }
              }
              break;
            }
          }
        }
}
```

L'onCreate es crida a l'inici de l'activitat, recull la informació de l'intent i inicialitza les variables

Agafem la informació del Intent que ens ha enviat la Activity que ha inicat aquesta

Aquest mètode es crida quan es sleecciona un botó de la vista. En aquesta cas la selecció de categories

Creem el nou Intent, li asignem la informació a enviar i amb el mètode startActivityForResult, cridem a la nova Activity.

Quan hem escollit una categoria, aquest mètode es dispara i recull el resultat i les variables enviades.

Actualitzem els camps de la vista actual amb els valors recollits de l'Activity anterior

*Aquesta classe no es funcional, ja que el codi ha estat reduït per motius il·lustratius i d'espai.

5.1.3. ESTRUCTURA DE LA VISTA

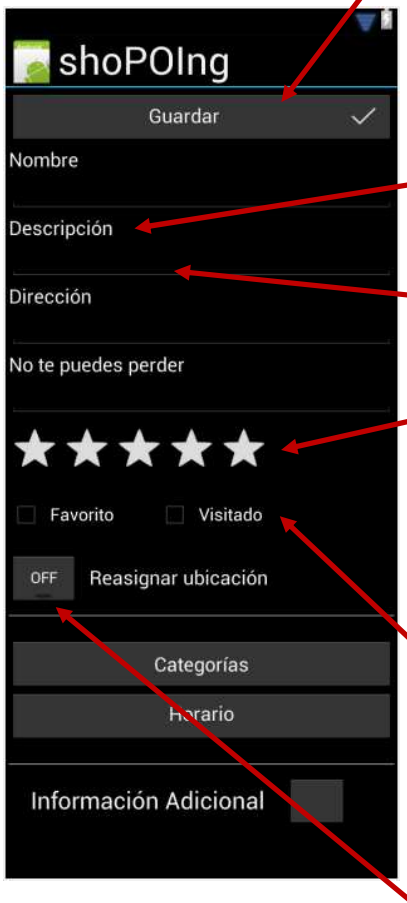
Android treballa amb arxius XML per presentar les pantalles a l'usuari. Aquests arxius consten d'una estructura especial, que s'agrupen en `Layouts`, que son elements no visuals destinats a controlar la distribució, posició i dimensió dels controls que s'insereixen en el seu interior. Aquests components estenen la classe base `ViewGroup` com molts altres components contenidors, es a dir, capaços de contenir altres controls. Aquests altres controls s'anomenen *View* i son els components bàsics amb els quals es construeix l'interfície gràfica de l'aplicació, de forma anàloga a com es faria a *.NET* o *Java*. D'inici les llibreries *d'Android* ja porten una sèrie de controls definits de sèrie (quadres de text, botons, llistes desplegables, etc.) però també existeix la possibilitat d'estendre la funcionalitat d'aquests controls bàsics per poder crear els nostres propis.

IMPLEMENTACIÓ

En el cas que ens ocupa, els controls que proporciona *Android* han sigut suficients en la gran majoria de casos. El major problema ha sigut com presentar la informació correctament en pantalla sense redundar en el contingut ni perdre cap funcionalitat, tenint en ment la reduïda dimensió d'aquesta. Aquests problemes venen bàsicament per haver dissenyar sempre aplicacions web o d'escriptori, mai per dispositius mòbils. Per exemple a mig desenvolupament em vaig adonar de la inutilitat de tenir un botó per tornar enrere a l'aplicació, ja que els telèfons mòbils ja porten un botó (normalment físic) que fa aquesta funció. Per tant el vaig treure per evitar confondre a l'usuari i vaig redissenyar les pantalles.

Un dels grans reptes va ser com mostrar la informació a l'hora escollir les categories del POI. El control havia de mostrar una vista en forma d'arbre, però Android no ofereix cap control d'aquest tipus i l'únic que es podria adaptar, l'`ExpandableListView`, nomes es capaç de mostrar dos nivells de profunditat. Per tant vaig haver de buscar un control que ho fes amb una llibreria externa, ja que desenvolupar un control tan complex m'hauria portat tot el temps que em quedava. Vaig trobar un projecte a *'Google Code'* anomenat `pl.polidea.treeview` i vaig adaptar-lo al meu projecte per tan que mostrés un arbre multi selector amb les categories. Com a concessió personal, diré que el dissenyador *d'Eclipse* per crear les vistes es bastant desagraït d'utilitzar, mentre que l'eina per dissenyar manualment es comporta de manera correcte, no es capaç de representar correctament el fitxer en el mode visual d'edició XML, amb la qual cosa es perd una quantitat ingent de temps intentant que el dissenyador mostri correctament el controls per poder dissenyar la vista.

Exemple d'un fragment del codi XML de la vista




```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/scrollLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="3dp" >


    <LinearLayout
        android:id="@+id/linLayoutNewPoi"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical" >

        <Button
            android:id="@+id/btnUpdate"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:drawableRight="@drawable/ic_aceptar"
            android:onClick="btnUpdatePoiClick"
            android:text="@string/txtGuardar" />

        <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:paddingTop="5dp"
            android:text="@string/nombre"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
            android:id="@+id/txtNombre"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:inputType="text" />

        <TextView
            android:id="@+id/textView2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:paddingTop="5dp"
            android:text="@string/descripcion"
            android:textAppearance="?android:attr/textAppearanceMedium" />

        <EditText
            android:id="@+id/txtDesc"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:inputType="text" />

        <RatingBar
            android:id="@+id/rbarPuntuacion"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:numStars="5"
            android:paddingTop="5dp"
            android:stepSize="0.5" />

        <TableRow
            android:id="@+id/tableRow1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <CheckBox
                android:id="@+id/chkFavorito"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/favorito" />

            <View
                android:layout_width="40dp"
                android:layout_height="5dp" />

            <CheckBox
                android:id="@+id/chkVisitado"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/visitado" />
        </TableRow>

        <TableRow
            android:id="@+id/tableRow2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:paddingBottom="5dp"
            android:paddingTop="5dp" >

            <ToggleButton
                android:id="@+id/tgUbicacion"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:onClick="tbUbicacionOnClick"
                android:text="@string/boolGuardarPos" />

            <TextView
                android:id="@+id/textView5"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:paddingLeft="10dp"
                android:text="@string/txtReasignarUbic"
                android:textAppearance="?android:attr/textAppearanceMedium" />
        </TableRow>
....
    </LinearLayout>
</ScrollView>
```

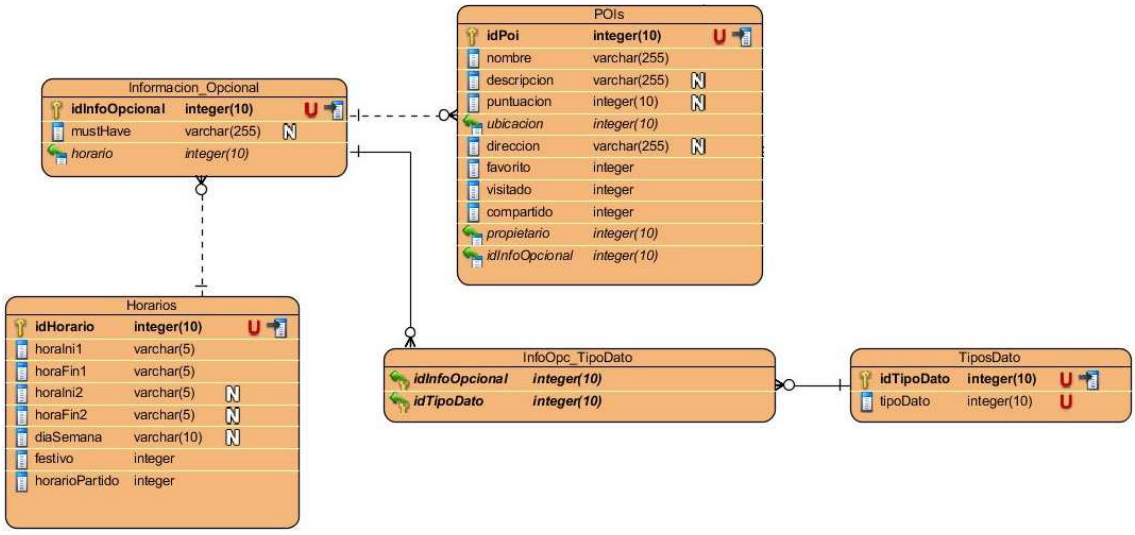
5.2. ESTRUCTURA DE LA BASE DE DADES

Com ja s'ha comentat en l'apartat 3.2 , la base de dades està construïda en *SQLite*. El disseny de la BD ha anat canviat a mesura que avançava el projecte. Des de la concepció inicial amb una sèrie de taules que després s'han suprimit o canviat per altres per optimitzar l'emmagatzematge de la informació i simplificar les consultes de l'aplicació a la BD.

IMPLEMENTACIÓ

L'estructura principal es la taula POIs, que es la que guarda el nucli de la informació. Després pengen taules per les dades opcionals, con es el cas dels *Horaris*, les *Categories* i els *Tipus de Dades* que ofereixen informació opcional com el telèfon, el nom de contacte, la direcció web, el fax, etc.

Com a exemple d'aquest procés posarem un cas concret: En un principi, es va pensar en guardar la informació opcional en una sèrie de taules que representaven fidelment els conceptes de Informació Opcional, y Tipus de Dades com es mostra en aquest diagrama:



Però una vegada es va començar a fer servir la Base de dades, es va veure que el concepte de Informació_Opcional a nivell de BD no servia per res i complicava excessivament les consultes sense donar cap benefici. Aleshores es va decidir suprimir aquesta taula i enllaçar les taules *Horario* y *Tipos de Dato* (amb la corresponent taula de relació N a N) directament amb el POI.

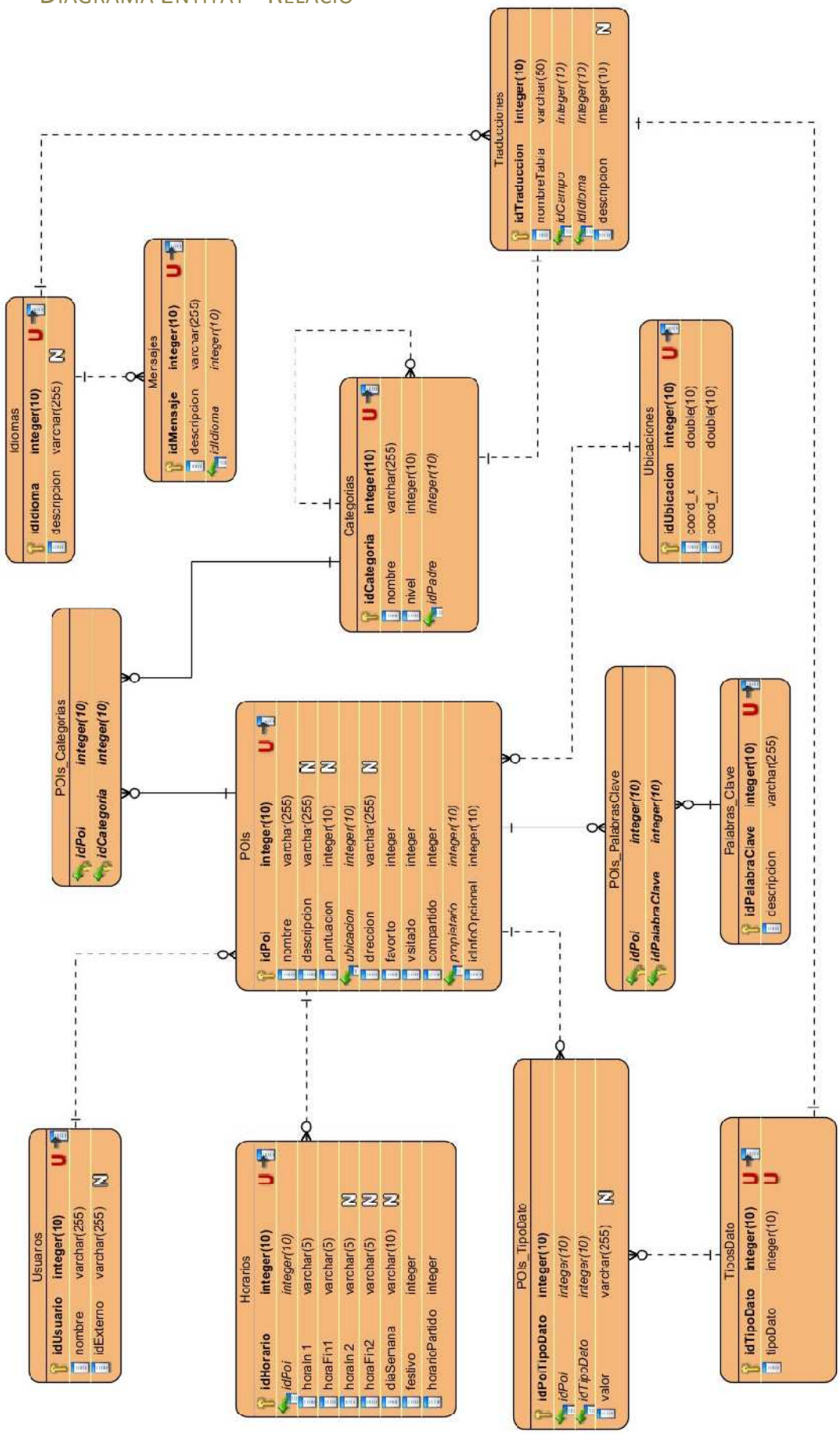
Un altre aspecte important que va afectar a la BD va ser al inclusió de les traduccions d'una part de l'aplicació per les categories i els tipus de dades opcionals. Es va passar d'un model de traducció a través de fitxers de recursos propis *d'Android* que no podia oferir el suport necessari per dur a terme aquesta tasca, a fer servir una taula de traduccions específica per poder extreure les dades necessàries amb la traducció ja incorporada pels formularis. Per això va aparèixer la taula *Traducciones*, on s'informa l'idioma, la taula d'on prové la traducció i l'identificador del camp d'aquesta taula per poder enllaçar els valors.

Mestre es fixava l'estructura final, va sorgir també un contratemps donat per la definició de *SQLite*. El fet que aquest motor de BD no accepti y doni suport a les processos (*procedures*) va fer replantejar també el paper de la BD com a simple contenidor d'informació, evitant que la lògica de l'emmagatzematge dels valors es fes a través de la capa del model de l'aplicació. És a dir, en comptes de crear un procés intern de la base de dades (posem per exemple el cas més crític, guardar un POI) fos la BD l'encarregada de gestionar la integritat de les dades i la gestió del les transaccions (*commit*, *rollback*, etc). En comptes d'això, es va haver d'adoptar un model de consultes senzilles i que el model s'encarregués de gestionar en tot moment si el procés s'executava amb èxit i si hi havia algun error, encarregar-se de la gestió.

Després d'aquests canvis, es va crear tota una interfície per connectar la BD i fer que les consultes es gestionessin a través d'una estructura que segueix el patró *Adapter*. Es va crear una interfície 'mare' que gestiona la versió actual de la BD, les variables de connexió i controla les transaccions per tal d'assegurar la estabilitat de la BD. També és la que gestiona l'accés a la variable *Context*, que es una interfície global pròpia del sistema que conté informació sobre l'aplicació. Aquesta classe permet l'accés als recursos específics de l'aplicació, entre ells la connexió a una base de dades, que es la que ens interessa ara mateix. Justament per aquest motiu es va crear aquest sistema, si es vol accedir amb més d'una classe a la BD, no es pot fer que cada classe estableixi una connexió amb un *Context* diferent com a argument. Tampoc es vàlid fer una consulta mentre una altre classe hagi obert prèviament una connexió amb la DB i escara la mantingui oberta, ni tan sols si la variable *Context* es la mateixa. Per tant es va crear un adaptador per a cada objecte de la BD que hereta de la classe *DBAdapter* mare i comparteix el seu *Context*. Així es va resoldre el problema de la integritat i les múltiples connexions a la base de dades.

A continuació es detalla un diagrama de l'estructura final de la base de dades amb les relacions entre les taules.

DIAGRAMA ENTITAT - RELACIÓ



6. PROVES

6.1. PROVES UNITÀRIES

INTRODUCCIÓ

Les proves unitàries consisteixen en provar o testejar peces de software petites a nivell de seccions, procediments, funcions i mòduls o aquelles que tinguin funcionalitats específiques. Aquestes proves es fan servir per assegurar el correcte funcionament de seccions de codi, molt més reduïdes que el conjunt i que tenen funcions concretes amb un cert grau d'independència.

IMPLEMENTACIÓ EN L'APLICACIÓ

En l''aplicació s'han fet proves unitàries en tots els mòduls que tenen funcionalitats susceptibles de generar errors. Per exemple en totes les connexions a la base de dades s'han provat els mètodes de forma individual per assegurar que son estables. Un cas concret d'aquest tipus de prova seria el testeig del mètode *getLlistaPoisFiltre(...)* que ens retorna una llista de POIs segons uns criteris de cerca. Aquest mètode es una funcionalitat que es pot aïllar i testejar sense que intervingui cap altre funció no trivial. En aquests cas, es va comprovar amb un joc de proves si hi havia algun valor que aturés l'execució de la funció o generés un error incontrolat com podria ser passar una llista de filtres buida, una llista sense inicialitzar, valors sense sentit en el camps de cerca, etc. I amb aquest banc de proves, assegurar que la nostra funció té le comportament esperat.

6.2. PROVES D'INTEGRACIÓ

INTRODUCCIÓ

Es realitzen una vegada que les proves unitàries han conclòs satisfactòriament. Amb aquestes proves s'intenta assegurar que el sistema complet, inclosos els subsistemes que composen les peces individuals més grans, funcionen correctament a l'operar en conjunt. Per simplificar-ho una mica, podríem dir que es el resultat d'agafar totes les funcions simples de les proves unitàries i crear un circuit complet o funcionalitat de l'aplicació (com les exposades als casos d'us), i a través d'aquestes confirmar que el cicle complet es comporta tal i com s'esperava.

IMPLEMENTACIÓ EN L'APLICACIÓ

L'exemple més clar d'aquest tipus de prova el podríem veure en la prova d'integració que es fa per crear un POI. En aquest circuit intervenen una gran par de les funcions d'altres proves unitàries i fins i tot altres circuits complets, ja que no només comprèn la creació del POI en si mateix, si no també la creació d'una ubicació, uns horaris, unes categories,etc. I l'assignació al POI. El circuit lògic d'aquesta prova el podríem resumir de la següent manera: Seleccionem el botó per crear un nou POI, aquesta acció desencadena unes crides a funcions que recullen els valors introduïts per l'usuari i es validen. Tot seguit s'estableixen les connexions amb al base de dades i es creen en un ordre en concret (per tal de mantenir la integritat de les dades) els registres necessaris per crear la ubicació, el POI, els horaris, les categories, les paraules clau, la informació opcional i les relacions entre tots ells. Tot aquest procés a l'hora es controla per evitar que si alguna d'aquestes accions falla, no impliqui un error en cadena que faci que el programa llanci un error no controlat. Un cop acabat el procés, es tanca la connexió s'allibera, s'informa a l'usuari del canvi fet i es redirigeix a la pantalla de visualització final del POI amb les dades informades.

7. CONCLUSIONS

INTRODUCCIÓ

En aquest apartat comentarem la situació final del projecte, la opinió i l'experiència resultant. Avaluarem de forma objectiva i subjectiva el procés i farem un repàs del que s'ha aconseguit i del que ha quedat pendent, així com possibles millores o ampliacions.

7.1. DESVIACIONS

Després del treball fet, amb una visió més clara del propòsit de l'aplicació podria dir que la primera gran dificultat trobada ha sigut el fet de ser el client i el desenvolupador al mateix temps. Moltes de les funcionalitats que esperava implantar d'una manera en concret (com a client tenia unes expectatives) després d'adquirir els coneixements sobre el llenguatge i la forma de treballar particular que té *Android*, he hagut de canviar-les per adequar-les a la seva filosofia . Per tant un dels primers reptes que he trobat durant ha sigut haver de redissenyar la navegabilitat entre les pantalles i com mostrar les opcions i accions de forma eficient i intuïtiva en una pantalla de telèfon mòbil.

Continuant amb el disseny, la següent gran dificultat ha sigut aprendre el llenguatge mentre dissenyava l'aplicació. Això ha fet que hagués de redefinir el cos de l'aplicació fins a tres vegades per tal d'optimitzar el codi i fer un nucli de l'aplicació el més correcte i optimitzat possible. Tant des del punt de vista de la BD, amb la inclusió de les traduccions de les 'categories' i els tipus de 'dades opcionals' a la pròpia BD en comptes de deixar-lo en mans del gestor d'idioma de Android, com des del punt de vista del codi del model on també es van redissenyar les classes que donaven suport al programa. Vaig crear una modelització de la BD per donar suport a les taules i uns adaptadors (*adapters*) per tal de poder accedir a les dades i transformar-los en objectes que fossin més fàcils de manipular. Per extensió, va comportar modificar la interacció la connexió de l'aplicació amb la BD per assegurar la integritat de les dades i la estabilitat de la connexió.

Tot això en realitat forma part de la gran trava real del projecte: el temps perdut en la cerca d'informació i les proves d'assaig/error per resoldre els problemes d'implementació en codi, que un principi estava contemplat i inclòs en la duració total de l'aplicació. Però a mesura que avançava em vaig adonar que l'estimació que vaig fer del temps emprat en l'aprenentatge va ser bastant alegre o laxa per dir-ho d'alguna manera. Això em va crear un llast que vaig arrossegar durant tot el temps i ha fet que el producte final no estigui tan depurat com m'hagués agradat.

7.2. MILLORES I AMPLIACIONS

A continuació detallarem millores que es poden fer al programa a partir de l'aplicació base que s'ha desenvolupat.

Els podem agrupar en dos grans blocs: Els mòduls que s'han quedat pendents degut a la falta de recursos tecnològics o que part del seu desenvolupament implicava modificacions de software de tercers que no es contemplava en aquest projecte. I per altra banda, les millores que han anat apareixent o pensant de forma natural a mesura que l'aplicació anava cobrant forma, expandint la funcionalitat o creant noves, però que per falta de temps no formaven part de la idea inicial i òbviament no s'han dut a terme.

MILLORES PENDENTS

Degut a la falta de mitjans tecnològics hi ha dos funcionalitats de la idea inicial que no s'han pogut desenvolupar i han quedat pendents:

Lectura mitjançant codis QR

La idea inicial era que mitjançant la càmera, la nostra aplicació fos capaç de escanejar un codi *QR* preparat per nosaltres amb la informació que necessita el nostre programa per guardar un POI. Però després d'analitzar el mercat, es va trobar cap *framework* que es pogués implementar en el nostre aplicació.

Amb això el primer benefici que tindríem seria poder agilitzar la lectura d'un POI i guardar-lo de forma pràcticament instantània. Però la part més interessant d'aquesta idea seria que els establiments generessin mitjançant un programa de creació de codis *QR* un codi únic pel seu establiment amb totes les dades que volguessin informar. Amb això podríem posar a la porta del seu establiment aquest codi enganxat i l'usuari només hauria de capturar-lo amb la càmera i tot seguit es guardaria de forma automàtica al seu dispositiu mòbil.

Lectura d'una targeta de visita

Aquesta millora es similar a l'anterior, però la seva execució era més complicada. El comportament es semblant: mitjançant la càmera, es podria llegir les dades d'una targeta de visita estàndard de paper i emmagatzemar-les en l'aplicació. Tindríem la

possibilitat de llegir qualsevol establiment o fins i tot contactes sense necessitat de estar físicament davant del punt per poder llegir el codi. Així estarien cobertes pràcticament totes les possibilitats per obtenir la informació d'un establiment.

Però per desgracia, ara mateix aquesta millora es inviable, ja que els *frameworks* capaços de fer això només funcionen amb adreces en format americà i no hi ha cap companyia que doni suport a adreces en format europeu.

MILLORES GLOBALS FORA DE LA CONCEPCIÓ INICIAL DEL PROJECTE

A mesura que es va anar desenvolupant l'aplicació em vaig anar adonant que no només es podrien catalogar botigues, si no qualsevol lloc susceptible de ser un punt que es pogués geoposicionar. Amb això s'obre una nova forma de plantejar l'aplicació, ja que no només serveix per localitzar botigues i es restringeix a l'àmbit urbà, si no que es podria fer servir per diferents propòsits, com per exemple:

- **Rutes per muntanya:** Punts útils a la muntanya, refugis, miradors, etc.
- **Guia turística per estrangers:** ubicar monuments, poder traçar rutes per visitar-los a peu, amb transport públic, etc.

També em vaig plantejar formes més lucratives de fer servir l'aplicació,com a possible font de beneficis.

La possibilitat de interaccionar amb el sistema *wifi/bluetooth* de les botigues creant un sistema d'alertes per enviar promocions. Es a dir poder enviar promocions als telèfons dels usuaris quan entrin a la botiga o passin pel davant. Un cas típic seria quan un usuari es para davant d'un establiment , es para al davant i fa una ullada del que hi ha. En aquest moment s'enviaria un missatge amb el POI de la botiga i promocions especials per animar a l'usuari a entrar a la botiga.

Una altre forma d'enfocar l'aplicació en termes econòmics es canviar la idea de fons i oferir a les empreses estar presents a l'aplicació. Aquesta idea desvirtua una mica la idea inicial de l'aplicació, que es poder evitar la publicitat de POIs que no ens interessin. En aquest cas, l'usuari partiria d'una aplicació plena de POIs amb les botigues anunciades. Per tal de mantenir la raó de ser de l'aplicació, l'acondicionament d'aquesta al seu gust seria en sentit invers, es a dir, l'usuari tindria l'opció d'escollir allò que vol veure. Tindria una llista de POIs '*banejats*', podent escollir una botiga en contret o totes les botigues d'una cadena. Per exemple un usuari que no vulgui veure cap botiga de *Zara* podria bloquejar aquesta cadena sencera i no veure cap establiment relacionat al seu mapa de POIs. Aprofitant la inclusió de les categories i

paraules clau, bloquejar botigues que venguin roba de pell d'animal o veure només botigues que venguin roba de materials naturals, sense sintètics.

7.3. CONCLUSIÓ FINAL

CONCLUSIONS DEL PROJECTE

Per resumir, podria dir que malgrat els continus canvis de planificació i les traves que he trobat per desenvolupar l'aplicació tal i com m'hagués agradat, l'experiència ha sigut gratificant. M'ha ajudat a aprendre un llenguatge nou, entendre més el funcionament del cicle de creació d'un aplicació i les implicacions que té cada part. He pogut fer un projecte propi que volia fer servir com a eina per solucionar un problema real personal i també aprofitat la oportunitat per portar-lo un pas més enllà i adaptar-lo per poder estendre la seva funcionalitat en un futur i amb el temps poder llançar una aplicació a *l'Android Market* o per poder treure'n un profit econòmic. I també com es obvi, aconseguir el títol de la carrera, que es en definitiva el motor motivador principal d'aquest projecte.

8. BIBLIOGRAFIA

DOCUMENTS REFERENCIATS EN L'ESTUDI DE VIABILITAT.

En el desenvolupament d'aquest projecte s'ha obtingut la informació principalment de webs de referència per desenvolupadors. A continuació es detallen les més rellevants a mode de exemple:

- *http://www.uab.cat/Document/639/153/normativaProjectesEEsabadell.pdf* : Normativa de projectes d'enginyeria tècnica.
- *https://www.agpd.es/portalwebAGPD/canaldocumentacion/informes_juridicos/reglamento_lopd/index-ides-idphp.php* : Llei Orgànica de Protecció de Dades
- http://developer.android.com/intl/es/training/index.html : Tutorial oficial de Google per a desenvolupadors Android, el més complet malgrat que a vegades es massa tècnic i li falten exemples per il·lustrar els conceptes.
- http://code.google.com/p/tree-view-list-android/ : Web del projecte que es va fer servir com a llibreria externa per implementar la vista d'arbre a l'aplicació.
- https://developers.google.com/maps/documentation/directions/ : Pàgina oficial de Google amb les APIs de Google Maps
- https://developers.google.com/maps/signup : Formulari per solicitar la key per a desenvolupar aplciacions que facin servir GoogleMaps (només serveix per desenvolupar)
- http://www.sgoliver.net/blog/ : Blog divulgatiu en castellà per principiants. Un dels millors que he trobat, amb codi d'exemple i que cobreix quasi tots els camps principals.
- http://www.javaya.com.ar/androidya/ : Web divulgativa en castellà amb bona documentació del estil de l'anterior esmentada, tot i que un amica inferior.
- http://research.cs.queensu.ca/~dingel/cisc836_F11/androidWithActivities.pdf : Estudi sobre l'adatació del patró MVC a Android (Engineering Android Applications Based on UML Activities)
- http://stackoverflow.com : Web col·laborativa per desenvolupadors on es pot trobar tot de qualsevol llenguatge. Actualment quasi tots els problemes està en algun del seus fòrums.
- http://www.androidhive.info/ : Web en anglès amb articles sobre FAQs solucionades.
- http://www.softwarepassion.com/ : Blog similar a l'anterior comentat.
- http://android-er.blogspot.com.es/ Web d'articles amb casso solucionats.

9. ANNEXES

9.1. GLOSSARI

DEFINICIÓ DELS TERMES, ACRÒNIMS I ABREVIACIONS UTILITZATS A LA MEMÒRIA

- **ACID:** Conjunt de característiques necessàries per a que una sèrie d'instruccions puguin ser considerades com una transacció. El seu acrònim en anglès significa: '*Atomicity, Consistency, Isolation and Durability*'.

- **Android:** Sistema operatiu per dispositius mòbils desenvolupat per *Android Inc*., una firma comprada per *Google* en el 2005. Android està basat en una versió modificada del *Kernel* de *Linux*.

- **Apache (llicència):** És una llicència de programari lliure creada per la *Apache Software Foundation* (ASF). la llicència *Apache* permet a l'usuari del programari plena llibertat d'ús per a qualsevol propòsit, distribuir-lo, modificar-lo, i distribuir versions modificades del programari.

- **API *Google Maps*:** Llibreria que permet interactuar el software amb els mapes i les bases de dades de *Google Maps*.

- **Dalvik:** És la màquina virtual que utilitza la plataforma per a dispositius mòbils Android. *Dalvik* ha estat dissenyada per Dan Bornstein amb contribucions d'altres enginyers de *Google*. *Dalvik* està optimitzada per requerir poca memòria i està dissenyada para permetre executar diverses instancies de la màquina virtual simultàniament, delegant en el sistema operatiu subjacent el suport d'aïllament de processos, gestió de memòria i fils.

- **Diagrama de Gantt:** Cronograma del projecte. És un eina gràfica que permet mostrar el temps de dedicació previst per diferents tasques o activitats al llarg d'un temps determinat.

- **Eclipse:** Entorn de desenvolupament integrat (IDE) de codi obert multi plataforma per desenvolupar aplicacions.

- **Framework:** Un framework (o marc de treball en català) és una infraestructura digital conceptual y tecnològica que dóna suport, normalment mitjançant mòduls de software concrets, en base a la qual un altre projecte de software pot ser organitzat I desenvolupat mes fàcilment.

- ***Google Maps*:** És el nom que rep el servei gratuït de *Google* d'aplicacions de mapes en la web. Ofereix imatges de mapes desplaçables, així com fotos de satèl·lit del món sencer i el càlcul de rutes entre diferents ubicacions.

- **GPS:** (*Global Positioning System*). Permet determinar en tot el món la posició d'un objecte, una persona, vehicle o nau amb una precisió que pot ser de metres o centímetres, depenent de la tecnologia.

- **Mercurial:** Eina per la gestió i control de projectes.
- **Microsoft Project:** programa de Microsoft utilitzat per la gestió de projectes
- **Milestone:** Punt de control del projecte, lliurament de documentació, assoliment d'un objectiu, realització d'una activitat.
- **MVC:** (Model-Vista-Controlador) Es un patró o model d'abstracció de desenvolupament de software que separa les dades d'una aplicació, l'interfície i la lògica del negoci en tres components distints. Aquest estil de desenvolupament va ser descrit per primer cop en 1979 per Trygve Reenskaug.
- **OpenCore:** *Framework* que implementa de sèrie Android per a la gestió del contingut multimèdia (vídeo i àudio còdecs, interfícies, etc.)
- **Open Handset Alliance:** És una aliança comercial de 84 companyies que es dedica a desenvolupar estàndards oberts per dispositius mòbils. Alguns dels seus membres son *Google, HTC, Dell, Intel, Motorola, Qualcomm, Texas Instruments, Samsung, LG, T-Mobile, Nvidia y Wind River Systems.*
- **Open Source:** Organització dedicada a la promoció del codi obert. Amb codi obert es coneix el software distribuït i desenvolupat lliurement. Té un punt de vista més orientat als beneficis pràctics de poder accedir al codi, que a les qüestions ètiques i morals les quals es destaquen en el software lliure.
- **POI:** Sigles de Point Of Interest en anglès o PDI (Punt d'Interès) en català i castellà. Punt útil per un usuari geoposicionat mitjançant unes coordenades donades per una latitud y longitud concretes.
- **SQLite:** És un sistema de gestió de Base de Dades Relacional contingut en una petita biblioteca en C.
- **SDK Android:** Kit de desenvolupament de software específic per Android.
- **Serialitzar/Deserialitzar:** Procés que consisteix en codificar un objecte a través d'un mitjà d'emmagatzematge (com por ser un arxiu o un *búfer* de memòria) amb la finalitat d'enviar-lo a través d'una connexió de xarxa com una sèrie de bytes o en un format humanament més llegible com XML.
- **WBS:** Work Breakdown Structure.
- **Webkit:** Plataforma per aplicacions que funciona com a base per poder renderitzar i implementar navegadors web als dispositius Android.