Desenvolupament d'aplicació de consulta de cases rurals per Android

Gerard Garcia Porlan

Enginyeria Informàtica

Consultor: Jordi Ceballos Villach

30 de Maig de 2011

Resum

Els dispositius mòbils i les seves prestacions han evolucionat notablement des de l'inici de la seva aparició fins l'actualitat. El sistema Android va ser l'aposta de Google l'any 2005 per entrar en el mercat dels dispositius mòbils i des de llavors la quota de mercat d'aquest sistema basat en el kernel de Linux i una màquina virtual Java anomenada Dalvik no ha parat de créixer.

Aquest projecte final de carrera pretén realitzar el desenvolupament d'una aplicació per al sistema Android, programada amb el llenguatge Java, que consulta una base de dades de cases rurals i aprofita les característiques i funcionalitats dels dispositius mòbils actuals com el posicionament geogràfic per GPS o la connexió a Internet.

Aquesta aplicació utilitza Serveis web per obtenir les dades d'un servidor remot i mostrar-les a l'usuari de forma que pugui visualitzar les dades, veure les cases més properes en un mapa, poder iniciar una trucada o l'enviament d'un correu electrònic.

Índex de continguts

Índex d'il·lustracions

1. Introducció

1.1. Justificació i context

Els dispositius mòbils han evolucionat de forma molt notable en els últims anys. La generalització de l'accés a Internet i la millora de les xarxes sense fils ha fet d'aquest tipus de dispositius un punt molt important d'accés a la xarxa i als serveis que hi podem trobar.

També l'aparició de noves funcionalitats com el posicionament mitjançant GPS i la generalització de l'ús de pantalles tàctils ha propiciat un creixement molt important en el nombre d'aparells existents al mercat i en conseqüència del desenvolupament d'aplicacions per aquest tipus de dispositius.

En aquest context, la compra per part de Google l'any 2005 de la companyia Android Inc. i la creació de la Open Handset Alliance per part d'un conjunt d'empreses capdavanteres en el sector per potenciar el desenvolupament d'estàndards oberts per als dispositius mòbils va suposar l'aparició del sistema Android al mercat amb un gran suport al darrera. Google va alliberar gran part del codi d'Android sota la llicència Apache i va crear Android Market, una botiga d'aplicacions on-line.

Aquest projecte consisteix en el desenvolupament d'una aplicació per al sistema Android que mitjançant l'ús de la connexió a Internet i aprofitat les funcionalitats dels dispositius mòbils abans esmentades permet la consulta de les dades d'una xarxa de cases rurals de Catalunya. Aquestes dades es troben en un servidor i són gestionades per l'aplicació OpenERP en un sistema implantat i desenvolupat per l'empresa ARGUS.net Tecnologia Creativa.

1.2. Descripció del projecte

L'objecte d'aquest projecte és el desenvolupament d'una aplicació per al sistema Android que utilitzi els Serveis web per a la consulta de les dades d'un servidor que disposa de l'aplicació OpenERP que conté les dades de diferents cases rurals de Catalunya.

L'aplicació permetrà obtenir del servidor la informació sobre les cases disponibles i mostrar-les en dos formats:

- En forma de llistat: l'usuari podrà veure un llistat de cases ordenat alfabèticament o bé ordenat en funció de la proximitat a la localització on es troba l'usuari fent ús del GPS.

- En format mapa: l'usuari veurà la situació de les cases en un mapa on es mostrarà una àrea circular centrat en la posició actual de l'usuari. L'usuari podrà modificar el seu radi per tal de veure una zona més àmplia o bé una zona diferent a la que es troba actualment.

Seleccionant una casa tant de la llista com del mapa es mostrarà la fitxa amb 4 apartats:

- Informació general: nom, descripció, adreça, correu electrònic, telèfon i enllaç al mapa per veure la situació de la casa. Seleccionant el número de telèfon s'iniciarà una trucada i seleccionant el correu electrònic s'iniciarà el programa de correu amb l'adreça de la casa preparada per a l'enviament.

- Galeria fotogràfica: recull de fotografies de la casa. L'usuari podrà seleccionar una fotografia i aquesta es mostrarà ampliada.

- Informació dels serveis dels que disposa la casa.

- Informació de preus de les diferents habitacions en funció de la temporada.

Es contempla la possibilitat d'afegir de forma opcional, en funció del temps i la dificultat, una funcionalitat que permeti veure en realitat augmentada mitjançant la càmera del dispositiu la situació de les diferents cases rurals.

1.3. Objectius

Els objectius que es volen assolir en la realització d'aquest projecte són els següents:

- Posar en pràctica els coneixements adquirits durant la carrera en diferents àmbits com el desenvolupament de programari, les comunicacions sense fils i la gestió de projectes entre d'altres.

- Adquirir i posar en pràctica els coneixements necessaris sobre la plataforma Android i el desenvolupament de les seves aplicacions.

- Aprofundir en el coneixement sobre els dispositius mòbils i el desenvolupament d'aplicacions que aprofitin les seves funcionalitats.

- Utilitzar tecnologies de comunicació basades en serveis web, XML-RPC i aprofundir en el seu coneixement i aplicació pràctica.

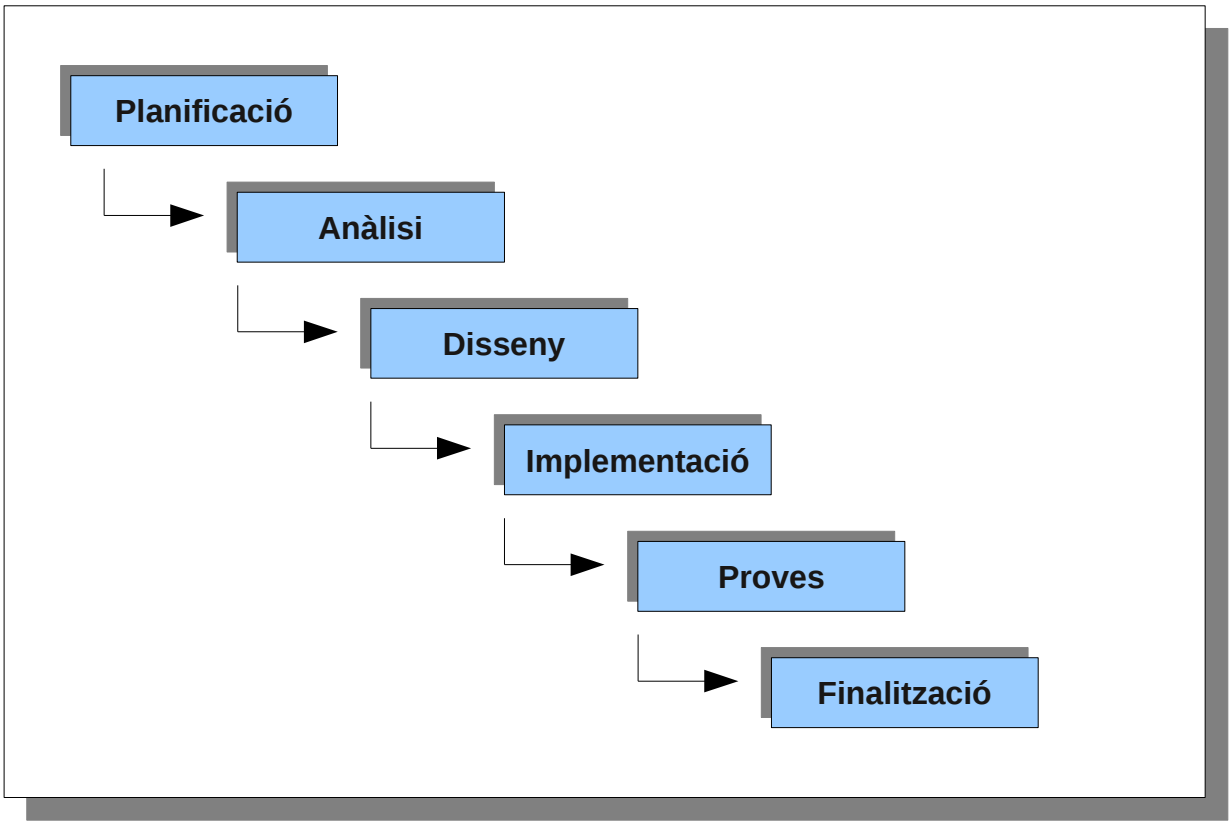
- Millora en el coneixement i ús de l'entorn de desenvolupament Eclipse.

• Aprendre les possibilitats de consulta i tractament de les dades que ofereix l'aplicació OpenERP.

• Obtenir una memòria i presentació del projecte i una aplicació amb les funcionalitats esperades que sigui instal·lable i utilitzable en un dispositiu mòbil que compleixi els requeriments.

1.4. Planificació

El projecte s'ha dut a terme seguint el cicle de vida clàssic (lineal seqüencial) en el desenvolupament d'aplicacions i tingut les etapes que es mostren a la figura següent:



Il·lustració 1. Cicle de vida clàssic

Les tasques realitzades s'han dividit en quatre etapes on al final de cadascuna s'ha fet l'entrega de la documentació i parts del producte realitats.

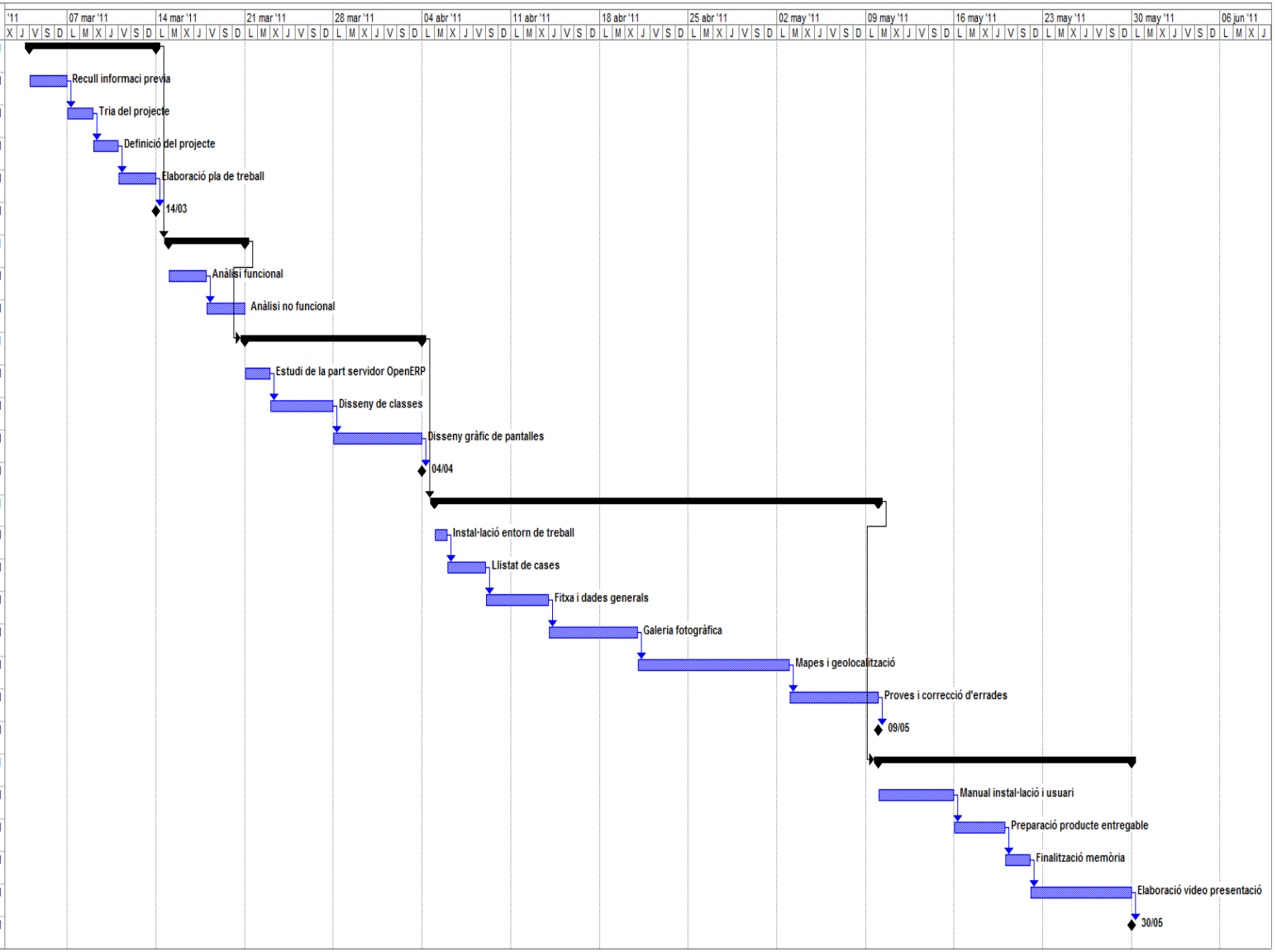
A continuació es mostra en forma de taula les tasques realitzades en cadascuna d'aquestes quatre etapes:

Planificació del projecte	
Tasca	**Descripció**
Definició del projecte	Definició de les característiques i abast.
Planificació del projecte	Planificació temporal de tasques i fites.
Creació del pla de treball	Redacció del document
Lliurament PAC1	Entrega del document amb el pla de treball
Anàlisi i disseny	
Definició de requisits funcionals	Establir requisits funcionals i casos d'ús
Definició de requisits no funcionals	Establir i definir requisits no funcionals
Estudi de la part servidor	Estudi del funcionament i obtenció de dades del servidor OpenERP
Definició classes i la seva organització	Establir classes, paquets en que s'organitzaran i funció de cadascuna d'elles.
Instal·lació entorn de treball	Instal·lació Eclipse i Android SDK
Disseny de pantalles i prototipus	Creació prototipus amb pantalles sobre Eclipse
LliuramentPAC2	Entrega del document anb l'anàlisi i disseny
Implementació i proves	
Implementació llistat de cases	Implementació obtenció del llistat de cases
Implementació fitxa de la casa	Implementació de la fitxa de la casa
Implementació galeria de fotos	Implementació de la galeria fotogràfica
Implementació del mapa	Implementació del mapa
Proves i correcció d'errors	Proves, correcció d'errors i control d'errors i alertes
Redacció manual instal·lació	Redacció del manual d'instal·lació a l'emulador
Lliurament PAC3	Lliurament document implementació i codi
Finalització	
Elaboració de la memòria	Redacció de la memòria
Elaboració de la presentació	Realització presentació i vídeo
Lliurament final	Lliurament de la memòria i presentació

A continuació es recullen les fites més importants del desenvolupament del projecte i posteriorment un diagrama de Gantt on es pot observar la planificació:

Data	Fita
14/03/2011	Lliurament PAC1: planificació
20/03/2011	Finalització de l'anàlisi
04/04/2011	Lliurament PAC2: anàlisi i disseny
09/05/2011	Lliurament PAC3: implementació
30/05/2011	Lliurament final: memòria i vídeo

Id	Nombre de tarea	Comienzo	Fin
1	**Anàlisi previ i definició**	**vie 04/03/11**	**lun 14/03/11**
2	Recull d'informació previ	vie 04/03/11	dom 06/03/11
3	Tria del projecte	lun 07/03/11	mar 08/03/11
4	Definició del projecte	mié 09/03/11	jue 10/03/11
5	Elaboració pla de treball	vie 11/03/11	dom 13/03/11
6	Lliurament PAC1	lun 14/03/11	lun 14/03/11
7	**Anàlisi de requeriments**	**mar 15/03/11**	**dom 20/03/11**
8	Anàlisi funcional	mar 15/03/11	jue 17/03/11
9	Anàlisi no funcional	vie 18/03/11	dom 20/03/11
10	**Disseny**	**lun 21/03/11**	**lun 04/04/11**
11	Estudi de la part servidor OpenERP	lun 21/03/11	mar 22/03/11
12	Disseny de classes	mié 23/03/11	dom 27/03/11
13	Disseny gràfic de pantalles	lun 28/03/11	dom 03/04/11
14	Lliurament PAC2	lun 04/04/11	lun 04/04/11
15	**Implementació**	**mar 05/04/11**	**lun 09/05/11**
16	Instal·lació entorn de treball	mar 05/04/11	mar 05/04/11
17	Llistat de cases	mié 06/04/11	vie 08/04/11
18	Fitxa i dades generals	sáb 09/04/11	mié 13/04/11
19	Galeria fotogràfica	jue 14/04/11	mié 20/04/11
20	Mapes i geolocalització	jue 21/04/11	lun 02/05/11
21	Proves i correcció d'errades	mar 03/05/11	lun 09/05/11
22	Lliurament PAC3	lun 09/05/11	lun 09/05/11
23	**Finalització del projecte**	**mar 10/05/11**	**dom 29/05/11**
24	Manual instal·lació i usuari	mar 10/05/11	dom 15/05/11
25	Preparació producte entregable	lun 16/05/11	jue 19/05/11
26	Finalització memòria	vie 20/05/11	sáb 21/05/11
27	Elaboració video presentació	dom 22/05/11	dom 29/05/11
28	Lliurament final	lun 30/05/11	lun 30/05/11



Il·lustració 2: Diagrama de Gantt de la planificació del projecte

1.5. Eines i tecnologies utilitzades

Les tecnologies que s'han utilitzat per realitzar aquest projecte són:

- Llenguatge de programació **JAVA** i **XML**: per la programació del codi de l'aplicació.
- Entorn de desenvolupament **Eclipse**: per al desenvolupament del codi de l'aplicació.
- **Android SDK**: framework per al desenvolupament d'aplicacions per al sistema Android que inclou l'emulador on s'han fet proves amb l'aplicació.
- **XML-RPC**: protocol de comunicació amb l'aplicació servidor. S'ha utilitzat una llibreria per a Android que implementa aquest protocol.
- **OpenERP**: aplicació servidor que emmagatzema i gestiona les dades. S'ha utilitzat per crear dades de prova per l'aplicació.
- **HTC Hero i HTC Desire**: telèfons mòbils amb el sistema operatiu Android on s'ha instal·lat i provat l'aplicació.
- **Microsoft Project**: per la realització de la planificació del projecte.
- **LibreOffice Writer**: per la redacció dels documents del projecte.
- **LibreOffice Draw**: per la realització de diagrames gràfics.
- Editor d'imatges **GIMP**: per a l'edició de les imatges utilitzades en l'aplicació i en els documents.
- **LibreOffice Impress**: per a la realització de la presentació Power Point.
- **Gtk-recordMydesktop**: per realitzar el vídeo de presentació.
- **OpenERP client**: programa client per a connectar amb la base de dades i introduir i modificar dades.

1.6. Productes obtinguts

Durant la realització d'aquest projecte s'han generat els següents productes:

- Document amb el pla de treball.
- Document amb l'anàlisi i disseny de l'aplicació.
- Aplicació per al sistema operatiu Android de la qual es disposa de:
    - Codi font de l'aplicació.
    - Binari instal·lable en un sistema Android.
- Manual d'instal·lació de l'aplicació.

- Memòria del projecte (present document).
- Vídeo de presentació del projecte.

1.7. Contingut de la memòria

En aquesta memòria del projecte es recull la informació i el contingut més important de la realització del projecte final de carrera.

Primer es fa una descripció de l'escenari en el que es desenvolupa aquest projecte i seguidament es descriuen els requisits funcionals i no funcionals. Tot seguit s'analitzen els casos d'ús i es fa una descripció de cadascun d'ells.

Després, en l'apartat de disseny, hi trobem la descripció de l'arquitectura del sistema on s'explica el funcionament de la part servidor i com és la comunicació entre l'aplicació i la part servidor. En la segona part del disseny s'hi troba el disseny de pantalles que formen el prototipus inicial de l'aplicació.

Després del disseny trobem l'apartat corresponent a la implementació. En aquest apartat s'expliquen els elements claus i més importants de la fase d'implementació de forma que es pugui entendre de forma bàsica el funcionament i aquelles parts més rellevants i que han suposat una major dificultat.

Finalment, trobem el capítol amb les conclusions extretes de la realització del projecte i tot seguit els capítols amb el glossari i les fonts d'informació utilitzades.

2. Requisits

En els següents apartats es descriu la situació de partida sobre la qual es desenvolupara el projecte i els requisits que ha de tenir l'aplicació.

2.1. Escenari inicial

L'empresa ARGUS.net Tecnologia Creativa, per la qual treballo, es troba actualment desenvolupant un projecte per a un client que disposa d'una xarxa de cases rurals a Catalnuya. En el marc d'aquest projecte web, s'ha implantant una aplicació OpenERP en un servidor al qual es pot accedir a través d'Internet i que conté les dades de de les cases d'aquesta xarxa de cases rurals.

En aquest escenari, la possibilitat de desenvolupar una aplicació per al sistema Android que permeti consultar les dades de les diferents cases rurals des de diferents dispositius mòbils ha suposat per una banda la possibilitat d'estendre aquest entorn a una nova plataforma i per altra banda l'adquisició de nou coneixement en aquest àmbit per a una empresa on la recerca i innovació és un element clau.

Al ser el projecte d'ARGUS.net un projecte encara en desenvolupament i per tant subjecte a canvis, l'aplicació que es desenvolupa en aquest projecte de final de carrera suposarà la base sòlida sobre la qual després es podran afegir noves funcionalitats que millorin el producte final.

Aquesta base serà una aplicació Android que aprofiti les característiques dels dispositius mòbils i permeti la connexió amb el sistema OpenERP a través d'Internet, la consulta de dades d'aquest sistema, la visualització en un mapa de les cases i les funcionalitats de GPS, trucades i enviament de correu electrònic.

2.1. Requisits funcionals

En aquest apartat es mostren les funcionalitats que l'usuari pot utilitzar quan fa ús de l'aplicació.

A continuació es detallen amb una breu descripció dels requisits funcionals que ha de tenir l'aplicació:

- **Obtenir llistat de cases:** mostra un llistat de les cases rurals disponibles.

- **Seleccionar ordenació:** permet seleccionar si es vol ordenar el llistat alfabèticament o per proximitat a la posició de l'usuari.

- **Iniciar mapa:** mostra el mapa a la pantalla amb la posició actual de l'usuari.

- **Desplaçar mapa:** desplaça la visualització del mapa segons el moviment de l'usuari.

- **Fer zoom in/out del mapa:** amplia o redueix l'escala del mapa.

- **Modificar perímetre del mapa:** permet ampliar o reduir la circumferència que ens mostra en el mapa la situació de les cases dins aquest perímetre.

- **Seleccionar casa:** escollir una casa en el llistat o bé en el mapa per veure'n la fitxa.

- **Trucar casa:** inicia una trucada a la casa quan l'usuari clica sobre el telefon.

- **Enviar correu:** iniciar escriptura de correu electrònic amb l'adreça de correu de la casa rural com a destinatari predeterminat.

- **Seleccionar fotos:** mostra l'apartat de la fitxa corresponent a la galeria fotogràfica.

- **Desplaçar fotos:** permet desplaçar el llistat de fotos a una banda o l'altra per veure'n tota la seqüència.

- **Ampliar foto:** selecció d'una foto que passa a veure's de forma ampliada.

- **Seleccionar serveis:** mostra l'apartat de la fitxa corresponent als serveis de la casa.

- **Seleccionar preus:** mostra l'apartat de la fitxa corresponent als preus de la cas segons la temporada.

L'actor d'aquests casos d'ús és únic ja que l'aplicació no disposa de perfils d'usuari diferents sinó que un mateix usuari pot fer ús de totes les funcionalitats de l'aplicació.

2.2. Requisits no funcionals

- La interfície gràfica de l'aplicació ha de ser fàcil d'utilitzar i adaptada gràficament per ser utilitzada en un dispositiu mòbil i per tant amb una pantalla de dimensions reduïdes.

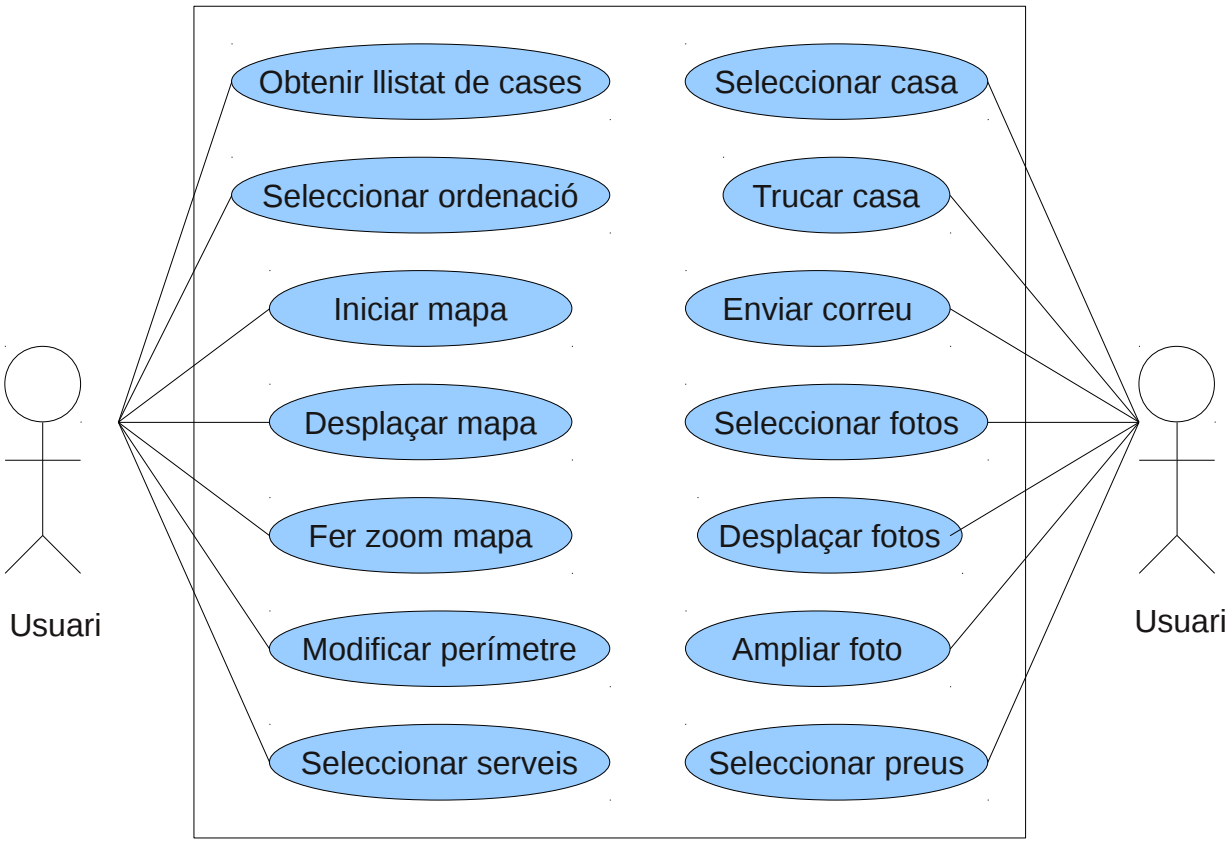
- L'aplicació ha de fer ús les funcionalitats dels dispositius mòbils com el fet que tingui una pantalla tàctil i dels recursos que el sistema Android ofereix a l'hora d'implementar funcionalitats en aquest tipus d'interfície.

3. Anàlisi del sistema

En aquest apartat es mostren els resultats de l'anàlisi del sistema. Primer es mostren els casos d'ús de l'aplicació de forma gràfica i posteriorment descrita i per acabar es mostra les característiques del sistema servidor.

3.1. Diagrama de casos d'ús

En el següent diagrama podem observar el diagrama de casos d'ús de l'aplicació:



Il·lustració 3. Diagrama de casos d'ús

Podem observar que hi ha un únic actor tot i que es mostri repetit en el diagrama per una millor disposició gràfica. Hi ha per tant un únic actor que interactua amb l'aplicació i pot utilitzar totes les seves funcionalitats.

3.2. Descripció dels casos d'ús

A continuació es mostra una breu descripció dels diferents casos d'ús.

3.2.1. Obtenir llistat de cases

Aquest cas realitza l'obtenció del llistat de cases disponibles al sistema. L'usuari selecciona obtenir el llistat i l'aplicació realitza una connexió amb la base de dades i obté les dades bàsiques per mostrar una llista de les cases disponibles.

Mostra un avís si no hi ha una connexió a Internet disponible i també informa de que no hi ha cases disponibles si pot fer la connexió però no obté cap resultat.

3.2.2. Seleccionar ordenació

Amb el llista de cases visible l'usuari pot visualitzar un menú on pot seleccionar si vol ordenar el llistat de forma alfabètica o bé en funció de la proximitat de la casa a la posició actual de l'usuari.

3.2.3. Iniciar mapa

L'usuari selecciona la opció d'iniciar el mapa i accedeix a una pantalla on es mostra un mapa centrat en la posició de l'usuari i un cercle representant l'àrea més propera a l'usuari.

Si l'aplicació no pot obtenir la posició actual de l'usuari es mostra un mapa centrat en una posició per defecte del mapa de Catalunya.

3.2.4. Desplaçar mapa

L'usuari pot desplaçar la visualització del mapa. Desplaçant el dit sobre la pantalla mentre la manté pressionada l'usuari pot desplaçar la visualització per tal de veure altres parts del mapa.

3.2.5. Fer zoom in/out del mapa

Quan l'usuari pressiona sobre els botons de zoom -/+ del mapa la visualització canvia mostrant una escala més gran o mes petita i veient una àrea del territori més gran o més petita.

3.2.6. Modificar perímetre del mapa

Quan l'usuari pressiona sobre la fletxa del cercle i la desplaça per la pantalla pot ampliar o reduir la mida del cercle fent que es mostrin les cases rurals existents dins del perímetre i es deixin de mostrar les que queden fora d'aquest.

3.2.7. Seleccionar casa

L'usuari selecciona una casa ja sigui des del mapa clicant sobre la icona corresponent o bé des del llistat de cases. L'aplicació realitza una connexió amb el servidor i obté les dades de la casa que l'usuari ha seleccionat. Tot seguit l'usuari visualitza la fitxa de la casa seleccionada.

Es mostra un avís si no hi ha una connexió disponible i per tant no es pot realitzar la connexió amb el servidor.

3.2.8. Trucar casa

Quan l'usuari visualitza la fitxa d'una casa pot iniciar una trucada pressionant sobre el telèfon que es mostra a la fitxa de la casa.

3.2.9. Enviar correu

Quan l'usuari visualitza la fitxa d'una casa pot iniciar l'aplicació d'enviament de correus del sistema Android pressionant sobre l'adreça de correu electrònic de la casa. S'inicia l'aplicació de correu amb l'adreça de la casa ja introduïda en l'apartat de destinatari.

3.2.10. Seleccionar fotos

Quan visualitza la fitxa de la casa l'usuari pot accedir a la galeria de fotografies pressionant sobre la pestanya corresponent a les fotografies.

3.2.11. Desplaçar fotos

En la secció de la galeria fotogràfica es mostren totes les fotografies disponibles. L'usuari pot desplaçar el conjunt de fotografies cap a l'esquerra o cap a la dreta pressionant sobre el conjunt i desplaçant amb el dit en una de les dues direccions.

3.2.12. Ampliar foto

Pressionant sobre qualsevol de les fotografies de la casa aquesta es mostra de forma ampliada a la galeria fotogràfica.

3.2.13. Seleccionar serveis

Quan esta visualitzant la fitxa de la casa l'usuari pot accedir a l'apartat dels serveis de la casa pressionant sobre la pestanya corresponent situada a la part superior de la fitxa de la casa.

3.2.14. Seleccionar preus

Quan visualitza la fitxa de la casa l'usuari pot accedir a l'apartat que mostra els diferents preus pressionant sobre la pestanya corresponent situada a la part superior de la fitxa.

4. Disseny

4.1. Arquitectura del sistema

Ens trobem davant d'un sistema amb una arquitectura client-servidor format per l'aplicació Android a la part client i el sistema OpenERP a la banda del servidor.

A continuació es detallen les característiques de cadascuna d'aquestes parts.

4.1.2. Servidor OpenERP

Les dades de les cases rurals que consulta l'aplicació Android es guarden en l'aplicació OpenERP.

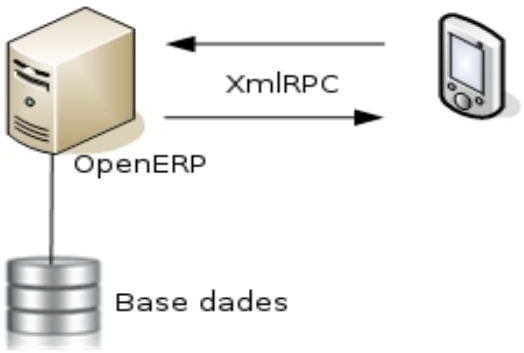
Aquesta part servidor esta formada per:

- Aplicació OpenERP: és un sistema ERP i CRM de programari lliure programat amb el llenguatge Python que disposa d'una interfície de comunicació que utilitza el protocol XML-RPC. Utilitza PostgreSQL com a sistema gestor de la base de dades.

- PostgreSQL: és un sistema de gestió de bases de dades relacional orientat a objectes. És un sistema de programari lliure publicat sota la llicència BSD.

Aquest sistema es troba instal·lat en un servidor amb el sistema operatiu Linux, versió Debian 5.1.

En la següent figura podem observar una imatge de la representació física d'aquesta arquitectura:



Il·lustració 4. Arquitectura física

4.1.3. Consulta de dades

El sistema OpenERP disposa de diverses formes d'accés a la informació. Permet la consultat de dades a través del protocol HTTP ja que porta integrat un servidor web.

Per a l'aplicació Android però, la solució que s'ha triat és la utilització del protocol XML-RPC pel qual OpenERP també disposa d'una interfície.

Per realitzar aquesta interacció entre l'aplicació i el servidor s'utilitzarà una llibreria XML-RPC per Android i una classe Java de la qual s'adaptarà el codi per a que funcioni amb aquesta llibreria.

4.1.3.1. Interacció client - servidor

L'aplicació obtindrà les dades en el moment en que les necessiti. Quan l'usuari vulgui obtenir el llistat o el mapa l'aplicació obtindrà els noms de les cases i la seva posició però no la resta de dades. En el moment en que l'usuari vulgui veure la fitxa d'una casa es realitzarà la consulta per obtenir aquestes dades.

La classe Java que ens permet interactuar des de l'aplicació amb el servidor ens ofereix les següents funcionalitats:

- Autenticació contra l'aplicació OpenERP. Això permet assegurar que és aquesta aplicació i no algú altre qui hi intenta accedir.

- Cerca: disposa d'una ordre **search** que el que fa és cercar i retornar els identificadors del tipus d'objecte demanat. Per exemple: les cases rurals, els serveis, els preus, etc..

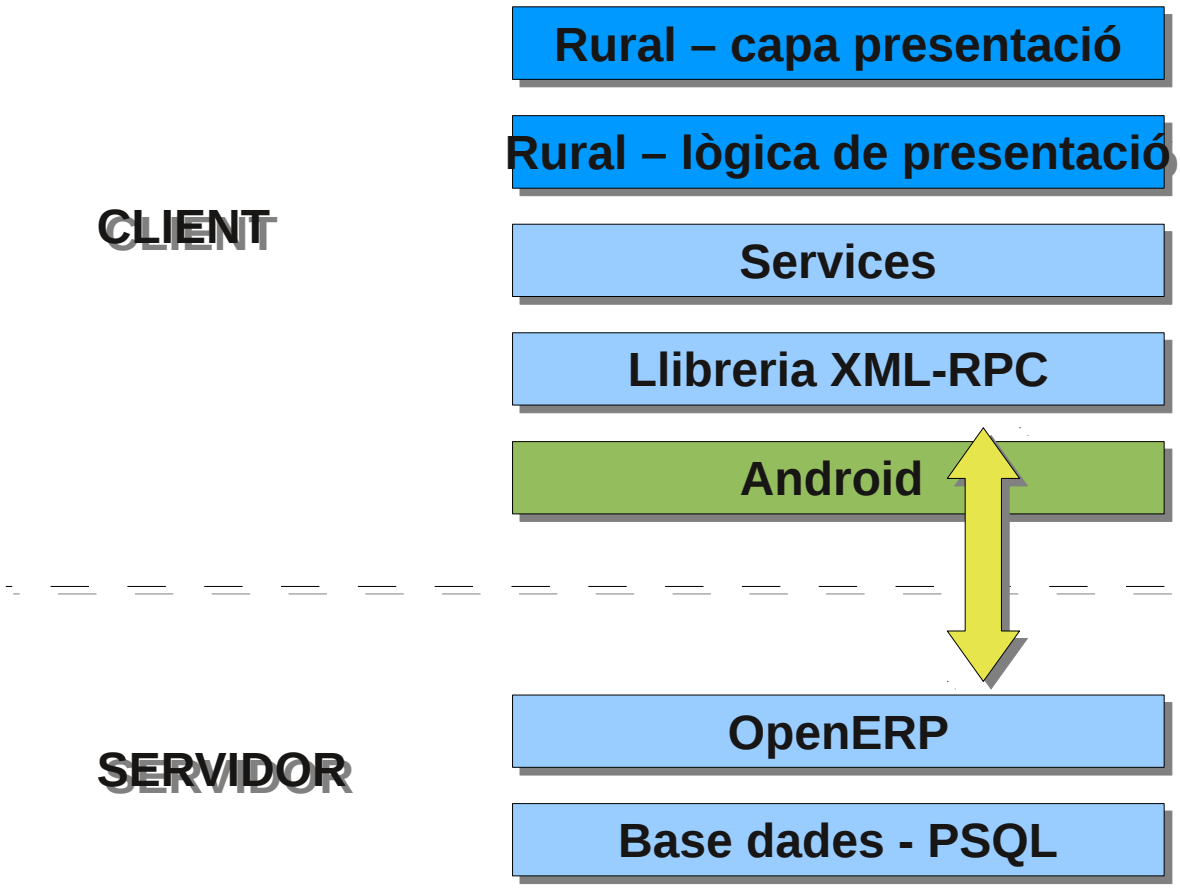
- Lectura: l'ordre **read** permet obtenir les dades d'un objecte mitjançant el seu identificador. Per exemple: obtenir les dades bàsiques de la casa si tenim el seu identificador, obtenir la descripció i disponibilitat d'un servei, etc..

En conclusió, l'aplicació Android disposarà un objecte Java que anomenarem **Services**. Aquest objecte permetrà executar les operacions: **login, search i read** per autenticar-se, cercar i llegir dades.

Aquest objecte Services és qui crea el client XML-RPC utilitzant la llibreria corresponent fa les crides corresponents al servidor OpenERP, que li retorna un resultat. Aquest objecte encapsula tota la lògica de la comunicació corresponent a la connexió i processat de l'XML de forma que el resultat que retorna l'objecte és en format de vector o taula de hash.

Per al programador resulta transparent tota aquesta comunicació no havent-se de preocupar per al processat de les comunicacions i resultat obtingut.

Es mostra a continuació una figura on s'observa de forma gràfica les capes que formen l'arquitectura d'aquesta solució:



Il·lustració 5. Arquitectura lògica per capes

En l'apartat Annex II d'aquest document es pot veure el diagrama de la base de dades relacional que conté la part del servidor.

4.2 Disseny de pantalles

En aquest apartat es mostren les diferents pantalles de les que disposarà l'aplicació i d'un petit diagrama de navegació que permet fer-se una idea de l'estructura i enllaç entre totes aquestes pantalles.

Aquest disseny s'ha realitzat de forma que funcioni com un prototipus i per tant es pugui interactuar amb l'aplicació i passar d'una pantalla a una altra tot i que les dades que s'hi vegin no siguin reals.

4.2.1 Pantalla inicial

Aquesta és la pantalla que es mostra una vegada l'usuari inicia l'aplicació. Permet accedir a la vista de mapa o bé al llistat de cases rurals.

Il·lustració 6. Pantalla inicial

4.2.2. Llistat de cases rurals

En aquesta pantalla es mostra el llistat de cases rurals. Es pot desplaçar amunt i avall en cas que el nombre de cases no es pugui mostrar tot a la pantalla. En clicar sobre una casa es passa a la fitxa d'aquesta.

Il·lustració 7. Llistat de cases rurals

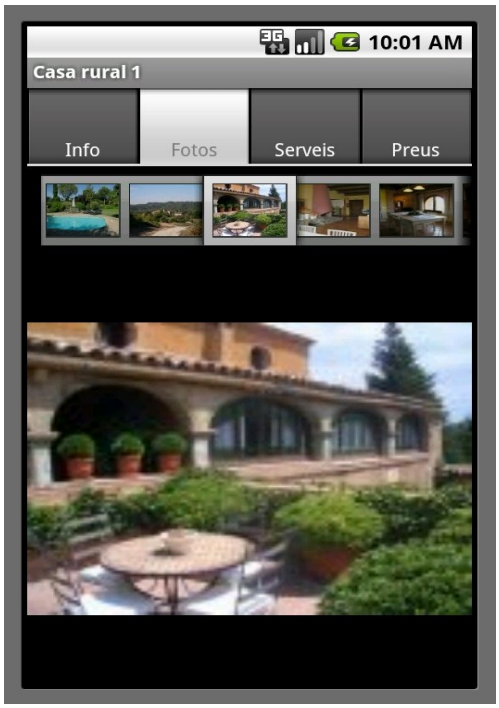
4.2.3 Fitxa de la casa

En aquesta pantalla hi ha 4 pestanyes i ens trobem a la que mostra la informació general quan hi accedim. Es poden seleccionar les altres pestanyes.

Il·lustració 8. Fitxa de la casa

4.2.4. Fotografies

En aquesta pantalla es mostra una seqüència de fotografies que l'usuari pot desplaçar a una banda i l'altra. En seleccionar-ne una es mostra ampliada al centre de la pantalla.



Il·lustració 9. Galeria fotogràfica

4.2.5 Serveis

Aquesta pantalla correspon a la tercera pestanya de la fitxa i mostra una taula amb els diferents serveis dels que disposa la casa.

Il·lustració 10. Serveis de la casa

4.2.6. Preus

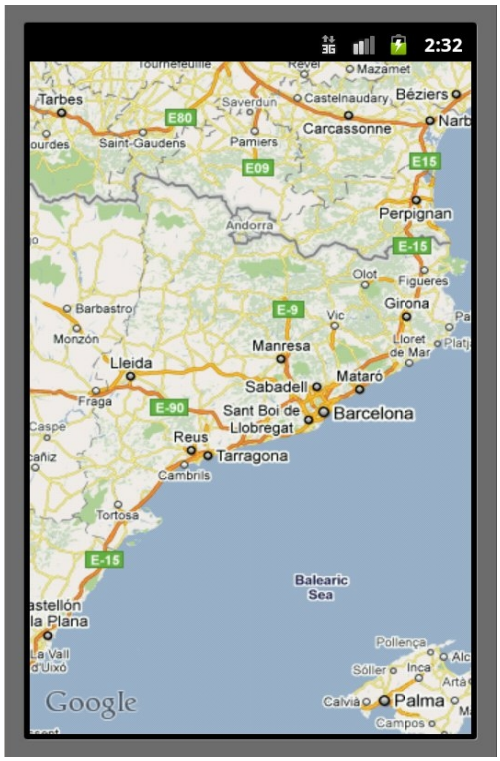
En la quarta i última pestanya accedim a la pantalla amb els preus de la casa en funció de la temporada i els dies. S'explica també les dates en que comença i acaba cada temporada.

Temporada	Alta	Mitja	Baixa
Cap Setmana	252€	252€	252€
Setmana	840€	683€	525€
Persona adicional	13€	11€	11€

Il·lustració 11. Preus de la casa

4.2.7. Mapa

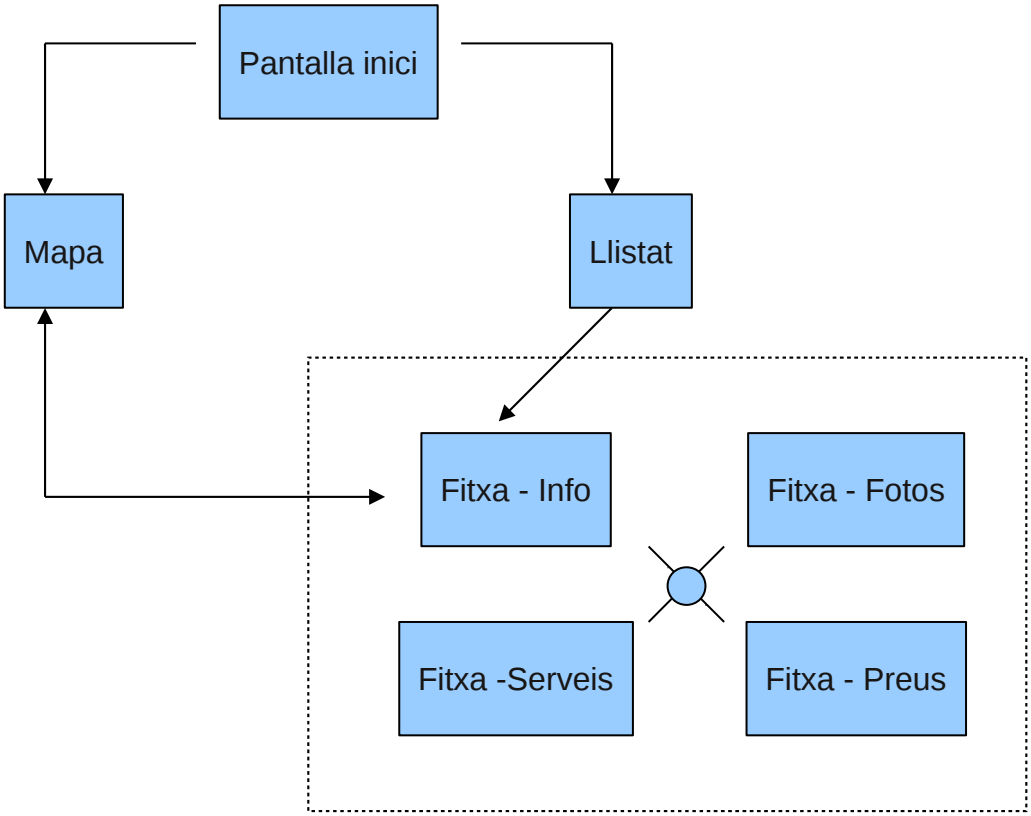
A aquesta pantalla hi accedim o bé des de la pantalla inicial clicant al botó de "Cerca al mapa" o bé des de la informació de la fitxa seleccionant "Mostra mapa".



Il·lustració 12. Mapa

4.2.8. Esquema de navegació

A continuació es mostra un diagrama de navegació de les pantalles exposades anteriorment. Les pestanyes de la fitxa es simbolitzen amb les pantalles dins d'un requadre i amb una connexió entre totes elles.



Il·lustració 13. Esquema de navegació de pantalles

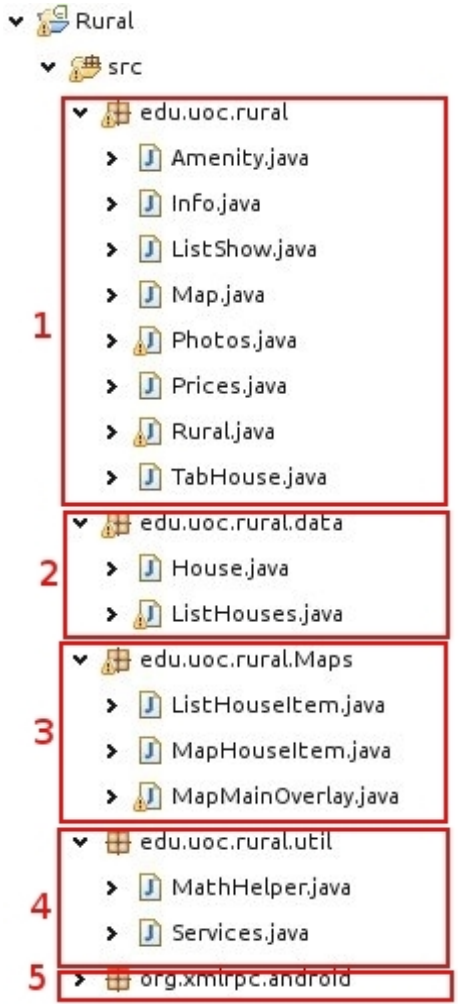
5. Implementació

En aquest apartat s'expliquen els punts més rellevants de la implementació de l'aplicació i que permeten tenir una visió de les característiques d'aquesta i del procés realitzar per a la seva implementació.

En el primer punt es fa una descripció de les diferents classes i fitxers XML de l'aplicació. En els punts posteriors s'expliquen els processos de càrrega de les dades, utilització de les funcions de trucada i correu electrònic i les parts més rellevants de la implementació del mapa.

5.1. Descripció dels fitxers i la seva organització

S'han dividit les diferents classes de l'aplicació en 5 paquets per tal de disposar d'una organització en unitats amb un mateix sentit dins l'aplicació. En la següent figura s'observa aquesta distribució pel que fa a les classes Java:



Il·lustració 14. Organització dels fitxers

El criteri per a agrupar-ho d'aquesta forma ha estat el següent:

1. **Paquet edu.uoc.rural**: cadascuna d'aquestes classes fa de controlador d'una de les pantalles de l'aplicació.

- **Amenity.java:** pestanya de la fitxa amb els serveis de la casa
- **Info.java:** pestanya de la fitxa amb la descripció, adreça i telefon de la casa
- **ListShow.java:** controlador de la pantalla que mostra el llistat de cases.
- **Map.java :** controlador que genera el mapa i gestiona l'afegit de les diferents capes.
- **Photos.java:** controlador de la galeria de fotos de la casa.
- **Prices.java :** controlador de la pestanya amb els preus i temporades.
- **Rural.java:** controlador de la pantalla inicial amb els botons de mapa i llistat.
- **TabHouse.java:** controlador que genera les 4 pestanyes a la fitxa de la casa.

2. **Paquet edu.uoc.rural.data:** conté dues classes utilitzades per guardar i tractar les dades de les cases rurals.

- **House.java:** guarda les dades corresponents a una casa quan aquestes es llegeixen de la base de dades. Es crea un objecte House per cada casa consultada.
- **ListHouses.java:** aquesta classe guarda una llista amb les cases (House). Aquesta llista és estàtica i per tant es pot obtenir des de qualsevol altre classe de l'aplicació en qualsevol moment. Té funcions per obtenir tots els ids, noms i cercar cases pel nom i pel seu id.

3. Paquet edu.uoc.rural.Maps: aquest paquet conté les classes utilitzades pel controlador del mapa (Map).

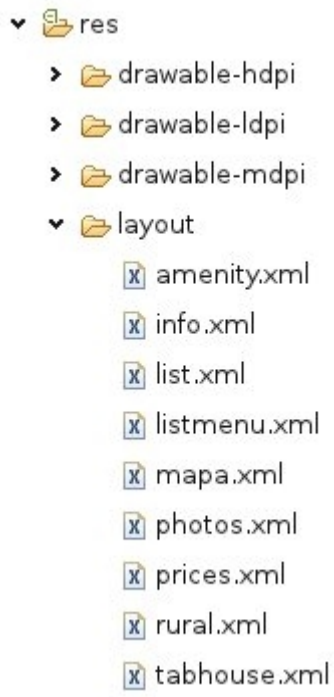
- **ListHouseItem.java:** conté i gestiona el llistat d'imatges amb la situació de les diferents cases en el mapa.
- **MapHouseItem.java:** correspon a l'ítem d'una casa i que conté l'identificador i la situació al mapa de la casa.
- **MapMainOverlay.java:** correspon a la capa del mapa que genera el cercle que redimensiona l'usuari i la que controla el posicionament geogràfic._memoria

4. **Paquet edu.uoc.rural.util:** aquest paquet conté classes que no guarden informació ni controlen cap visualització sinó que contenen funcions utilitzades en diversos punts de l'aplicació.

- **MathHelper.java:** conté una funció utilitzada per calcular la distància en metres entre dos punts del mapa expressats en coordenades de latitud i longitud.
- **Services.java:** conté les funcions necessàries per a fer consultes al servidor openerp com les funcions de login, search i read.

5. **Paquet org.xmlrpc.adrnoid:** conté les classes de la llibreria Android per a connectar amb el servidor openerp.

En la següent figura es poden veure els fitxers XML corresponents a les vistes de l'aplicació, controlades per les classes vistes anteriorment:



Il·lustració 15. Fitxers de vistes

- **Amenity.xml:** vista corresponent als serveis de la fitxa de la casa.
- **Info.xml:** vista corresponent a la pestanya Info de la fitx de la casa.
- **List.xml:** vista corresponent al llistat de cases.
- **Listmenu.xml:** vista corresponent al menú del llistat de cases.
- **Mapa.xml:** vista corresponent al mapa.
- **Photos.xml:** vista corresponent a la secció de fotos de la fitxa de la casa.
- **Prices.xml:** vista corresponent a la pestanya preus de la fitxa.
- **Rural.xml:** vista de la pantalla inicial de l'aplicació.
- **TabHouse.xml:** vista corresponent a la fitxa i que conté les 4 pestanyes.

5.2. Obtenció de les dades

El punt més important de l'aplicació és l'obtenció de les dades del servidor OpenERP. En aquest apartat s'expliquen els punts bàsics d'aquest procés. Un d'ells són les tasques asíncrones per obtenir les dades, també l'obtenció de les fotografies i finalment la classe **Services.java** que implementa l'obtenció de dades del servidor.

L'aplicació obté les dades només quan les necessita i ho fa en 4 punts diferents:

- En l'obtenció del llistat de les cases rurals: obté el llistat de noms de les cases
- Quan s'inicia la fitxa de la casa: obté les dades corresponents a la casa
- Quan s'inicia el mapa: obté el llistat amb les coordenades geogràfiques
- Quan s'accedeix a la galeria de fotos: obté les fotos mitjançant les URL's.

S'ha implementat l'obtenció de les dades de dues formes diferents. En els dos primers punts es fa mitjançant una tasca asíncrona que mostra un diàleg de progrés demanant a l'usuari que esperi. En els altres dos punts no es mostra aquest diàleg i l'obtenció de les dades és una crida a una funció i no un procés asíncron.

5.2.1. Tasca asíncrona

La tasca asíncrona per obtenir les dades que s'utilitza en els dos primers punts permet que l'usuari vegi un diàleg de progrés informant-lo que ha d'esperar.

Veiem el cas de l'obtenció del llistat de cases. El que es fa primer és crear una instància de la classe que gestiona el llistat **ListHouses** i es comprova si la llista és buida. Si la llista és buida, es crea i es crida a la tasca asíncrona **LoadListHouses.**

```
//Creating list houses
ListHouses listHouses = new ListHouses(this.getApplicationContext());

try{
//Calling to create and populate list if its empty
if(listHouses.isEmpty()){
      Services s = new Services();
      listHouses.createList(s,new Handler());
      new LoadListHouses().execute(s);
}
```

Il·lustració 16. Creació de la tasca

La tasca asíncrona **LoadListHouses** implementa un objecte del tipus **AsyncTask** que proporciona Android i sobreescriu 3 de les funcions: onPreExecute, doInBackground i onPostExecute.

La primera funció, **onPreExecute**, el que fa és iniciar el diàleg de progrés:

```
protected void onPreExecute() {
   //Creating progress dialog and showing it
   progDialog.setMessage("Carregant. Espereu si us plau ... ");
   progDialog.show();
 }
```

Il·lustració 17. Funció onPreExecute

La segona, doInBackground, obté les dades necessàries. Ho fa amb un objecte Services que s'explicarà més endavant.

```
//Main task, populates house basic info
protected Boolean doInBackground(Services... s) {
      try{
            Services serv = (Services)s[0];
            //Checking if list is empty or not
            if(listHouses.isEmpty()){
                  listHouses.populateBasicInfo(serv);
            }
     }catch(Exception e){
        Log.e(TAG,"run(): error running thread"+e.toString());
        cancel(true);
        progDialog.dismiss();
        showAlert();
        return false;
     }
     return true;
     }
```

Il·lustració 18. Funció doInBackground

Es pot observar com s'atura el diàleg de progrés i es mostra una alerta en cas de que l'obtenció sigui fallida.

En cas de no fer això en el llançament de l'excepció el diàleg es quedaria permanentment esperant i l'aplicació es penjaria.

Per últim, la funció **onPostExecute**, atura el diàleg de progrés, comprova que s'hagin obtingut les dades i genera el llistat de les cases o bé mostra una alerta si el procés ha fallat.

```
        protected void onPostExecute(Boolean b) {
        progDialog.dismiss();

        //Initializing string names vector
        String[] names;

        //Filling vector with names if list is not empty
        if(listHouses!= null && !listHouses.isEmpty()){
              names = listHouses.getNames();

              //Setting adapter
              dataAdapter = new
        ArrayAdapter<String>(ListShow.this,R.layout.list,names);

              //Showing list with house names
              ListShow.this.setListAdapter(dataAdapter);
               }else{
                     showAlert();
               }
             }
```

Il·lustració 19. Funció onPostExecute

Aquest mateix procés asíncron es realitza també per a obtenir les dades d'una casa a l'hora de veure la seva fitxa.

5.2.2. Obtenció de les fotografies

L'aplicació carrega de la base de dades OpenERP un vector on guarda totes les URL's de les fotografies d'una casa. Quan l'usuari vol veure la galeria de fotos hi ha una funció que obté les imatges convertint-les en bitmaps. Podem veure el codi d'aquesta funció a continuació:

```
    private Drawable getImageFromWeb(String url){
          Drawable drb = null;
          try{
          InputStream is = (InputStream)new URL(url).getContent();
          drb = Drawable.createFromStream(is, "src name");
          }catch(Exception e){
          Log.e(TAG,"error getting image"+e.toString());
          return null;
          }
          return drb;
    }
```

Il·lustració 20. Obtenció de fotografies

5.2.3. La classe Services.java

La classe **Services.java** és la que implementa les funcions de **login, search i read** que utilitza l'aplicació per obtenir les dades del servidor OpenERP.

Aquesta classe utilitza la llibreria XML-RPC per crear un client XML-RPC i executar les funcions que retornen les dades. A continuació es detallen les característiques d'aquestes funcions.

Login

La funció **login** obté un client XML-RPC i executa l'ordre de login contra el servidor OpenERP retornant un booleà en funció de si l'autenticació ha estat correcte o no. Es pot observar en el següent extracte del codi:

```
public boolean login() throws Exception {
      checkConnectionAndLoginData();

      //Creating XML-RPC client
      XMLRPCClient client = getClient("/xmlrpc/common");


      Object result;
      try {
            //Calling login function with authentication params
            result = (Object) client.callEx("login",
            new Object[] {m_database,m_user, m_passwd});
```

Il·lustració 21. Funció login

Search

La funció search rep com a paràmetres:

• Model: és el tipus d'objecte que volem ja que OpenERP pot guardar altres tipus a banda de les cases.

• Vector de vectors de text: això és així de complex ja que podem fer una cerca especificant diferents criteris de cerca i diferents valors per cada criteri. Per exemple: que el nom sigui un determinat i el nombre de places un altre.

Aquesta funció ens retorna un Vector de Integers amb els identificadors de les cases que compleixen amb els criteris de cerca.

Observem a continuació un extracte del codi d'aquesta funció on podem veure també la creació d'un client XML-RPC i la posterior execució de la cerca.

```
public Vector<Integer> search(String model, Vector<Vector<String>> filter)
throws Exception {
      checkModelExists(model);
      if (filter==null) throw new Exception("FILTER_NOT_SUPPLIED");

      XMLRPCClient client = getClient("/xmlrpc/object" );

      Object[] params = new Object[] {m_database,m_userId,
                                      m_passwd,model,"search",filter};

      try {
            Object[] result = (Object[]) client.callEx("execute", params);
```

Il·lustració 22. Funció search

En el cas de la nostra aplicació volem obtenir tot el llistat de cases i per tant la crida que fem té com a paràmetre un Vector buit de forma que ens retorni totes les cases. La crida que es fa a l'aplicació és la següent:

```
//Calling search with empty Vector returns all houses ids from database
houseIds = s.search("houserenting.house", new Vector<Vector<String>>());
```

Il·lustració 23. Crida a la funció search

Read

La funció read obté les dades d'un seguit d'objectes donat els identificadors d'aquests i el nom dels camps a obtenir. En l'aplicació quan l'usuari selecciona una casa, s'executa un read per obtenir-ne les dades.

Els paràmetres que rep aquesta funció són:

- Una cadena de text amb el model de l'objecte a consultar.
- Un vector amb els identificadors dels objectes a consultar.
- Un vector amb els noms dels camps a consultar.

Donats aquest paràmetres la funció retorna una taula de hash on la clau és un Integer amb l'identificador de la casa i el valor és una altra taula de hash amb el nom del camp (String) i el seu valor (Object).

Com en la funció search, és un resultat complex a causa de la intenció d'aquesta implementació de ser molt genèrica i poder ser utilitzada en molts casos diferents.

Observem a la següent figura un extracte del codi d'aquesta funció:

```
public Hashtable<Integer, Hashtable<String, Object>> read
                  (String model, Vector<Integer> ids, Vector<String> fields)
                                    throws Exception {

      //Checking model
      checkModelExists(model);

      //Checking parameters
      if (ids==null) throw new Exception("IDS_NOT_SUPPLIED");
      if (ids.size()==0) return null;
      if (fields==null) throw new Exception("FIELDS_NOT_SUPPLIED");

      //Creating XML-RPC client
      XMLRPCClient client = getClient("/xmlrpc/object");

      //Creating params
      Object[] params = new Object[] {m_database,
                        m_userId, m_passwd,model,"read",ids,fields};

      try {
            //Executing read function
            Object[] result = (Object[]) client.callEx("execute", params);
```

Il·lustració 24. Funció read

5.3. Implementació del mapa

El mapa és un altre dels punts rellevants per la seva complexitat i funcionalitats.

Les classes que intervenen en la visualització del mapa són quatre: la classe **Map.java** que és el controlador de la vista **map.xml** i les tres classes del paquet **edu.uoc.rural.Maps** que controlen les funcionalitats de posicionament i mostrat de les icones al mapa.

La funció més rellevant de la classe Map és com realitza la comunicació amb la classe MapMainOverlay que gestiona la localització, el cercle i els canvis que s'hi produeixen. Aquesta comunicació és produeix a través del llançament de missatges i un objecte de tipus **Handler** que els captura i processa.

Quan a la classe Map es crea l'objecte MapMainOverlay es fa de la següent forma:

```
   //Creating main overlay (Location management and Circle drawing)
   mainOverlay = new MapMainOverlay(this,mapOverlays,mapView,new Handler(){
         @Override
         public void handleMessage(Message m){
```

Il·lustració 25. MapMainOverlay

Això ens permet implementar la funció **handleMessage** dins la mateixa classe Map.

Aquesta funció recull dos tipus de missatges:

• L'usuari s'ha mogut i ha canviat la seva localització. En aquest punt cal actualitzar les distàncies entre les diferents cases i l'usuari i refrescar la visualització. El codi és aquest:

```
        else if (m.what == MapMainOverlay.LOCATION_CHANGED){
             try{
                //Re-calculating house distances from user geopoint
                listHouses.updateDistances(mainOverlay.getPoint());
                mapOverlays.clear();
                mapOverlays.add(mainOverlay);
                mapView.invalidate();
             }catch(Exception e){
                Log.e(TAG,"Error handeMessage(): in location changed");
             }
         }
```

Il·lustració 26. Canvi de localització

• L'usuari ha modificat la mida del cercle i es rep el missatge de que s'ha aturat el desplaçament del dit sobre la pantalla. Cal obtenir les cases que es troben dins el cercle (funció getNearHousesOverlay) i afegir totes les icones (Overlays) de les cases al mapa. El codi és el següent:

```
    if(m.what == MapMainOverlay.MOTION_STOP){
          //Adding main overlay to list overlays
          mapOverlays.clear();
          mapOverlays.add(mainOverlay);
          //Getting houses inside the circle, nearHouses
                  try{
               List<MapHouseItem> nearHouses =
                listHouses.getNearHousesOverlay(mainOverlay.getRadius());

                if(nearHouses!=null && nearHouses.size()>0){
                      for(int i=0;i<nearHouses.size();i++){
                            houseItem.addOverlay(nearHouses.get(i));
                      }
                      mapOverlays.add(houseItem);
                }
                mapView.invalidate();
          }catch(Exception e){
                Log.e(TAG,"Error handleMessage() in Motion stop:");
          }
```

Il·lustració 27. Redimensió del cercle

Per últim, destacar com s'obté la posició on es troba l'usuari. El que fa la classe MapMainOverlay és crear un LocationListener que captura els canvis en la posició de l'usuari. A continuació es mostra una part del codi que fa això:

```
        //Creating location manager
        LocationManager locationManager = (LocationManager) this.context
                    .getSystemService(Context.LOCATION_SERVICE);
        List<String> providers = locationManager.getProviders(true);

        //Creating listeners
        listeners = new LinkedList<LocationListener>();
        for (int i = 0; i < providers.size(); i++) {
        LocationListener ll = new LocationListener() {
              @Override
              public void onLocationChanged(Location location) {
                    update(location);
              }
```

Il·lustració 28. Obtenció de la posició

5.4. Trucada i enviament de correu electrònic

Unes altres funcionalitats dels dispositius mòbils que utilitza l'aplicació són les trucades i l'enviament de correu electrònic.

Quan l'usuari visualitza la fitxa de la casa pot iniciar una trucada o bé la redacció d'un correu electrònic seleccionant el telèfon o l'adreça de correu de la casa.Al fer-ho, la funció que controla l'acció de l'usuari inicia aquests processos. En la figura següent es mostra un extracte d'aquest codi:

```
    //Starting call to house phone number
      case 2:
      try{
            //Getting phone number
            String phoneUri = "tel:"+house.getPhone();
            Intent phoneIntent = new Intent(Intent.ACTION_CALL);
            phoneIntent.setData(Uri.parse(phoneUri));
            //Starting call
            startActivity(phoneIntent);
      }catch(Exception e){
            Log.e(TAG,"Error starting phone call"+e.toString());
      }
      break;

      //Sending e-mail to house mail
      case 3:
      Intent emailIntent = new Intent(android.content.Intent.ACTION_SEND);
      emailIntent.setType("plain/text");
      emailIntent.putExtra(android.content.Intent.EXTRA_EMAIL,
new String[]{house.getEmail()});
      startActivity(Intent.createChooser(emailIntent,"Enviar correu"));
      break;
```

Il·lustració 29. Trucada i correu electrònic

5.5. Procés d'implementació

Durant el procés d'implementació s'han hagut de resoldre i prendre decisions per tal de dur a terme el procés com estava especificat i en el termini previst.

Al principi la llibreria XML-RPC a utilitzar era la implementació feta per Apache (org.apache.xmlrpc). Aquesta solució va tenir problemes de compatibilitat amb Android, fet que va suposar la recerca, trobada i utilització d'una nova llibreria específica per Android.

Un altre dels punts importants ha estat la càrrega de dades. Després de comprovar que el procés de càrrega té una durada que pot fer pensar a l'usuari que l'aplicació no funciona, es va decidir implementar aquesta càrrega mostrant un diàleg de progrés informatiu.

6. Conclusions

Com a balanç a la conclusió d'aquest projecte es pot afirmar que els objectius marcats a l'inici s'han assolit quasi per complert. S'ha realitzat el producte que s'havia definit complint les diferents fites amb el temps previst.

La única funcionalitat prevista i que resta per finalitzar és poder ordenar la llistat de cases en funció de la distància o per ordre alfabètic. El codi per realitzar aquesta operació esta fet i un problema a l'hora de refrescar la llista un cop reordenada m'ha impedit resoldre aquest inconvenient en el temps previst.

Analitzant els objectius del projecte, considero que per una banda, he posat en pràctica els coneixements assolits durant els estudis ens els diferents àmbits (gestió de projectes, desenvolupament de programari, comunicacions sense fils, entre molts d'altres) i per altra banda he pogut aplicar les habilitats que s'esperen d'una enginyeria per treballar amb tecnologies que no coneixia, adquirir nous coneixements i resoldre les situacions imprevistes en un projecte d'aquest tipus.

També puc concloure molt satisfactòriament que l'aplicació desenvolupada en aquest projecte no té el seu final en l'acabament d'aquest projecte. Les possibilitats d'ampliació de l'aplicació són possibles en diferents línies: ús de la càmera per a realitat augmentada, afegir nova informació de consulta, possibilitat d'afegir comentaris per part dels usuaris, possibilitat de reservar en línia. També se'n pot millorar la velocitat de càrrega de les dades.

Aquestes possibilitats per millorar-la i afegir-hi noves funcionalitats tenen moltes probabilitats de ser realitat en un projecte proper.

A nivell personal, destaco com una gran experiència la realització del projecte per dos motius: la satisfacció de realitzar un projecte en un camp nou per mi amb il·lusió i satisfacció i sobretot pel gran esforç dedicat durant aquest període i la satisfacció personal que suposa el resultat obtingut.

7. Fonts d'informació

A continuació es detallen les fonts d'informació consultades durant la realització d'aquest projecte:

- **Android developers:** pàgina web amb informació per a desenvolupar amb el sistema Android. http://developer.android.com

- **StackOverflow:** fòrum per a programadors. http://stackoverflow.com/

- **Blog personal** amb informació sobre Android: http://www.samcoles.co.uk/mobile/

- **Blog personal** amb informació sobre Android: http://blog.pocketjourney.com/2008/03/19/

- Grup de google per a desenvolupadors d'Android: http://groups.google.com/group/android-developers/

- **Android Community:** Pàgina dedicada a la comunitat Android: http://androidcommunity.com/

- **Android Snippets:** Pàgina fòrum amb codi per Android: http://www.androidsnippets.com/

- **Wikipedia:** nformació història sobre diverses tecnologies i Android. http://es.wikipedia.org/

- **Mobiforge.com:** informació sobre el món dels dispositius mòbils. http://mobiforge.com/

- **Google code:** obtenció de la llibreria xml-rpc. http://code.google.com/intl/ca/

- **AndroidWiki:** wiki amb informació sobre desenvolupament amb Android. http://en.androidwiki.com

- **Mobile.davidods.com:** Blog personal sobre desenvolupament per iPhone i Android. http://mobile.davidocs.com/

- **Androidzteam.com:** web dedicada al desenvolupament per Android. http://androidzteam.com/

ANNEX I – Manual d'instal·lació de l'aplicació

Aquest manual explica pas a pas la configuració de l'emulador d'Android i l'execució de l'aplicació que he desenvolupat.

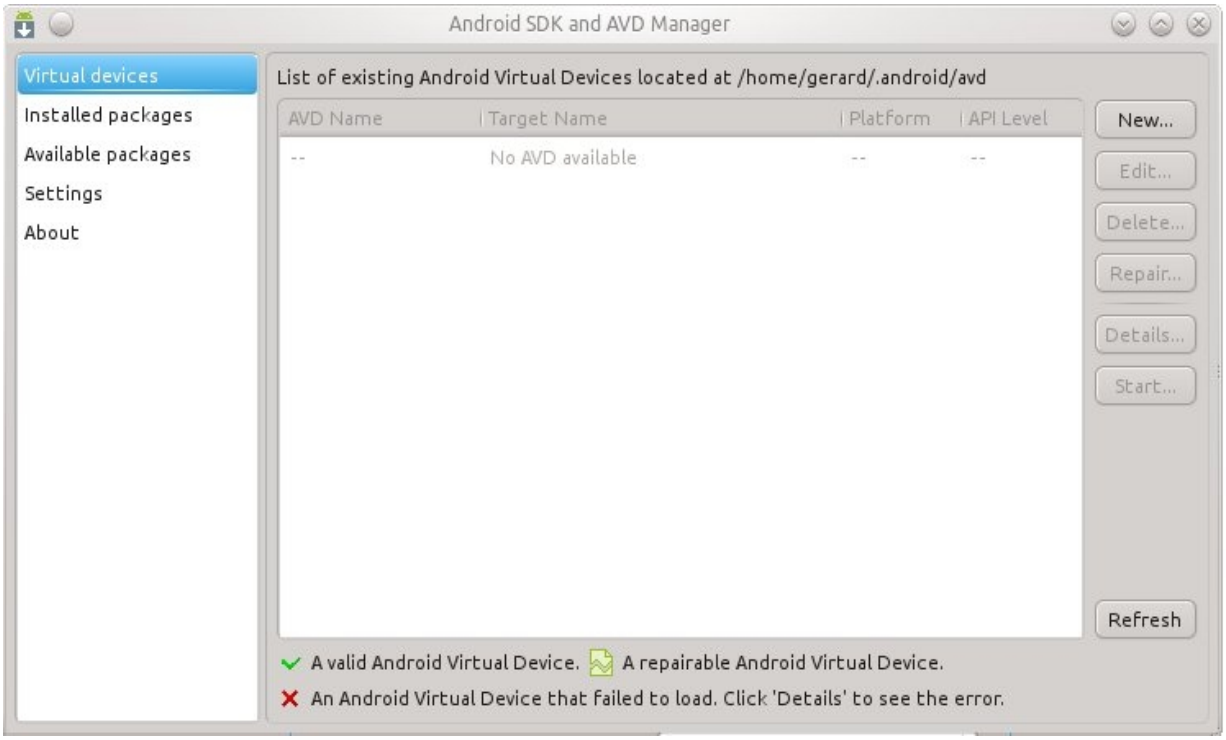
Primer de tot cal disposar de la plataforma Android SDK, descarregable aquí: http://developer.android.com/sdk/index.html

Una vegada tenim això instal·lat o descomprimit en un directori, hem de seguir els següents passos:

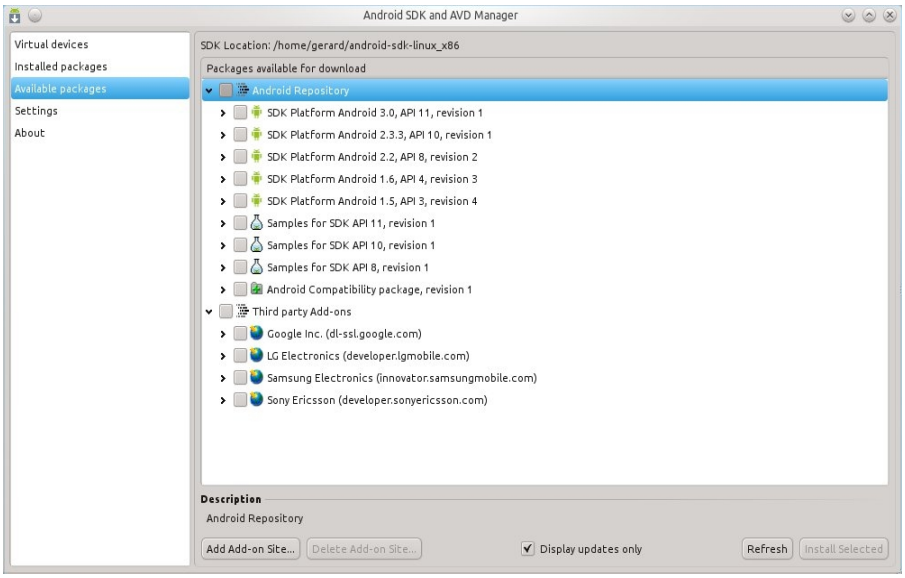
- Hem d'incorporar en el nostre PATH els directoris:
  - android-sdk-linux_x86/platform-tools/
  - android-sdk-linux_x86/tools
- Per fer-ho, executem: (modificant la ruta en funció d'on tinguem l'Android sdk)

  export PATH=$PATH:/home/gerard/android-sdk-linux_x86/tools/:/home/gerard/android-sdk-linux_x86/platform-tools/

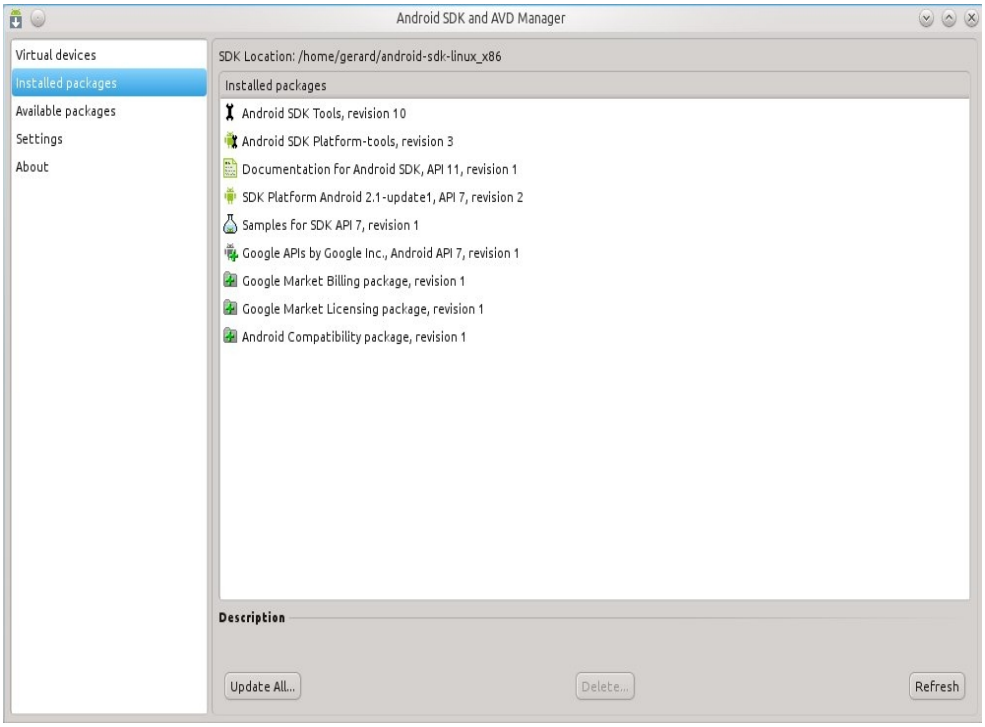
- Per iniciar l'emulador, executem la comanda: **android&** per iniciar el manager en background i veurem la pantalla següent



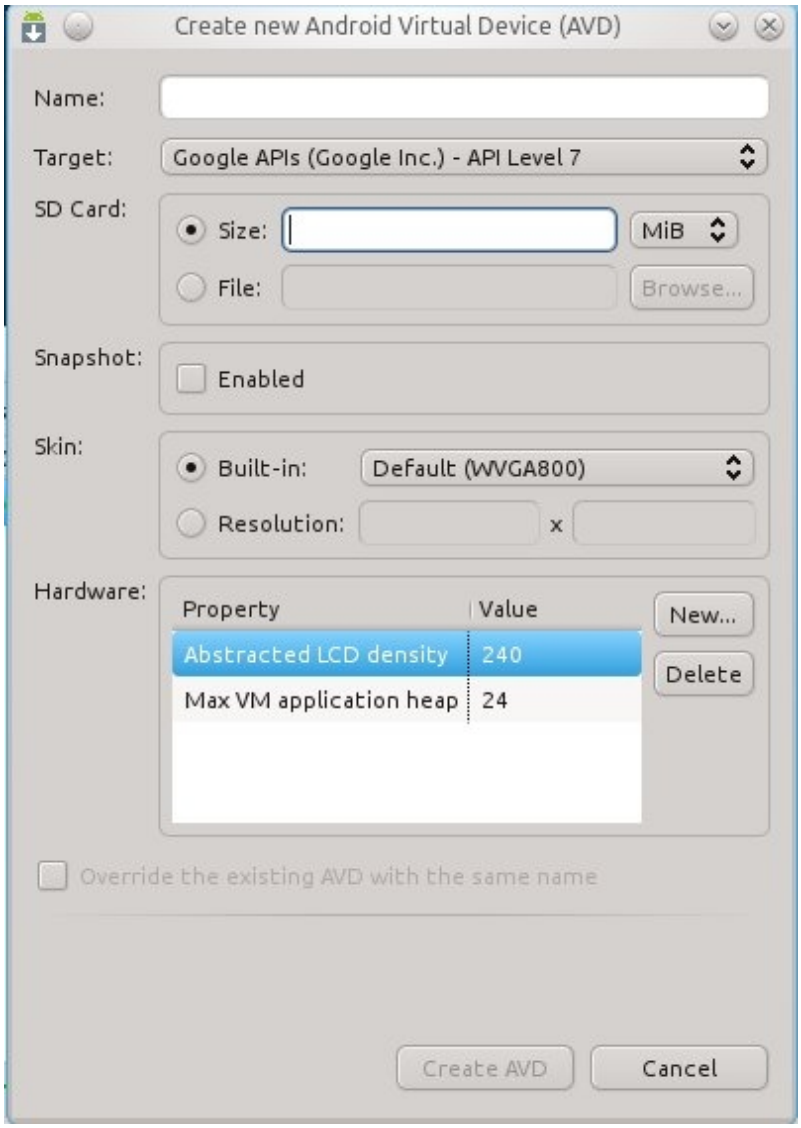
- Anem a **"Available packages"** on veurem els paquets disponibles.



- El meu projecte esta realitzat amb la versió **2.1-update1** d'Android i amb el **Google maps API7 revision 1** i per tant cal instal·lar aquests paquets.



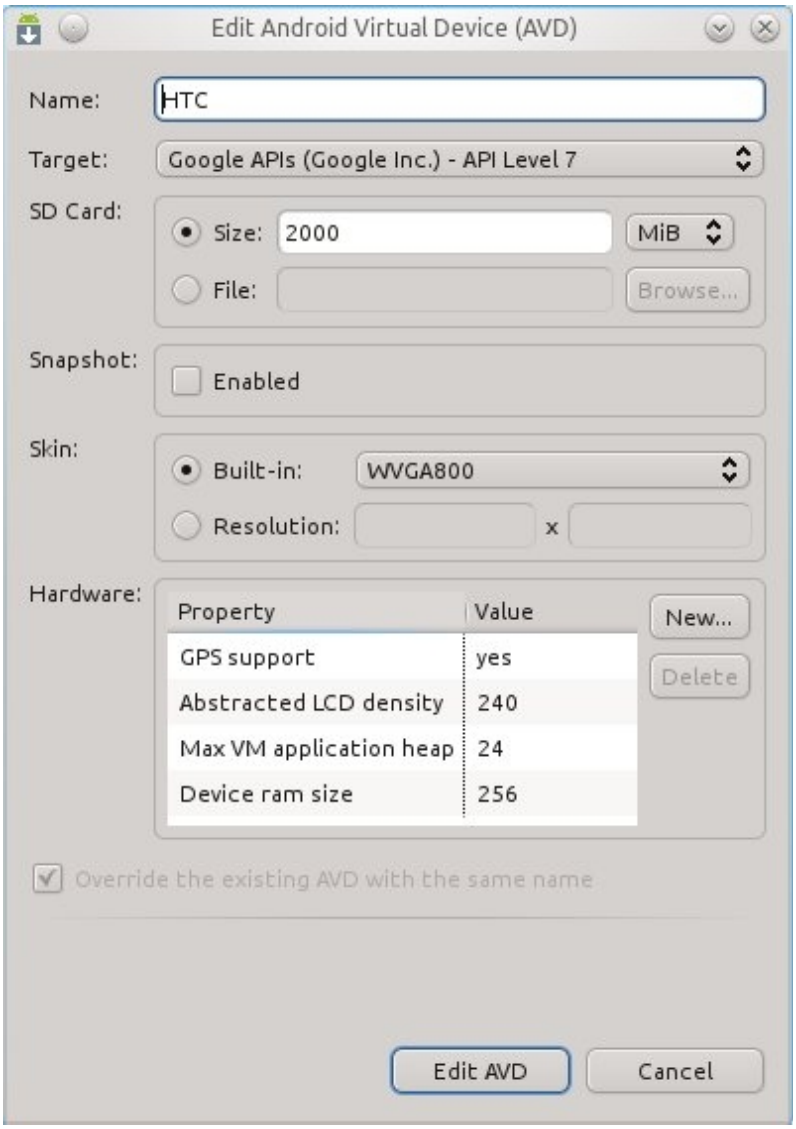
- Una vegada instal·lat, anem a **Virtual Devices** del menú esquerre i cliquem sobre el botó **New** a la part superior dreta. Se'ns obre una finestra com que es veu a continuació:



- Definim el target com a **Google Apis Level7**. En triar això ja ens afegeix a l'apartat de hardware els parèmteres de la pantalla LCD i el Max VM application Hep. Cliquem sobre **New** a l'apartat de hardware i triem:

    - **GPS Support** amb el valor **yes**
    - **Device ram size**. Clicant a la part del valor ho editem i posem 256.

- Ens queda una configuració com la següent:



Amb això ja tenim un dispositiu creat en el qual executar la nostra aplicació.

- Tornem a clicar a **Virtual Devices**, seleccionem el dispositiu que hem creat i cliquem a **Start.** S'obre una finestra on cliquem a **Launch** per tal d'engegar l'emulador.

- Esperem que arrenqui l'emulador. Des de la consola de Linux, si executem la comanda **adb devices** haurem de veure el dispositiu de l'emulador:

```
gerard@terminus:~$ adb devices
List of devices attached
emulator-5554   device
```

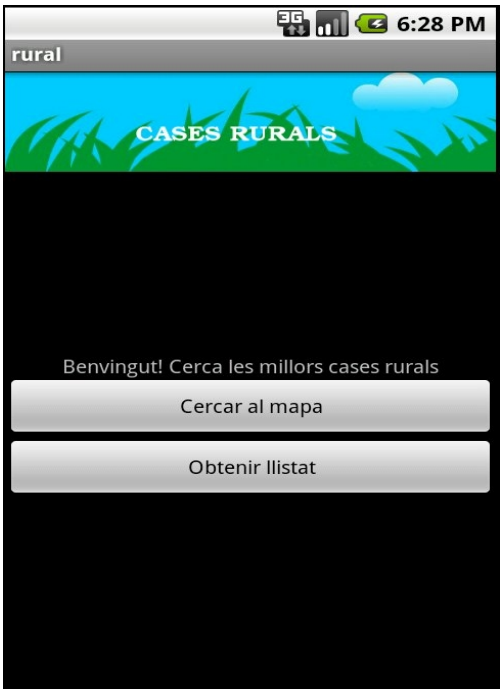
- Una vegada l'emulador esta funcionant, ja podem instal·lar-hi l'aplicació. El fitxer .apk es troba a de la carpeta **Rural/bin/Rural.apk**

- Executem la comanda:

```
adb install /home/gerard/workspace/Rural/bin/Rural.apk
1130 KB/s (226399 bytes in 0.195s)
pkg: /data/local/tmp/Rural.apk
Success
```

- Això ens ha instal·lat l'aplicació a l'emulador. Només ens queda clicar sobre la pestanya que desplega les aplicacions de l'emulador i trobarem l'aplicació Rural que podrem executar clicant-hi.

- Ja podem veure l'aplicació en funcionament.



ANNEX II – Diagrama de la base de dades OpenERP