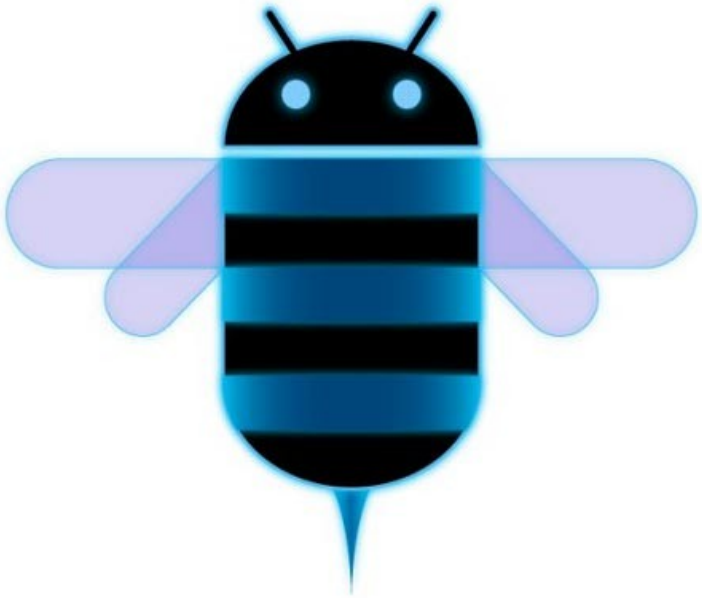


PFC- Arquitectura de Computadors i Sistemes Operatius

Títol: Estudi sistema Android 3.0 i desenvolupament d'aplicacions per a *tablets*

Professor responsable de l'assignatura: Josep Jorba Esteve
Consultor: Francesc Guim Bernat
Alumne: David Cerveró Aured

Versió del document 3.1
Data: 15 juny de 2011

Índex

Introducció

L'aparició dels *tablets*, representa una novetat per al mercat de les TIC. Aquests aparells tant sols tenen poc més d'un any de vida, si marquem com a data de naixement l'aparició de l'iPad d'Apple. Només tenim com a referència els *tablets-pc* com a antecessors dels *tablets*. Els *tablets-pc* eren PC portàtils amb una pantalla tàctil que es podia girar sobre si mateixa, de manera que el portàtil quedava plegat amb la pantalla a disposició de l'usuari. El sistema operatiu Windows oferia un programari que permetia reconèixer l'escriptura manual utiltizant un petit *pen,* com els que es fan servir a les PDA. Aquests portàtils tenien un preu força elevat, i malgrat que inicialment auguraven una segura implantació, no van triomfar. Encara avui en continuen apareixent models nous molt de tant en tant.

Els *tablets*, tot i ser més petits que els *tablets-pc*, són molt més econòmics. Han eliminat el teclat físic i han augmentat l'autonomia de la bateria. Aquests equips majoritàriament fan servir el sistema operatiu iOS per als dispositius d'Apple o el sistema operatiu Android per a la resta.

Android és un sistema operatiu de Google i es fa servir per a telèfons mòbils del tipus *smartphones* i per als *tablets*. Fa poc s'ha publicat una versió enfocada als *tablets* (la 3.0) i les expectatives són força interessants. Poc abans de finalitzar aquest TFC Google ha publicat la revisió 3.1 d'Android. La filosofia que Google hi ha aplicat és la de ser un sistema obert. Google ofereix gratuïtament un entorn de desenvolupament per crear aplicacions que puguin funcionar amb Android. També ofereix *Android Market*, una plataforma de divulgació de les aplicacions per a Android on les aplicacions es poden distribuir gratuïtament o a un preu força assequible.

Pel que fa a la relació d'Android amb el maquinari, Android es basa en un nucli en Linux per realitzar totes les transaccions amb el maquinari. Les aplicacions fan servir Java i la seva màquina virtual.

1. Motivació del projecte

La evolució de les TIC[i] és constant. Les eines tecnològiques que avui són habituals fa deu anys no existien i fins i tot algunes no estaven encara en la seva fase embrionària.
En aquests moments es poden destacar quatre elements com els *trending topics* de les TIC: les xarxes socials, la visualització en 3D, la virtualització de sistemes i el nou mercat dels *tablets*.

En el primer cas, les xarxes socials ja estan tan integrades en el nostre dia a dia, de manera que ja no es veuen com un element innovador sinó com un element habitual i necessari.
La visualització en 3D sembla que va penetrant en el mercat dels televisors domèstics i pel que fa al món dels ordinadors, van apareixent tímidament programari i targetes gràfiques que incorporen aquesta característica. Ningú sap si aquesta tecnologia (3D) acabarà imposant-se o caldrà d'esperar l'enèsima evolució per incorporar-la de forma definitiva als nostres perifèrics.
La virtualització de sistemes permet crear una capa d'abstracció entre la màquina real i el SO, de manera que es poden definir diferents sistemes fàcilment escalables i monitoritzables. Aquesta tècnica s'aplica tant a nivell d'usuari com a nivell de *hosting*[ii] o de servidors corporatius.
Pel que fa als *tablets*, Apple[iii] va llençar el 27 de gener de 2010 el ja famós iPad[iv]. Un dispositiu amb pantalla tàctil de 9'7 polzades, sense teclat, sistema operatiu iOS[v] 3.2, connexió *bluetooth* i opcionalment Wifi i 3GS.

Poc més d'un any després podem afirmar que l'iPad és un èxit. El 2 de març de 2011 se'n va presentar la revisió (iPad 2), la qual està condemnada a mantenir la línia d'èxits del seu predecessor. Aquest iPad va crear un nou mercat, el mercat dels *tablets*, elements propers als PC portàtils, però sense teclat. Bàsicament una pantalla multitàctil amb diverses possibilitats d'interconnexió, capacitat limitada d'emmagatzematge i capaços de reproduir arxius multimèdia, obrir documents i navegar per Internet. En aquest moments la gran alternativa a l'iPad és el Galaxy Tab 10.1[vi] de Samsung. Tot i que probablement la resta de marques del sector aniran desenvolupant i aportant al mercat dels *tablets* noves

unitats cada vegada més competitives amb la referencia de l'iPad.

El sistema operatiu (SO) que utilitzen els *tablets* que volen competir amb l'iPad és Android[vii]. Aquest SO basat en Linux[viii], fins ara utilitzat als *smartphones*[ix], fa el salt a un dispositiu més complex i fins i tot crea una branca nova (la 3.0 *Honeycomb*) diferenciada de la branca de telefonia (la 2.3 *Gingerbread* març 2011). Android és un dels productes de Google[x], el qual seguint la filosofia de la marca aposta per l'obertura dels seus productes i l'apropament als usuaris pel que fa a la definició i creació de programari. Així doncs, és "relativament" fàcil crear aplicacions per a Android, ja que aquesta plataforma proporciona gratuïtament entorns de programació (Android SDK[xi]). Google també facilita la divulgació de les aplicacions per Android mitjançant l'*Android Market*[xii]. Aquesta plataforma de distribució ha permès que el nombre d'aplicacions per a Android disponibles superessin la xifra de les 130.000 en finalitzar l'any 2010.

Aquest projecte vol investigar els nous dispositius (*tablets*) i quines són les seves possibilitats. Pel que fa al programari que fan servir, es vol estudiar l'opció Android com a SO i quines son les eines que ofereix per dissenyar aplicacions.

2. Descripció del projecte

2.1. Objectius del projecte

- Conèixer què és un *tablet*, què aporta i quines poden ser les seves possibles evolucions.
- Conèixer el SO Android i les diferències entre les branques 2.3 i 3.0
- Conèixer les eines disponibles per poder de dissenyar aplicacions per a *tablets* que utilitzin Android com a SO.
- Descriure una metodologia per realitzar una aplicació per a Android.
- Dissenyar un aplicació similar a l'"iPhone Battery Optimizer[xiii]" enfocada a l'ús en un *tablet* i aportar noves funcionalitats.
- Arribar a conclusions sobre els *tablets* que fan servir SO Android.

2.2 - Resultats del projecte

- L'actual document finalitzat, el qual haurà de desenvolupar els punts assenyalats als objectius del projecte.
- L'aplicació explicada a l'útim punt dels objectius.
- Una presentació que resumeixi el projecte.

3. Pla de treball

3.1. Relació d'activitats

El pla docent marca quatre grans fites: les tres PACs i l'entrega final.

FITA	DATA	CONTINGUT	
PAC 1	13 març 2011	Definició general del projecte.	13 dies = 16.25 hores
PAC 2	15 abril 2011	Documentació del projecte. Definició de l'aplicació a desenvolupar.	33 dies = 41.25 hores
PAC 3	19 maig 2011	Presentació dels resultats. Balanç del projecte. Conclusions.	29 dies = 36,25 hores
Entrega final	15 juny 2011	Entrega final del document. Entrega de la presentació.	32 dies = 40 hores
TOTAL			**133.75 hores**

A l'entrega de la primera PAC caldrà aportar les definicions inicials del projecte: les motivacions que han originat el projecte i els objectius que es volen assolir. També caldrà definir el pla de treball que marcarà el ritme.

Per a la segona PAC s'haurà de tancar la fase de documentació i redactar totes les investigacions que permetran afrontar l'assoliment dels objectius plantejats a la primera PAC. També caldrà concretar com serà l'aplicació a desenvolupar.

La tercera PAC presentarà els resultats del projecte, tant els materials (aplicació i document) com les conclusions que es derivaran d'observar el projecte en perspectivaa més de les desviacions del pla de treball i desviacions dels objectius (positives i negatives).

L'entrega final tindrà tres productes: el document del projecte revisat, l'aplicació i una presentació del projecte.

A banda de les fites que marca el pla docent, podem definir dues fases principals. La primera fase (PAC 1 i PAC 2) correspon a la fase de documentació. S'hi haurà d'investigar sobre els *tablets*, el SO Android i sobre els entorns de programació per aplicacions Android.

A la segona fase (PAC 3 i Entrega final) es realitzarà la feina de programació i posada en marxa dels coneixements adquirits a la fase de documentació, paral·lelament s'aniran definint les conclusions del projecte.

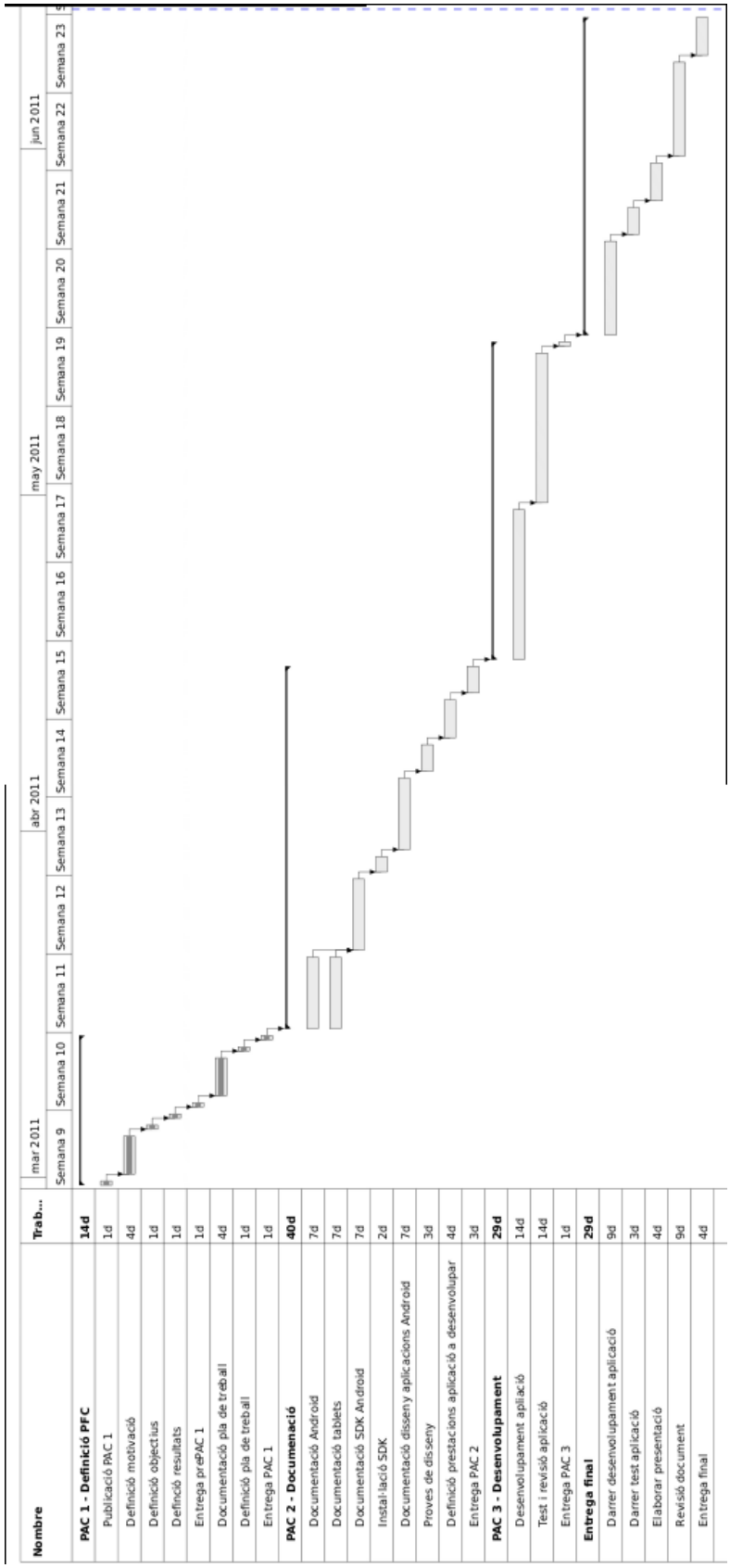
3.2. Calendari de treball

A continuació es detallen les diferents actuacions necessàries per a la finalització del projecte. Per cada activitat, es defineixen les dates inicials, la dependència, la durada i la data de finalització. Totes les activitats estan agrupades segons les quatre fites definides al punt 3.1. El projecte hauria de finalitzar el 15 de juny de 2011.

Fita		Activitat	Act. prèvia	Inici	Durada	Fi
PAC 1	1	Enunciat PAC 1		28/02/11	1	28/02/11
PAC 1	2	Definir motivació	1	01/03/11	4	04/03/11
PAC 1	3	Definir objectius	2	05/03/11	1	05/03/11
PAC 1	4	Definir resultats	3	06/03/11	1	06/03/11
PAC 1	5	Entrega prePAC1	2,3,4	07/03/11	1	07/03/11
PAC 1	6	Documentació per pla de treball	5	08/03/11	4	11/03/11
PAC 1	7	Definició Pla de treball	6	12/03/11	1	12/03/11
PAC 1	8	**Entrega PAC1**	5,6,7	13/03/11	1	13/03/11
PAC 2	9	Documentació Android	8	14/03/11	7	20/03/11
PAC 2	10	Documentació tablets	8	14/03/11	7	20/03/11
PAC 2	11	Documentació SDK Android	9	21/03/11	7	27/03/11
PAC 2	12	Instal·lació SDK	11	28/03/11	2	29/03/11
PAC 2	13	Documentació disseny aplicacions Android	12	30/03/11	7	05/04/11
PAC 2	14	Proves de disseny	13	06/04/11	3	08/04/11
PAC 2	15	Definició prestacions aplicació a desenvolupar	14	09/04/11	4	12/04/11
PAC 2	16	**Entrega PAC2**	15	13/04/11	3	15/04/11
PAC 3	17	Desenvolupament de l'aplicació	16	16/04/11	14	29/04/11
PAC 3	18	Test exhaustiu i revisió de l'aplicació	17	30/04/11	14	13/05/11
PAC 3	19	**Entrega PAC3**	18	14/05/11	6	19/05/11
E. Final	20	Darrer desenvolupament de l'aplicació	19	20/05/11	9	28/05/11
E. Final	21	Darrer test de l'aplicació	20	29/05/11	3	31/05/11
E. Final	22	Elaboració de la presentació	21	01/06/11	4	04/06/11
E. Final	23	Darrera revisió del document	22	05/06/11	9	13/06/11
E. Final	24	**Entrega final**	23	14/06/11	2	15/06/11

El calendari preveu setmanes senceres, és a dir, inclou els dissabtes i els diumenges. S'estableix que cada jornada tindrà una dedicació de dues hores. Aquesta estimació horària correspon a la mitjana, ja que és més probable que la càrrega de treball es traslladi al cap de setmana i es compensin jornades entre setmana amb dedicació inferior a la mitjana establerta. El projecte s'inicia el 28 de febrer de 2011 i finalitza el 15 de juny de 2011, en conseqüència tindrà una durada de 107 jornades. Aplicant la relació d'una hora i quinze per jornada, tindrà una dedicació de 133,75 hores (El pla docent estableix una dedicació de 9 crèdits X 15 hores = 135 hores).

A continuació el diagrama de Gantt del projecte:



4. Valoració econòmica

El programari utilitzat per a aquest projecte és de lliure distribució, en conseqüència no ha tingut cap cost. Pel que fa al maquinari, tan sols es farà servir un PC portàtil Compaq Presario CQ56, el qual es va adquirir per a un altre projecte i ja es considera amortitzat. Els únics costos seran d'energia (electricitat), connexió a Internet (ADSL) i material d'oficina (paper i estris per escriure).

Pel que fa als recursos humans, només es dedicarà una persona a la realització del projecte. De manera que s'ha estimat un cost de 50 € per hora, en concepte d'honoraris i desgast energètic, consum de comunicacions i material d'oficina.

Com que el projecte té una durada de 133,75 hores, el cost final serà de 133,75 x 50 € = 6.687,50 €.

A continuació es relaciona el programari utilitzat:

- Sistema operatiu de l'ordinador, Ubuntu 10.10
- OpenOffice 3.3.0 per a la realització del document.
- Planner 0.14.4 per a la realització del diagrama de Gantt.
- Gimp 2.7 per al tractament de imatges.
- Firefox 4.0.1 per a la navegació per Internet.
- Eclipse 3.5.2
- Java JDK 1.6.22
- Shutter 0.86.3 per realitzar les captures de pantalla.
- OpenShot 1.3.1 per la realització de vídeo resum.

5. Què és un *tablet* ?

Un *tablet* és un nou dispositiu electrònic molt jove (podem dir que va néixer el 27 de gener amb l'aparició de l'iPad d'Apple), el qual unifica característiques pròpies d'altres dispositius electrònics. Dels PC portàtils, en comparteix la capacitat de treball i dimensions (per als models de 10 polzades o més). Dels *smartphones,* la portabilitat i l'ús de la pantalla tàctil. Dels reproductors mp3 (o iPods[xiv]), la capacitat d'emmagatzemar documents multimèdia (música, imatge i vídeo) i reproduir-los. En funció de l'ús que cada usuari li acabi donant, trobarà més similituds amb un dispositiu o un altre. A continuació s'analitzaran les característiques següents:

- Pantalla
- Forma d'interacció
- Capacitat d'emmagatzematge
- Connectivitat
- Comunicació
- Autonomia
- Sistema operatiu
- Altres consideracions
- Resum i conclusions

5.1. Pantalla

L'iPad (el primer *tablet*) té una pantalla de 9.7" polzades en format 4x3 (20 cm x 15 cm) LCD retroil·luminada per LED. Seguint l'estela de l'iPad, han aparegut nombrosos *tablets* amb diferents mides de pantalla que van des de les 5" polzades del Dell Strike 5 (on desapareix la frontera entre l'smartphone i el *tablet*) fins a les 10.1" del Samsung Galaxy Tab 10.1. Com a curiositat, també es poden trobar models superiors (pel que fa a la pantalla) fins a arribar a les 14" i doble pantalla, com l'Acer *Iconia* i el *Kno*. El més habitual són les pantalles de 7" i les de 10" (incloent-hi la 9.7" de l'iPad) amb tecnologia LED, la qual té un baix consum.

Dell Streak 5 (5") – SO Android

Samsung Galaxy Tab 10.1 (10.1") – SO Android

Acer iconia (14") – SO Windows7

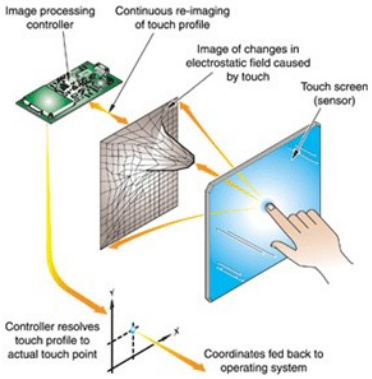
Kno (14.1") – SO Ubuntu 9.10

Actualment les pantalles multitàctils es divideixen en dues tendències, les pantalles resistives i les capacitives. Les resistives tenen diverses capes, i quan es pressiona la pantalla ja sigui amb els dits o amb un *pen stylus*[xv] es produeix un contacte entre les dues

capes. Aquesta necessitat de pressionar la pantalla pot produir la sensació de resposta lenta per part del dispositiu. Són força resistents a la pols o a l'aigua. Un petit inconvenient és que el fet d'utilitzar múltiples capes fa que la brillantor és redueixi fins a un 25%, i produeixi una sensació de foscor en entorns molt il·luminats. Les pantalles capacitives necessiten per funcionar un objecte que disposi de capacitància[xvi] excloent els *pen stylus* i els dits amb guants. En no necessitar pressionar la pantalla, la sensació per a l'usuari és de rapidesa i agilitat. El processament del senyal tàctil és una mica més complex i això fa que el seu cost sigui més elevat en comparació de les pantalles resistives.

5.2. Interacció

Els *tablets* no tenen teclat, i basen la seva interacció amb l'usuari utilitzant pantalles tàctils, on si és necessari, apareix un teclat virtual per poder introduir text. Aquestes pantalles majoritàriament són multitàctils i reconeixen simultàniament diversos punts de contacte. La tecnologia multitàctil va aparèixer l'any 2005, però no va ser fins a l'any 2007 quan Apple la va incorporar a l'iPhone i es va començar a generalitzar.



5.3. Capacitat d'emmagatzematge

La immensa majoria de *tablets* utilitzen discos durs SSD (*Solid state drive*). Aquest discos no es basen en l'ús de plats giratoris magnètics, sinó en memòria no volàtil *flash*[xvii]. Consumeixen poca energia i tenen una alta velocitat d'accés. És possible trobar-ne en versions de 256 mg, 512 mg, 2 g, 4 g, 8 g, 16 g, 32 g i fins i tot 64 g. És previsible que en el futur el mercat comenci a proporcionar discos durs SSD més grans, gairebé és una limitació econòmica, ja que quantitats superiors als 64 g no tenen un preu competitiu. A banda de la memòria interna en format SSD, es pot trobar una extensa possibilitat d'ampliació d'emmagatzematge afegint-hi targetes SD[xviii] o fins i tot en algun cas del tipus *memory stick*[xix].

5.4. Connectivitat

La connexió dels *tablets* s'acostuma a ser per USB[xx] o connector *dock* de 30 pins per als *tablets* d'Apple. Amb els ports USB, el *tablet* es pot connectar a un ordinador o se li pot afegir al *tablet* algun perifèric o unitat d'emmagatzematge extra (*pendrive*[xxi] o disc dur). A banda dels ports USB per connectar amb ordinadors, també es poden trobar dispositius amb connectors HDMI[xxii] per poder connectar una pantalla o televisor. Connectors minijack estèreo[xxiii] per poder connectar auriculars, o entrada de micròfon/línia pel que fa a les necessitats d'àudio. En alguns casos també es disposa d'un port *ethernet*[xxiv], el qual permet connectar el *tablet* a una xarxa local i així poder compartir continguts. Aquesta darrera opció no és gaire habitual, ja que es considera supèrflua perquè s'assumeix l'ús de xarxes *wifi*[xxv].

5.5. Comunicació

Com ja s'ha introduït en l'apartat anterior, els *tablets* acostumen a fer ús de les xarxes *wifi* (IEEE 802.11b, IEEE 802.11g i IEEE 802.11n) per realitzar tasques de navegació i comunicació. A banda de la connectivitat per *wifi*, també és força habitual disposar de comunicació per 3G o UMTS[xxvi], com si es tractés d'un *smartphone*. Cal remarcar que pròximament s'aniran afegint connexions per 4G[xxvii], la qual aportarà més velocitat i qualitat de servei a les actuals connexions 3G. Un altra opció de comunicació és el *bluetooth*, el qual permet connectar-se a ordinadors i altres perifèrics que puguin donar valor afegit al *tablet*. També cal destacar la comunicació GPS, que difereix de les anteriors, ja que doncs és unidireccional (en mode receptor), però proporciona al dispositiu capacitat de posicionament global, la qual combinada amb el processament de mapes dóna un valor afegit al *tablet*.

5.6. Autonomia

El fet d'utilitzar disc durs SSD en lloc dels discos durs més tradicionals amb plats giratoris magnètics ha permès alliberar el consum d'energia, deixant com a font principal de consum la pantalla. Així doncs, l'eliminació d'elements basats en moviment han reduït les necessitats energètiques dels *tablets*. Tant els *tablets* d'Apple com els dels competidors en versió Android o Windows7 asseguren una autonomia de fins a 10 hores. Altres *tablets* redueixen la capacitat de la bateria en proporció al preu dels seus dispositius. La majoria de bateries són de liti, en sintonia amb les bateries del mercat de PC portàtils.

5.7. Sistema operatiu

En un principi, els sistemes operatius utilitzats als *tablets* derivaven del mercat d'*smartphones*, però a poc a poc estan apareixent SO especialment dedicats a aquests dispositius. Al capdavant[xxviii] se situen l'iOS d'Apple (darrera versió 4.3.3 apareguda el 4 de maig de 2011) i el Honeycomb d'Android (darrera versió 3.1 apareguda el 10 de maig de 2011). Aquest dos SO comparteixen les característiques de fiabilitat i qualitat, però es poden definir com a antagonistes. El primer és força tancat, només es pot utilitzar als dispositius d'Apple i manté a Apple el control de publicació i revisió d'aplicacions. El segon es força obert, no està lligat a un dispositiu concret i proporciona eines de desenvolupament de manera lliure, de manera que implica l'usuari en la definició i revisió de les aplicacions. En un segon nivell podem trobar altres SO, com el Windows 7 de Microsoft. Es tracta d'un SO derivat de la seva versió per a PC, però afegint-hi reconeixement multitàctil, una opció força interessant si es volen executar aplicacions Windows. Una altra opció és Blacberry *tablet* OS. Igual que l'iOS d'Apple, aquest SO està lligat als dispositius que proporciona Blackberry. WebOS desenvolupat per Palm i propietat de HP és l'opció que trobarem als *tablets* de HP. Abans de tancar aquest estudi, cal remarcar que molts dispositius que utilitzen Android com a SO el fan servir en versions anteriors a la 3.0, com són la 2.1, 2.2 i 2.3. Aquests dispositius aturats en versions més primitives (dissenyades pel mercat dels *smartphones*) no han assolit la suficiència necessària per poder fer servir Android 3.0.

5.8. Altres consideracions

Altres aspectes que caracteritzen els *tablets* poden ser la incorporació d'òptiques fotogràfiques o de vídeo, per tal de realitzar fotografies o vídeos. Alguns dispositius proporcionen dues càmeres, frontal i posterior, per tal de fer servir el *tablet* en mode càmera o videoconferència. Sobre les prestacions de les òptiques, van en funció del preu, i poden arribar a nivells HD[xxix].

El pes està molt lligat al tipus de bateria i pantalla escollida en el disseny del *tablet*. Aquest pot anar des dels 300 g dels models petits (propers als *smartphones*) fins als 800 g dels models de 10" polzades. Igual que el pes, les dimensions van molt relacionades amb el tipus de pantalla i en el cas dels models de 10" polzades tenen una mida aproximada de 25 x 17 cm i un perfil que difícilment sobrepassa els 2 cm.

5.9. Resum i conclusions

En poc més d'un any el mercat dels *tablets* ha passat de la soledat de l'iPad d'Apple, (*tablet* que ha esdevingut la referència a seguir per la competència) a l'aparició de nombrosos *tablets* de la majoria de marques principals de les TIC i de marques de nova creació. Aquestes darreres han inundat el mercat dels *tablets* amb dispositius de prestacions més limitades, però amb un preu força competitiu. A mode representatiu (i gairebé anecdòtic) cal destacar l'aparició de l'*aPad*[xxx] i de l'*ePad*. Dos *tablets* "*clònics*" de l'iPad d'Apple, però que utilitzen Android. Aquesta proliferació de *tablets* ha proporcionat molta diversitat de mides, pantalles, capacitats d'emmagatzematge, tipologies de comunicació, etc.

Encara així, cal destacar que l'iPad d'Apple continua marcant la referència per a la resta de productors, tot i que el model més privatiu d'Apple a vegades fa que ignori característiques que la resta de competidors aporten de sèrie, com la possibilitat d'afegir targetes SD. Pel que fa als SO, en aquests moments iOS d'Apple i Android en les seves diferents versions acumulen la majoria del mercat dels *tablets* (s'estima que la suma dels dos SO podrien arribar al 75 % dels *tablets*[xxxi]).

6. Que és Android ?

6.1. Història

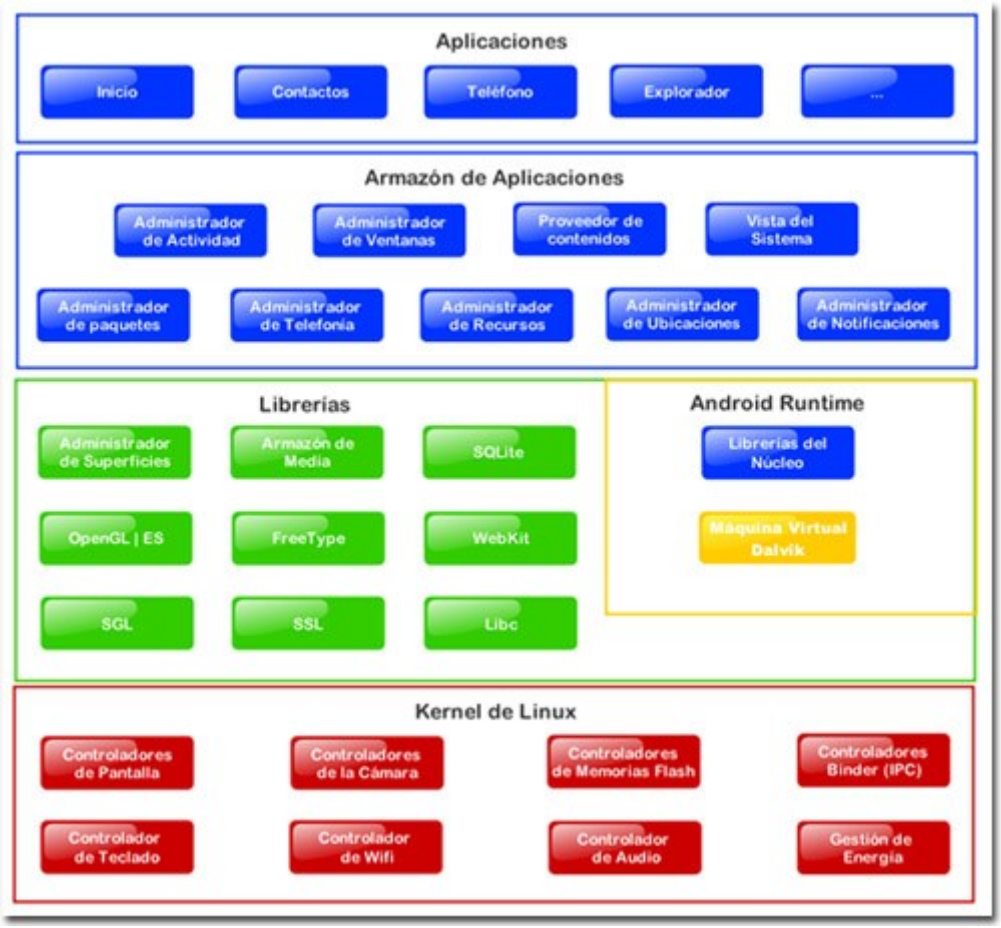
Palo Alto[xxxii] és una petita població (aprox. 60.000 habitants) de l'estat de Califòrnia (Estats Units d'Amèrica). La ciutat forma part de Silicon Valley, dins del àrea de la badia de San Francisco. Hi podem trobar la seu de Facebook, la Universitat d'Stanford i algunes oficines d'empreses tecnològiques. L'any 2005 l'empresa Android Inc, que es dedicava a desenvolupar un SO basat en Linux per a dispositius mòbils i telèfons intel·ligents, va ser comprada per Google. Andy Rubin[xxxiii], cofundador d'Android Inc, va passar a ser el director de la divisió de plataformes per a mòbils de Google. Aquesta compra va desencadenar una febre de rumors sobre la incorporació de Google al mercat de la telefonia mòbil, fins i tot es va crear el mite que Google posaria a la venda un competidor de l'iPhone, que els internautes van anomenar *GooglePhone*[xxxiv]. Finalment el *GooglePhone* s'ha materialitzat com diversos terminals de les marques HTC, Samsung, LG, Acer i Motorola. Google no ha llançat cap mòbil com a producte propi (com seria el cas de l'iPhone d'Apple), però sí presta la seva marca per a telèfons mòbils amb SO Android, els quals tenen aplicacions de Google (Gmail, Google docs, Google maps, etc.). Tornant a l'any 2005, la compra d'Android Inc i la incorporació d'Andy Rubin a Google van disparar l'evolució del SO Android. El 5 de novembre del 2007 Google anuncia la creació de l'Open Handset Alliance, una organització els objectius de la qual són la difusió de la plataforma mòbil Android. El nombre d'empreses involucrades en aquesta aliança és força gran[xxxv]. Abasta els fabricants tecnològics, les companyies de telecomunicacions i altres companyies comercials. Però la que encapçala el projecte és Google. Aquesta aliança assegura que el SO final no estarà lligat a cap marca o telefon concret gràcies al seu *kernel*[xxxvi] de Linux. Per tant, esdevé el primer SO per a dispositius mòbils obert. Cinc dies després, Google va llençar el SDK (Software Kit Development), el qual inclou un emulador d'Android, que permet provar el codi a dissenyar.

El primer telèfon mòbil amb Android va ser el G1 T-Mobile G1/HTC Dream. Es va llançar el 23 de setembre de 2008 al mercat americà i marca l'aparició de la primera versió d'Android, la 1.0. El 9 de febrer de 2009 n'apareix una revisió anomenada 1.1 que solucionava petits errors de codi. El 20 d'abril del mateix any se'n publica la versió 1.5, anomenada *Cupcake*, a partir de la qual totes anirien acompanyades d'un nom relacionat amb els dolços. Aquests noms estan ordenats alfabèticament, i s'incrementen una posició alfabètica a cada versió alliberada. La *Cupcake* aportava suport per a gravació i reproducció de vídeo, per a la càmera del dispositiu i suport per a *bluetooth* entre altres novetats. El 15 de setembre de 2009 se'n publica la versió 1.6, anomenada *Donut*. Aquesta versió va millorar l'ús d'*Android Market* i l'indicador d'ús de la bateria. Poc després, el 26 d'octubre de 2009 surt la versió 2.1, anomenada *Eclair*. En aquesta versió es van millorar les possibilitats de resolució de pantalla, es va donar suport a Microsoft Exchange i tenia zoom digital. El 20 de maig de 2010 se'n publica la versió 2.2, anomenada *Froyo*, la qual aporta integració de Flash i suport a pantalles de 4" (720p). El 6 de desembre de 2010 se'n publica la versió 2.3, anomenada *Gingerbread,* la qual implementa suport per VoIP i una renovació de la interfície d'usuari. El 2 de febrer de 2011 se'n publica la versió 3.0, anomenada *Honeycomb*. Aquesta versió d'Android només és per a *tablets* i aporta un redisseny del SO dirigit als *tablets*. Paral·lelament es publica el SDK per a la versió 3.0.

Versió	Data de publicació	*kernel* Linux
1.0	23 setembre 2008	2.6.25
1.1	9 febrer 2009	2.6.29
1.5 Cupcake	30 abril 2009	2.6.27
1.6 Donut	15 setembre 2009	2.6.29
2.0 Eclair	26 octubre 2009	2.6.29
2.2 Froyo	20 maig 2010	2.6.32
2.3 Gingerbread	6 desembre 2010	2.6.35
3.0 Honeycomb	2 febrer de 2011	2.6.36
3.1 Honeycomb	10 maig de 2011	2.6.36

6.2. Arquitectura

L'arquitectura interna d'Android està formada bàsicament per cinc components: aplicacions, *framework* d'aplicacions, llibreries, *runtimes* i *kernel*. L'esquema dels components és el següent:



A continuació s'analitza aquesta estructura des del punt de vista més intern, més proper al maquinari fins al punt més proper a l'usuari:

Kernel

Android basa el seu *kernel* en el de Linux 2.6. La darrera versió publicada d'Android (la 3.1 *Honeycomb*) utilitza el *kernel* 2.6.36. Aquest *kernel* actua com una capa d'abstracció entre el hardware i la resta del conjunt de software, a més de prestar serveis de seguretat, gestió de memòria, gestió de processos, *network stack* i *driver* model. El *kernel* de Linux es caracteritza perquè reconeix la majoria del hardware disponible. La seva constant revisió permet incorporar-hi nous elements. Qualsevol comunicació amb el hardware

passa pel *kernel*; d'aquesta manera les aplicacions i els programadors es poden despreocupar de les particularitats del hardware i poden utilitzar crides més o menys "estandarditzades" per sol·licitar l'ús del hardware.

Android *runtimes*

Android utilitza una màquina virtual de *Dalvík* per a la gestió dels processos i de la memòria. Aquesta màquina virtual està especialment dissenyada per utilitzar poca memòria i permetre diverses instàncies de la màquina virtual executant-se simultàniament. La màquina virtual està basada en registres i utilitza classes compilades per un compilador de llenguatge Java. A banda de la màquina virtual *Dalvík* en aquesta capa, Android ofereix una biblioteca de funcions base equivalents a les funcions base del llenguatge de programació Java.

Llibreries

Android inclou un conjunt de llibreries C/C++ utilitzades pels components del sistema. Entre d'altres, podem trobar: *PacketVideo*, per A la reproducció I gravació de formats d'àudio i vídeo, així com arxiu d'imatge. Superfície Manager, per a la gestió a l'accés a la pantalla 2D i gràfics 3D. *LibWebCore*, un navegador web. *FreeType*, mapa de bits i vectors de la renderització de fonts. *SQLite*, un motor de bases de dades relacionals.

***Framework* d'aplicacions**

Aquí és on treballen els programadors mitjançant l'API que proporciona Google. L'API s'ha dissenyat per simplificar la reutilització de components i qualsevol aplicació pot publicar les seves capacitats, les quals poden ser utilitzades per una altra aplicació. Hi ha definides llistes, reixes, caixes de text, botons i navegadors webs incrustables. Tot aquest entorn està totalment definit per donar facilitats al programador a l'hora de definir la seva aplicació i poder fer ús dels components disponibles.

Aplicacions

Hi ha un paquet d'aplicacions bàsiques que donen servei de SMS, calendari, plànols, navegadors, contactes, entre d'altres. A més, mitjançant l'*Android Market* es pot accedir a més de 200.000 aplicacions[xxxvii]. Les aplicacions estan escrites en Java.

6.3. Android 3.0 *Honeycomb*

Android 3.0 es publica el 2 de febrer de 2011. Fins ara els *tablets* utilitzaven el mateix SO que els *smartphones*. Google decideix desenvolupar una branca nova d'Android especialment dissenyada per al món dels *tablets* i així poder competir directament amb l'iPad. *Honeycomb* està estretament lligat a les aplicacions de Google. Accés a Gmail, accés a GoogleTalk per realitzar videoconferències, accés a Youtube, accés a GoogleBooks, accés a GoogleMaps, navegació per Chrome. Tots aquests serveis apareixen per defecte a *Honeycomb*. La interfície ha estat redefinida pensant en pantalles més grans que les dels *smartphones*. La barra superior permet accedir a les aplicacions actives. La barra inferior ens mostra notificacions i estat de bateria, *wifi* i l'hora. L'estètica reflecteix les noves dimensions de treball, nous efectes de transparències, acceleració 2D i 3D. Els *widgets* són més grans i més interactius. El reproductor multimèdia també s'adapta a les noves possibilitats i aporta una galeria de portades. Android aprofita les millors prestacions per a hardware dels *tablets* i aporta multitasca. La gestió de la càmera s'ha millorat pensant en les dimensions de la pantalla, nous efectes i la possibilitat de gestionar la càmera frontal i posterior. El teclat per pantalla també s'ha dissenyat de nou, amb més tecles i més grans.

Honeycomb és l'aposta clara de Google per conquerir el mercat de *tablets* i arribar a ser el competidor directe de l'iPad. *Honeycomb* manté un íntim lligam amb les aplicacions de Google. De fet, si l'usuari no utilitza *Honeycomb* integrant-lo amb el seu compte de Google, *Honeycomb* perd molt potencial.

6.4. Curiositats

El nom de les versions d'Android corresponen a postres. A cada nova versió el nom escollit comença per una lletra diferent seguint l'ordre alfabètic.

1.5 Cupcake	Magdalena glacejada	
1.6 Donut	Rosquilla	
2.0 Eclair		
2.2 Froyo	Iogurt gelat	
2.3 Gingerbread	Pa de gingebre	
3.0 Honeycomb	Panell de mel	

El nom d'Android i Nexus One (primer *smartphone* de Google) fan referència a la novel·la de Phillips K. Dick *Somien els androides amb ovelles cibernètiques?*, que va ser adaptada al cinema com *Blade Runner*. Tant el llibre com la pel·lícula es basen en un grup d'androides anomenats *replicants*, del model Nexus 6.

El tipus de lletra utilitzat al logo d'Android té el nom de Norad i tan sols s'utilitza al logo.

La font utilitzada al web d'Android és Droid Sans

Android Market és la plataforma de distribució d'aplicacions per a Android. En l'actualitat es poden trobar més de 200.000 aplicacions per a Android. *Android Market* retorna el 70% del preu de l'aplicació als programadors.

6.5. Resum i conclusions

Google es caracteritza per realitzar grans apostes en els seus projectes. Comparteix amb Apple la fama de crear aplicacions de gran qualitat. La diferència amb Apple és un disseny marcat per la funcionalitat respecte de l'estètica i la universalitat, ja que els seus productes no acostumen a estar monopolitzats per un tipus de hardware determinat. En aquest sentit, Android és un clar exemple. En el mercat dels *smartphones* ha triomfat indiscutiblement i la batalla entre l'iPhone i els *smatphones* Android ara es trasllada a l'iPad i la branca 3.0 d'Android. Les facilitats que proporciona Android als fabricants de hardware per crear nous *tablets* que puguin fer servir Android garanteixen una àmplia oferta de *tablets* de diverses prestacions i preus per a tota mena d'economies. A més, la distribució del SDK per dissenyar aplicacions que puguin funcionar sota SO Android i la disponibilitat de l'*Android Market* per distribuir mundialment les aplicacions que puguin aportar tant empreses de *software* com usuaris anònims contribueix al fet que el llistat d'aplicacions disponibles augmenti exponencialment.

Android és un excel·lent SO, tant per a *smartphones* com per a *tablets*. És molt funcional i està en constant evolució gràcies a les revisions que apareixen periòdicament. Està disponible per a qualsevol fabricant de maquinari que el vulgui fer servir. Tot i competir directament amb els productes d'Apple (iPhone i iPad), els perfils d'usuari de cada producte (iOS i Android) són diferents. És molt probable que cap dels dos productes acabi fent desaparèixer el competidor i a curt termini no hi ha altres alternatives amb prou potencial com per trencar aquesta polaritat en el mercat dels SO per a *smartphones* i *tablets*.

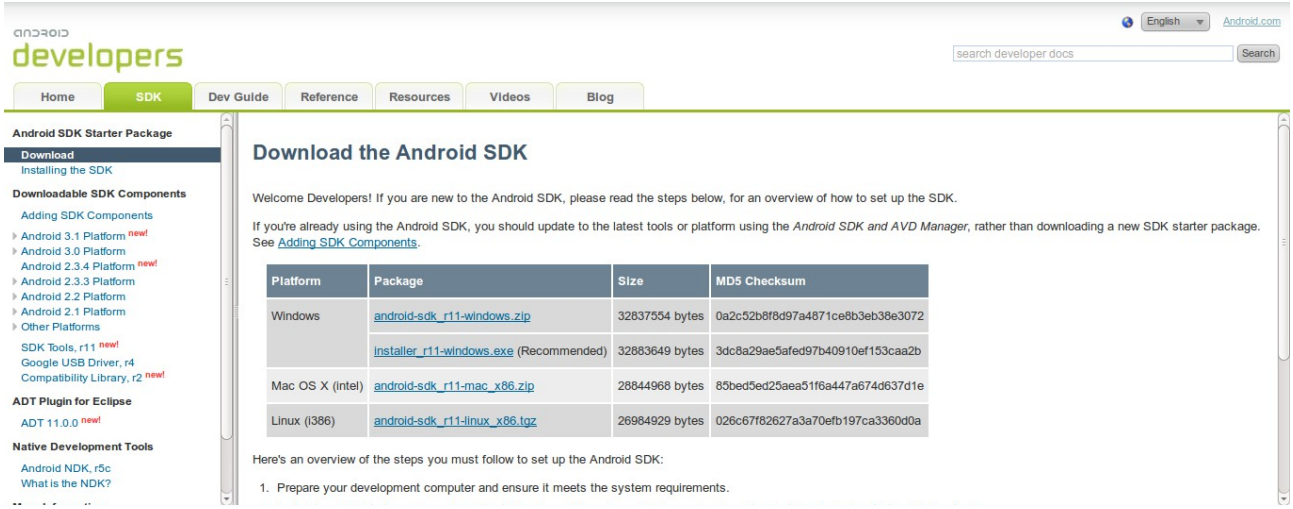
Pel que fa a la nova branca d'Android 3.0 *Honeycomb*, el seu nivell d'excel·lència es deriva de l'èxit de la branca per a *smartphones* i proporciona l'oportunitat als fabricants de generar nous *tablets* amb un gran ventall d'aplicacions disponibles. Encara és molt aviat per saber si els nous dispositius *tablets* mantindran la seva quota de mercat o no seran capaços de trobar el seu espai dins de les necessitats tecnològiques dels usuaris. Si finalment es mantenen, és segur que Android 3.0 *Honeycomb* i les seves futures evolucions continuaran presents com una de les referències.

7. Disseny aplicacions per a Android

Les aplicacions per a Android es programen en llenguatge Java. Tot el projecte s'empaqueta dins d'un fitxer amb extensió *.apk* (Android *Package*) i ja és instal·lable dins del SO Android. El projecte especifica per a quines versions d'Android és executable. Un cop instal·lat, Android fa servir una màquina virtual *Dalvík* per executar-lo. Aquesta màquina virtual és molt similar a la màquina virtual Java, però te petites diferències, ja que està optimitzada per ser utilitzada en equips petits (*smartphones*), a més, prioritza la gestió de memòria i la possibilitat de córrer paral·lelament més d'una instància de la màquina. Google ofereix a cada nova versió d'Android un SDK perquè els programadors de tot el món puguin crear noves aplicacions per a Android.

7.1. SDK

Per instal·lar el SDK d'Android, primer cal visitar el web http://developer.android.com/sdk/index.html, d'on es pot baixar la versió del SDK adient a les diferents plataformes (Linux, Mac OS i Windows).



Aquest instal·lador en realitat és un mànager d'instal·lació i possibilita escollir quines parts del SDK Android es volen instal·lar. Aquest mànager permet concretar la versió d'Android per la qual es vol dissenyar escollint un o diversos *targets* corresponents a les diferents versions publicades d'Android. També es pot escollir baixar documentació de les diferents versions i llibreries relacionades amb hardware de diferents marques. Un cop baixat i

instal·lat el SDK, és necessari instal·lar JAVA JDK per poder programar en Java. El paquet JDK de Java està disponible a

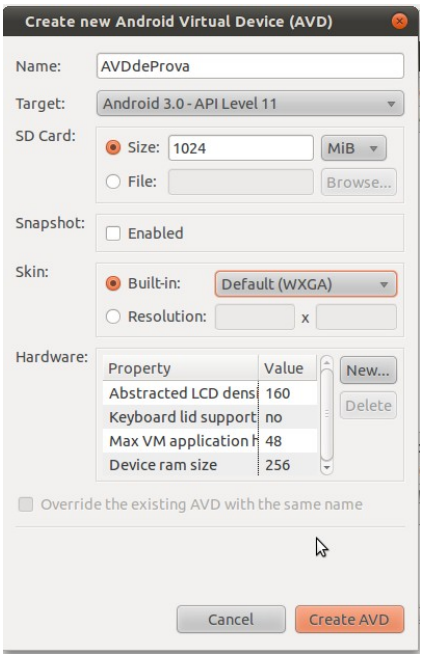
http://www.oracle.com/technetwork/java/javase/downloads/jdk6-jsp-136632.html.

Arribats a aquest punt ja es pot començar a programar aplicacions per a Android. Però cal destacar que facilita molt la feina del programador fer servir Eclipse i el plugin per a Eclipse ADT (*Android Development Tools*) que proporciona Google.

Per instal·lar Eclipse[xxxviii], primer cal visitar el web www.eclipse.org. Escollir la distribució adient per al SO de l'ordinador on s'ha d'instal·lar i iniciar el procés d'instal·lació. En els moments d'escriure aquest projecte, Eclipse té publicada la versió 3.6.x Helios. Aquesta versió encara no està gaire integrada amb les eines per programar Android i per aquest motiu es recomana fer servir la versió d'Eclipse 3.5.x Galileo.

Per instal·lar el *plugin* ADT es poden seguir les recomanacions publicades al web d'Android http://developer.android.com/sdk/eclipse-adt.html. Cal afegir el plugin a Eclipse mitjançant l'enllaç https://dl-ssl.Google.com/android/eclipse/ i posteriorment configurar-lo al menú Windows/preferences/Android d'Eclipse. Aquest plugin facilita molt la feina, ja que proporciona una eina "*drag and drop*" i accés a les propietats de tots els elements que es dissenyen.

Per poder provar i depurar les aplicacions Android sense utilitzar un dispositiu Android, es pot fer servir un emulador o dispositiu virtual AVD (*Android Virtual Device*). Aquest AVD es pot configurar amb diverses característiques de hardware (resolució de pantalla, capacitat de la SD o disponibilitat de GPS) i concretar quina versió d'Android utilitzarà. És possible definir diferents AVD per realitzar qualsevol tipus de test. Per crear els AVD utilitzant Eclipse, cal anar a l'opció Windows/Android SDK and AVD mànager i clicar a NEW.



7.2. Components d'una aplicació Android

Una aplicació Android es pot descompondre en diversos components, no tots són necessaris. Els components es poden dividir en sis grups, corresponents a la llista següent:

1. *Activity*

Representa el component principal de la interfície gràfica d'una aplicació Android. Té certa similitud amb una finestra de qualsevol altre llenguatge visual. L'*Activity* és el component on es concentren una serie de *View*s i on cal definir el comportament d'aquest *View* i de l'*Activity* en general.

2. *View*

Els objectes *View* són els components bàsics amb els quals es construeix la interfície gràfica de l'aplicació (quadres de text, botons, llistes desplegables o imatges). Android proporciona una gran quantitat de controls gràfics, però el programador pot definir nous controls. Els *View*s es troben associats a una *Activity* on es defineix la seva situació i comportament.

3. *Service*

Els serveis són components sense interfície gràfica, els quals s'executen en segon pla. Conceptualment són exactament iguals que els serveis presents a qualsevol altre SO. Poden realitzar accions concretes com actualitzar dades, llançar modificacions, o fins i tot mostrar elements visuals (*Activities*), si cal alguna interacció per part de l'usuari.

4. *Content provider*

Un *content provider* és un mecanisme que s'ha definit per compartir dades entre aplicacions. Mitjançant aquests components és possible compartir determinades dades de la nostra aplicació sense mostrar detalls sobre el seu emmagatzemament intern, estructura o implementació. Anàlogament també es pot accedir a les dades d'una altra aplicació que posi a disposició el seu *Content Provider*.

5. *Broadcast Receiver*

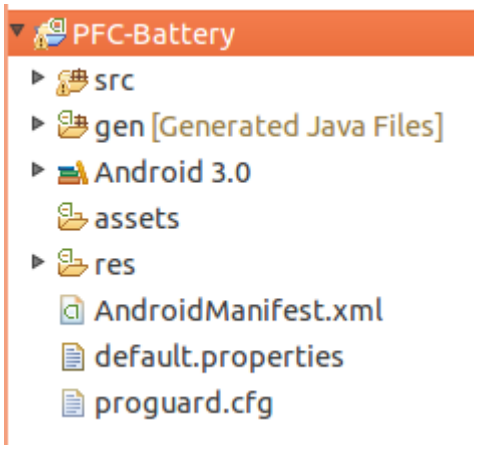
És un component destinat a detectar i reaccionar davant determinats missatges o esdeveniments globals generats pel sistema (p.e. bateria baixa, SMS rebut ...) o per altres aplicacions que generen intents per a qui vulgui escoltar-los.

6. Intent

Un *Intent* és un element bàsic de comunicació entre diversos components Android. Es pot entendre com els missatges o peticions que són enviats entre els diferents components d'una aplicació o entre diferents aplicacions. Mitjançant un *intent* es pot visualitzar una *Activity* sobre qualsevol altra, iniciar un servei o una altra aplicació.

7.3. Estructura d'un projecte Android

Una aplicació Android ha de proporcionar una serie de fitxers estructurats seguint un ordre concret. En el cas d'utilitzar Eclipse, en obrir un projecte nou, automàticament es creen aquestes estructures, algunes de buides:



Carpeta /src/

En aquesta carpeta es guarden tots els fitxers .java. És la carpeta principal de treball per al programador, ja que als fitxer .java es defineix el comportament de les *Activities*.

Carpeta /gen/

Conté una sèrie d'elements de codi generats automàticament en compilar cada projecte. Cada cop que es compila un projecte, la maquinària de compilació d'Android genera una sèrie de fitxers font de java. El més important és el fitxer R.java i la seva classe R.

Aquesta classe R contindrà en tot moment una col·lecció de constants amb els identificadors (ID) de tots els recursos que l'aplicació guarda a la carpeta /res/, de manera que l'aplicació pugui accedir fàcilment als recursos del codi.

Carpeta /Android/

En aquesta carpeta es localitzen les llibreries oficials d'Android que es faran servir en l'aplicació i que el codi java pot necessitar per fer servir les classes que el programador importa dins del seu codi.

Carpeta /assests/

Conté tota la resta de fitxers auxiliars necessaris per a l'aplicació i que acabaran inclosos al paquet final. A diferència dels elements auxiliars de la carpeta /res/raw/, els elements de la carpeta /assests/ no tenen un identificador (ID).

Carpeta /res/

Aquesta carpeta aglutina diversos elements, els quals també estan ordenats en un arbre de carpetes. En aquestes carpetes es troben tots els fitxers de recursos auxiliars necessaris per al projecte: imatges, vídeos, cadenes de text, etc. Els diferents tipus de recursos s'han de distribuir en les subcarpetes següents:

/res/drawable/ És la carpeta on es guarden totes les imatges a utilitzar.

/res/layout/ Es desen els fitxers xml on es defineixen (a nivell gràfic) les diferents pantalles de la interfície gràfica.

/res/values/ Hi han altres recursos de l'aplicació com les cadenes de text, estils i colors.

/res/raw/ Conté recursos addicionals, que no s'inclouen en la resta de carpetes.

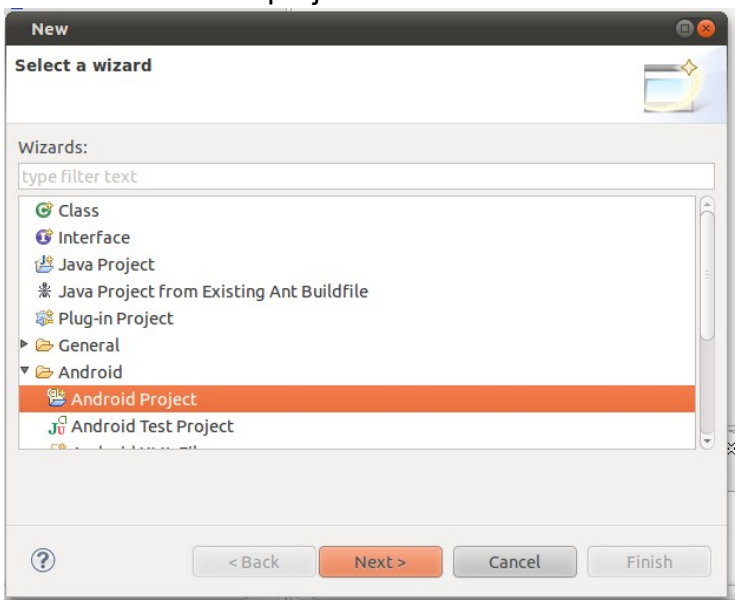
Fitxer AndroidManifest.xml

En aquest fitxer es pot trobar la definició en XML dels aspectes principals de l'aplicació, per exemple la identificació, els components o els permisos necessaris per l'execució.

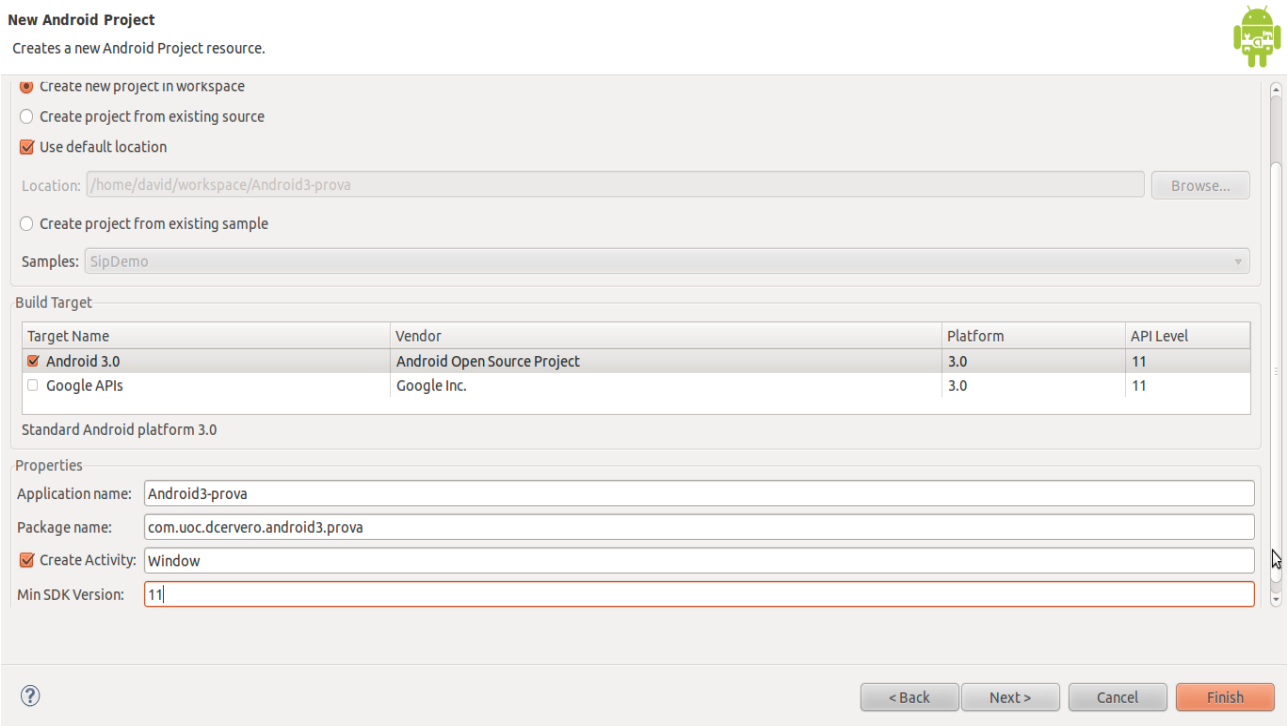
Un cop finalitzat el projecte, s'ha d'empaquetar dins d'un fitxer per poder-lo distribuir. Aquest procés es pot fer via el SDK i permet crear un únic fitxer apk que s'haurà de signar. Aquest fitxer apk es pot instal·lar en qualsevol sistema Android o pujar-lo a l'*Android Market* per a la seva distribució.

7.4. Una aplicació de prova

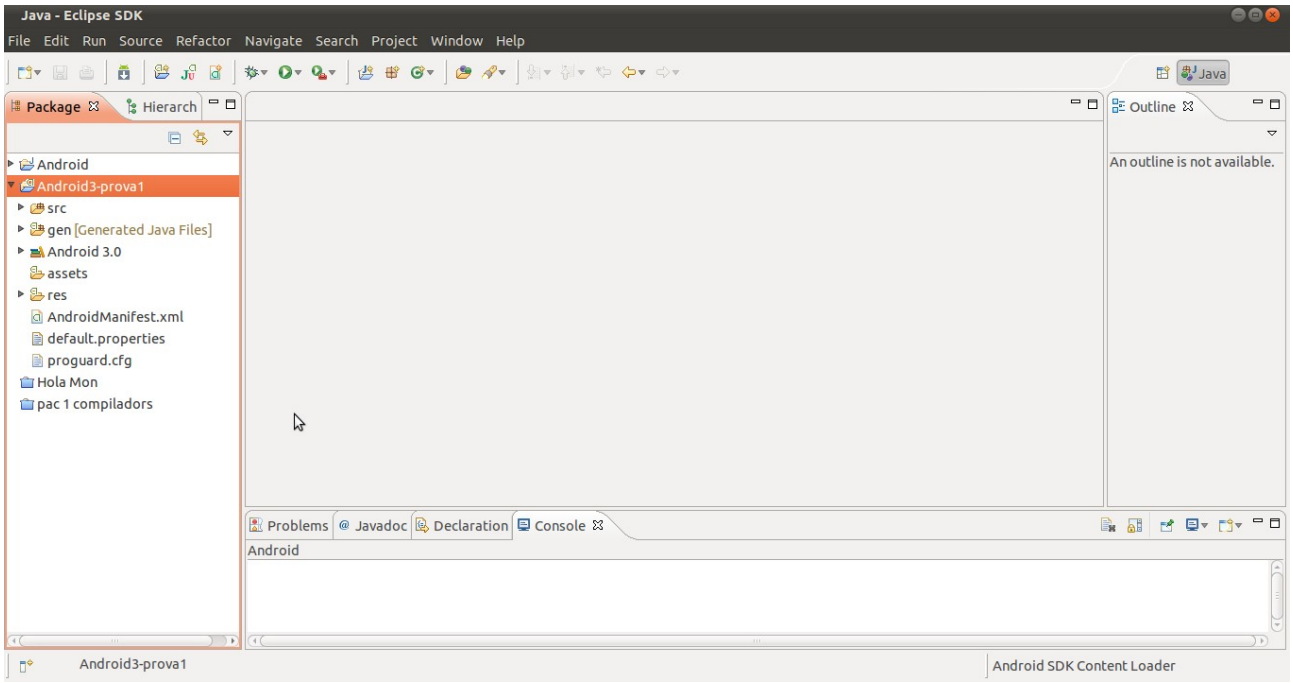
Amb tot el programari necessari instal·lat, caldrà verificar la correcta instal·lació. Això es pot fer amb la típica aplicació "Hello World". En el cas d'Android, cal obrir Eclipse i clicar a: New – Other - Android – Android project



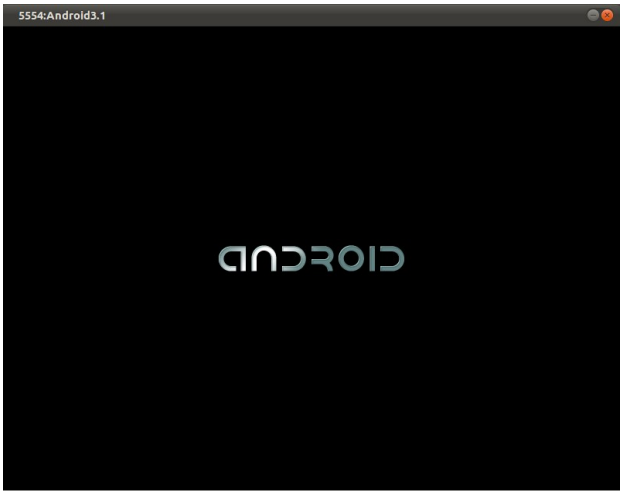
En la finestra següent s'hauran de definir els paràmetres del nou projecte, tals com el *target* Android del projecte, el nom, el *package*, *l'Activity* principal i el mínim Sdk que el podrà executar.

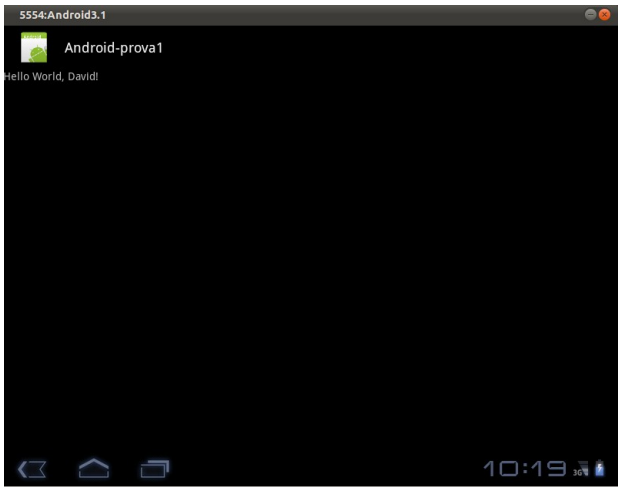


El nou projecte generat per Eclipse té tota l'estructura de fitxers necessària per poder treballar.



Arribats a aquest punt, es pot provar la primera aplicació tipus “Hello world”. Cal escollir l'opció Run/Run as Aplication, arrencarà l’AVD i s'executarà l'aplicació.





7.5. Crear una aplicació simple amb Android

En obrir un projecte Android nou a Eclipse obtindrem tots els fitxers necessaris per a la nova aplicació. Aquesta aplicació inicial correspon al ja famós i comentat “Hello World!!!”. A continuació es descriu el procés per modificar l'aplicació de manera que es mostri una pantalla amb dos botons. L'aplicació ens preguntarà el nom i el primer botó ens retornarà la salutació. El segon botó tancarà l'aplicació. Per realitzar la modificació caldrà retocar el codi dels fitxers:

/res/layout/main.xml

/res/values/../strings.xml

/src/../Windows.java

/AndroidManifest.xml

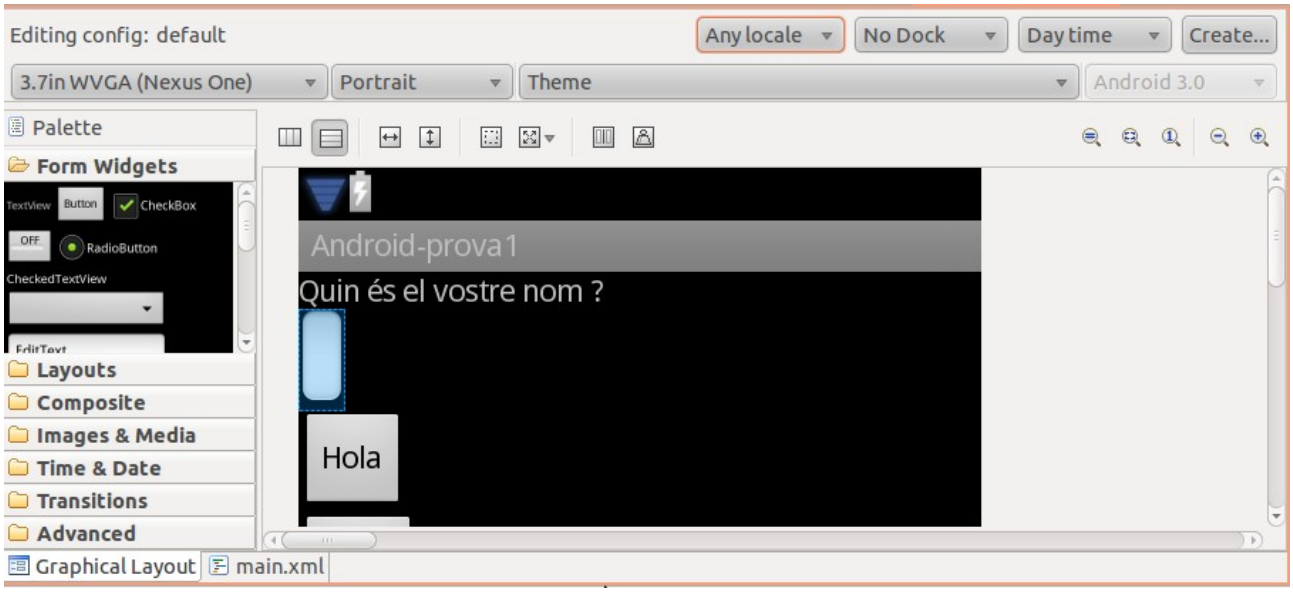
i afegir-hi els fitxers:

/res/values/.../salutacio.xml

/src/..salutacio.java

Fitxer /res/layout/main.xml

Els fitxers *layouts* es poden editar directament modificant el codi o gràcies al plugin ADT afegint gràficament (arrossegant) els elements que es volen afegir al *layout*.



Primer es defineix la disposició dels elements, que en aquest cas serà vertical, i després s'hi afegeixen: un text (Text*View*), un quadre de text (EditText) i dos botons. El codi corresponent és:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
      <TextView android:text="@string/P1_pregunta"
            android:id="@+id/text_pregunta"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
      </TextView>
      <EditText android:id="@+id/TxtNom"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
      </EditText>
      <Button android:text="@string/P1_boto_salutacio"
            android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
      </Button>
      <Button android:text="@string/P1_boto_exit"
            android:id="@+id/button2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
      </Button>
</LinearLayout>
```

Fitxer /res/values/strings.xml

Tot el text a publicar per pantalla es relaciona a les variables localitzades en aquest fitxer. El codi que utilitza és el següent:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, David!</string>
    <string name="app_name">Android-prova1</string>
    <string name="P1_pregunta">Quin Ã©s el vostre nom ?</string>
    <string name="P1_boto_salutacio">Hola</string>
    <string name="P1_boto_exit">Sortir</string>
    <string name="P2_salutacio">Sortir</string>
</resources>
```

Fitxer /src/../Windows.java

La lògica de la finestra principal es defineix en aquest fitxer. Tan sols hi ha tres controls a definir: el quadre de text, on cal introduir el nom; el primer botó, que realitzarà la salutació, i el segon botó, que tancarà l'aplicació. En el cas del primer botó, fa un intent amb la intenció que el reculli la segona activitat (salutacio.java) i així el pugui mostrar per pantalla. El codi que utilitza és el següent:

```
package com.uoc.dcervero.android.prova1;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.View.View;
import android.View.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class Windows extends Activity {
    /** Called when the Activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        final EditText TxtNom = (EditText)findViewById(R.id.TxtNom);
    /** LÃ³gica del botÃ³ de salutaciÃ³ */
        final Button button1 = (Button)findViewById(R.id.button1);
        button1.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View arg0) {
                        Intent intent = new Intent(Windows.this,
Salutacio.class);
                        Bundle bundle = new Bundle();
                        bundle.putString("NOM",
TxtNom.getText().toString());
                        intent.putExtras(bundle);

                        startActivity(intent);
                  }
          });


    /** LÃ³gica del botÃ³ de sortida */
        final Button button2 = (Button)findViewById(R.id.button2);
        button2.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View v) {
                        finish();
                  }
          });
    }
}
```

Fitxer /AndroidManifest.xml

Al fitxer AndroidManifest.xml cal declarar totes les activitats de l'aplicació. L'activitat

principal ja ve definida, però caldrà afegir-hi l'activitat corresponent a la segona pantalla, on es rep la salutació.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.uoc.dcervero.android.prova1"
      android:versionCode="1"
      android:versionName="1.0">
    <uses-sdk android:minSdkVersion="11" />

    <application android:icon="@drawable/icon"
android:label="@string/app_name">
        <Activity android:name=".Windows"
                  android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </Activity>

            <Activity android:name=".Salutacio"></Activity>

    </application>
</manifest>
```

Fitxer /res/layout/salutacio.xml

En aquesta pantalla tan sols es defineix un Text*View* corresponent a la salutació. El codi és:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"    >
      <TextView android:text="@string/P2_salutacio"
            android:id="@+id/text_salutaciÃ³"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
      </TextView>
</LinearLayout>
```

Fitxer /src/../salutacio.java

La lògica de la segona finestra s'encarrega de captar l'intent llançat pel primer botó de Windows.java i amb el resultat obtingut es modifica el text de la salutació. El codi és:

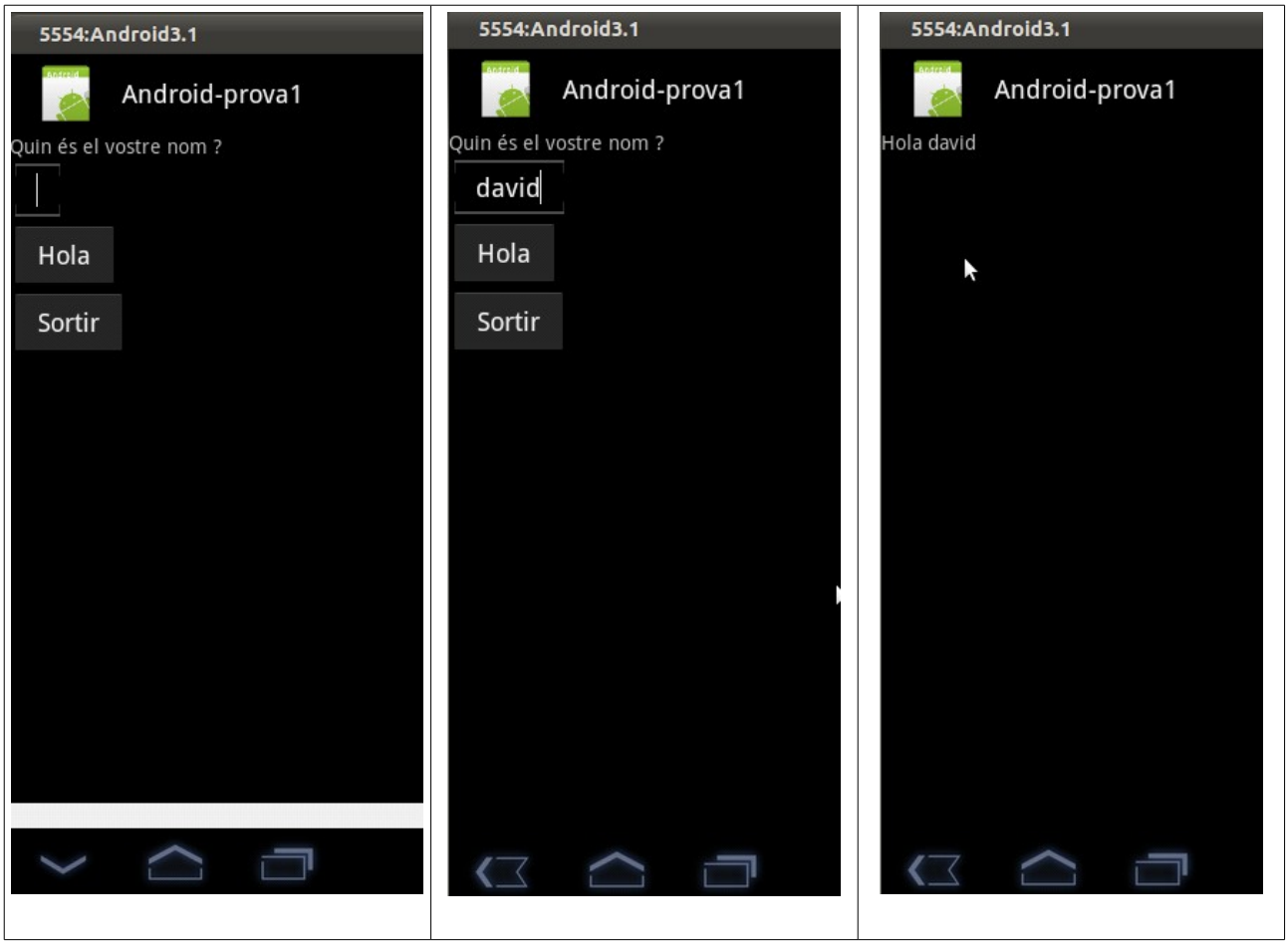
```
package com.uoc.dcervero.android.prova1;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class Salutacio extends Activity {
    /** Called when the Activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.salutacio);

        TextView text_salutacio
=(TextView)findViewById(R.id.text_salutaciÃ³);
        Bundle bundle = getIntent().getExtras();
        text_salutacio.setText("Hola "+ bundle.getString("NOM"));
    }
}
```

A continuació es mostren captures de pantalla de l'ús de l'aplicació.



8. Disseny de l'aplicació

8.1. Definició de l'aplicació

L'aplicació ha de proporcionar informació sobre la bateria. Cal que s'informi de l'estat de càrrega, si s'està carregant o descarregant, i proporcionar una estimació del temps de vida útil de la bateria en funció de l'ús que es faci del *tablet* (navegació per Internet, reproducció de música, reproducció de vídeos, etc). Aquesta informació l'haurà d'obtenir del hardware, gràcies a les llibreries que Android proporciona per treure dades de la bateria. Relacionat amb l'ús de la bateria, l'aplicació informarà de les aplicacions obertes i que estan consumint l'energia del dispositiu. A més de les aplicacions en ús, també mostrarà els processos actius, molts dels quals són propis del SO i no són susceptibles de ser tancats per alleugerir la càrrega de la memòria, processador i bateria. Per obtenir aquest llistat de aplicacions i processos oberts, també es farà ús de les llibreries que proporciona Android. L'accés a l'aplicació és farà per una icona situada a l'escriptori del dispositiu Android, de manera que l'usuari pugui fer la consulta en qualsevol moment.

Es planteja una millora de les característiques de l'aplicació que permetria tancar arbitràriament les aplicacions obertes que es poden veure al llistat proporcionat.

Totes les icones i les imatges utilitzades a l'aplicació s'han creat utilitzant el programa GIMP. Pel que fa a la icona que identifica l'aplicació, s'han fusionat tres elements, el logo de la UOC, la icona corresponent a Android 3 i una icona que simbolitza una bateria.

S'ha definit una imatge que es mostra a totes les pantalles amb el logo de la UOC, el logo d'Android i el nom de l'alumne. D'aquesta manera s'identifiquen la universitat, l'estudiant i l'objecte d'estudi del TFC.

8.2. Estructura de l'aplicació

L'aplicació es basa en tres pantalles, les quals corresponen als *layouts* main.xml, estat.xml i processos.xml.

La primera pantalla (layout main.xml) mostra el menú inicial, identifica el nom de l'aplicació i mostra les tres opcions disponibles, accés a l'estat de la bateria, accés al llistat de processos i aplicacions obertes, i finalment l'opció de tancar l'aplicació.

La segona pantalla (layout estat.xml) mostra la informació relativa a la bateria proporcionada per l'ús de la llibreria android.os.BatteryManager. Per pantalla apareix la càrrega de la bateria, la temperatura, el voltatge i s'informarà de la situació de la bateria (carregant-se, descarregant-se, desconnectada, etc.). Sota aquesta informació apareix un botó de retorn al menú principal, de manera que tanca el *layout* obert.

La tercera pantalla (layout processos.xml) mostra el llistat d'aplicacions obertes i el llistat de processos en ús. Tota aquesta informació sobre les aplicacions i el processos s'obté de la llibreria android.app.ActivityManager. Sota aquests dos llistats apareix un botó de retorn al menú principal, de manera que tanca el *layout* obert.

L'usuari tan sols pot interactuar per tancar l'aplicació i en el cas que s'implementi, l'opció de tancar aplicacions en ús.

8.3. Casos d'ús

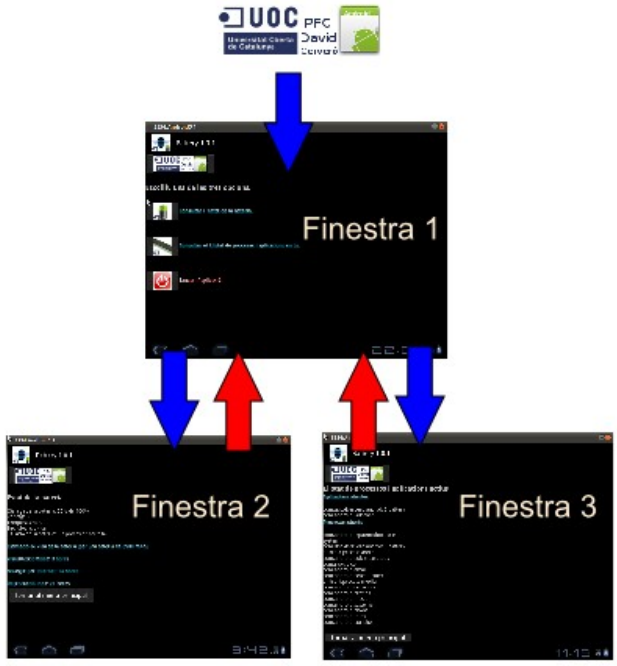
En obrir l'aplicació, es mostra el menú principal amb tres opcions:

- Accés a l'estat de la bateria
- Accés al llistat de processos
- Sortir de l'aplicació

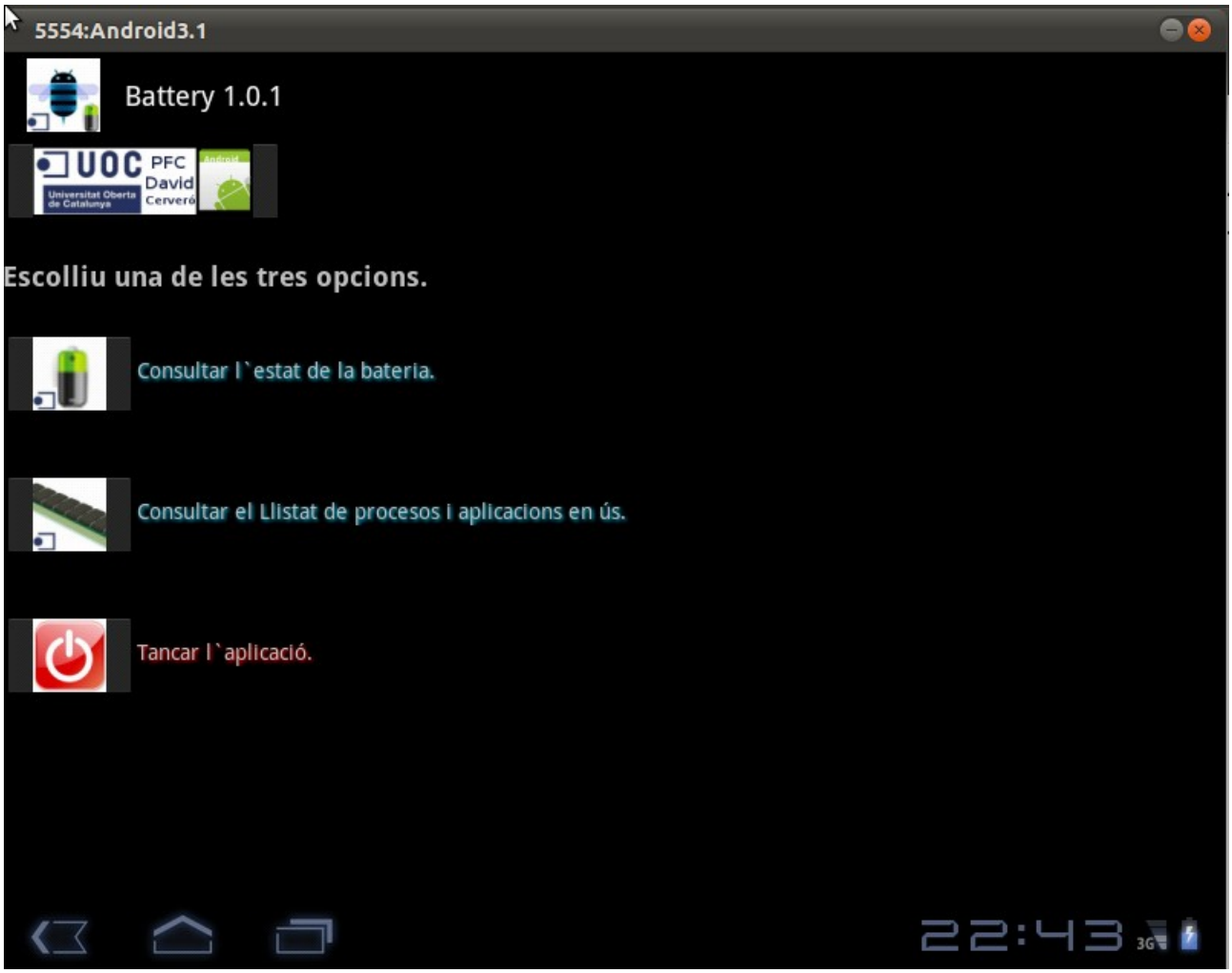
En cas d'accedir a la primera opció, es mostrarà una nova pantalla que comunicarà l'estat l'estat de la bateria. També es mostrarà l'estimació de durada (en minuts), en funció de l'ús que es faci del *tablet* (reproducció música, vídeo o consulta Internet). En tancar aquesta finestra, es torna a la pantalla principal.

En cas d'accedir al llistat de processos, es mostraran les aplicacions en ús. Són candidates de tancar-se i així economitzar el consum d'energia. En tancar aquesta finestra, es torna a la pantalla principal.

En cas d'accedir a l'opció de sortida, l'aplicació es tancarà.



8.4. Finestra principal



En aquesta pantalla es mostra la icona de l'aplicació, el nom (Battery 1.0.x) i la imatge que identifica el TFC. A continuació es mostra un missatge que indica a l'usuari que pot escollir entre una de les tres opcions: accedir a la informació de la bateria, accedir a la informació de les aplicacions i processos oberts. La darrera opció permet tancar l'aplicació. Aquesta primera pantalla s'ha definit en tres fitxers:

/src/com/uoc/dcervero/android3/battery/Window.java

/res/layout/main.xml

/res/values/strings.xml

Al fitxer main.xml s'ha definit la pantalla o *layout*. Concretament, s'han exposat els elements següents:

- ImageButton1, per mostrar la icona amb les imatges de la UOC, Android i el nom de l'alumne.

- Text*View,* que mostra per pantalla un missatge suggerint que s'esculli una opció. Aquest Text*View* s'ha definit amb un tipus de lletra més gran, per tal de captar l'atenció de l'usuari.

- Text*View,* que afegeix una línia buida.

- Tres botons. Cada un està format per dos elements ordenats de forma horitzontal. El primer element és un ImageButton (buttonEstatBateria, buttonLlistaProcessos i buttonExit) que carrega la icona corresponent a cada botó. El segon element és un Text*View* que descriu el que fa cada botó. El text dels dos primers botons s'ha formatat amb ombrejat blau i el text del tercer botó s'ha ombrejat en vermell. Aquesta codificació de colors correspon a blau accés opcions, vermell sortida de la pantalla. Android espera trobar el fitxer imatge en format png a la carpeta /res/drawable-[x]ldpi on *x* pot ser *h*, *m* o *l* en funció de la qualitat de la imatge.

Tots els textos que apareixen per pantalla es troben al fitxer /res/values/strings.xml i cada text està localitzat a una variable. Així, quan és defineix un nou Text*View*, com pot ser la primera pantalla i es vol publicar per pantalla un text, es fa referència a la variable corresponent.

Pel que fa a la lògica de la pantalla, està definida al fitxer Windows.java. Aquesta pantalla és força simple, tan sols cal definir tres accions, que corresponen als tres botons de la pantalla. Per als dos primers es defineix un intent que llança una nova Activity que fa saltar el nou *layout* corresponent a una de les dues pantalles.

L'Activity que defineix el fitxer Window.java segueix esquemàticament l'estructura següent:

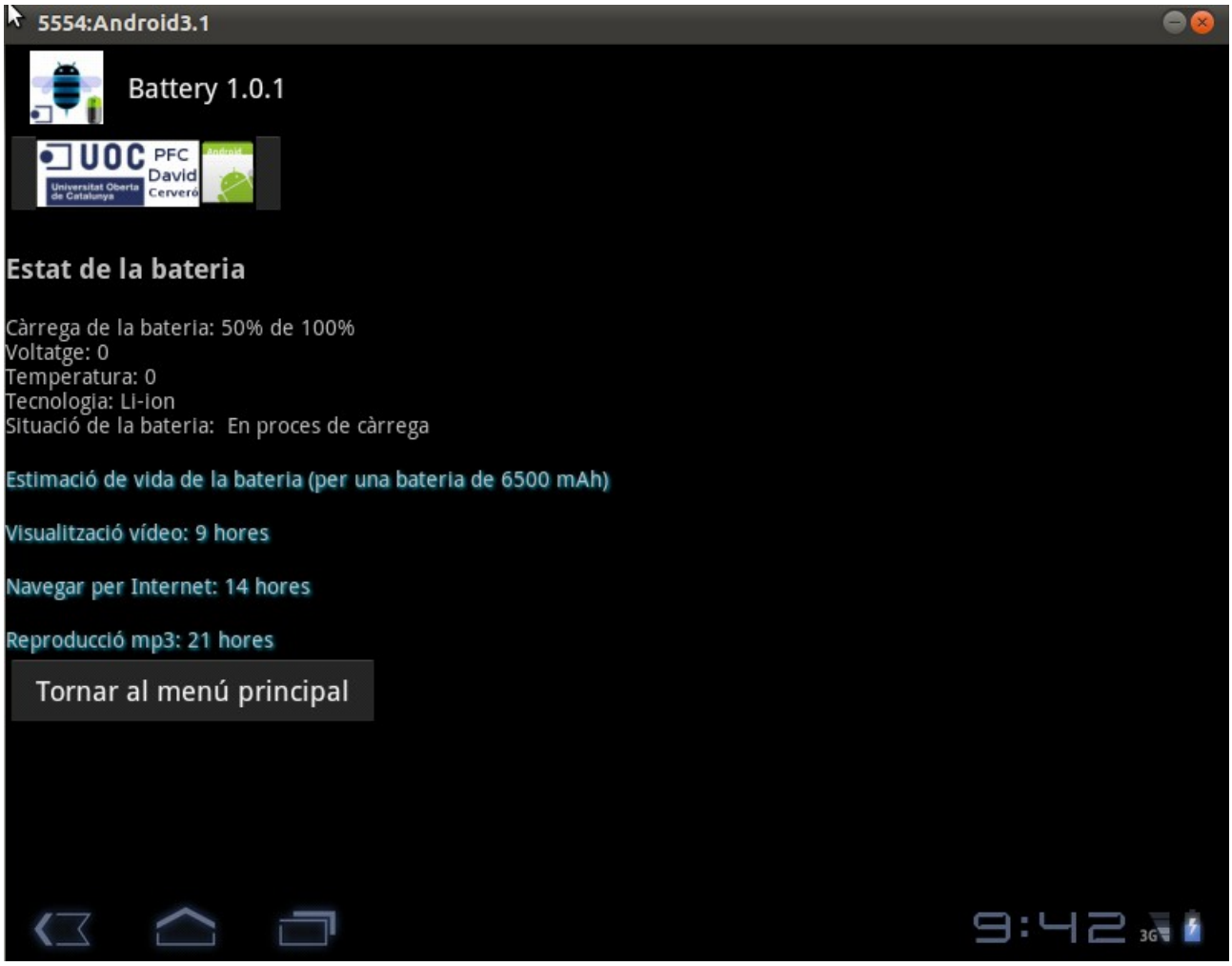
Botó"EstatBateria" - Si es clica, s'inicia l'Activity definida per Estat.class.

Botó"LlistaProcessos" - Si es clica, s'inicia l'Activity definida per Processos.class.

Botó"Exit" - Si es clica, es tanca l'aplicació.

El codi corresponent als fitxers main.xml i Window.java està explicat al punt 9 del document.

8.5. Finestra informació bateria



En aquesta segona pantalla es mostra el mateix encapçalament que a la primera (icona aplicació, nom i imatge identificativa del TFC). Després d'aquest encapçalament identificatiu de l'aplicació es mostra el títol de la pantalla “Estat de la bateria”. A continuació és mostren les dades corresponents a:

Càrrega de la bateria. Expressada com el % de càrrega.

Temperatura. En graus Celsius. Aquesta dada dóna resultat 0 en cas d'utilitzar el dispositiu virtual.

Tecnologia de la bateria. Mostra el tipus de bateria, per exemple, per al dispositiu virtual mostra que és de Liti.

Voltatge de la bateria. Mostra el voltatge de la bateria; en cas d'utilitzar el dispositiu virtual dóna 0.

Estat de la bateria. Mostra la situació de la bateria, la qual pot ser: no hi ha bateria, en

procés de càrrega, descarregant-se, en ús, carregada al 100% i estat desconegut.

Després d'aquestes dades que Android proporciona de la bateria del dispositiu, es mostra una estimació en minuts del temps de vida de la bateria en funció d'ús que l'usuari doni al *tablet*. Aquesta estimació es calcula en funció d'una mitjana estadística per a una bateria de 6500 mAh[xxxix].

Aquesta segona pantalla s'ha definit en tres fitxers:

/src/com/uoc/dcervero/android3/battery/Estat.java

/res/layout/estat.xml

/res/values/strings.xml

Tots els elements gràfics propis d'aquesta pantalla son del tipus Text*View*, excepte el botó final per tornar al menú principal. El títol s'ha formatat més gran. El Text*View* de les dades de la bateria està en lletra estàndard. Els Text*View* corresponents a l'estimació d'hores de vida de la bateria s'han formatat amb ombrejat blau. Les variables (string.xml) que alimenten aquest Text*View* es recuperen al fitxer Estat.java i es reomplen amb les dades obtingudes de l'ús de la llibreria BatteryManager i amb valors calculats per estimar la vida de la bateria. La resta d'elements gràfics són comuns a la pantalla principal (icona i imatge identificativa del TFC) o a la tercera pantalla (botó de tornar al menú inicial).

Tots els textos que apareixen per pantalla es troben al fitxer /res/values/strings.xml i cada un està localitzat a una variable.

Pel que fa a la lògica de la pantalla, està definida al fitxer Estat.java. S'hi llança un broadcastReceiver que recull una serie d'informació relativa a la bateria proporcionada per l'ús de la llibreria android.os.BatteryManager. Tota aquesta informació s'ubica en diferents variables numèriques i en alguna del tipus String. Un cop rebuda la informació, es recupera una de les variables que alimenten el Text*View,* que es mostrarà per pantalla, i es va afegint a aquesta variable els valors rebuts i *strings* per ajudar a interpretar les dades. Un cop s'hi han afegit totes les dades, es retorna aquesta variable per tal que es mostri per pantalla.

Inicialment es volia proporcionar l'estimació en minuts del temps que queda de bateria. No ha sigut possible implementar aquesta dada, ja que les llibreries només permeten recuperar l'estimació de càrrega restant de la bateria. Per poder proporcionar aquesta estimació, caldria poder obtenir una d'aquestes dues dades:

- El consum individual de cada aplicació, el qual permetria realitzar un test de les aplicacions i després extrapolar-lo en minuts.

- Els miliampers per hora de la bateria.

Com que no és possible obtindre aquestes dues dades per Android, s'han escollit dades estadístiques per realitzar els càlculs. Pel que fa als mAh, tot i que hi ha molta diversitat de bateries que van des de les petites com la del Konia Tab A100 amb 1530 mAh fins a d'altres de més potents com es el cas de la bateria del *tablet* Eee Pad Slider amb 8300 mAh, hi ha una majoria de *tablets* que fan servir bateries sobre els 6500 mAh. Per aquest motiu els càlculs es fan sobre una bateria de 6500 mAh. A l'annex B es mostra una llista de *tablets* on els fabricants publiquen les dades de la bateria. Aquest llistat ha sigut un dels models per escollir la xifra de 6500 mAh com a dada mitjana de les bateries i aplicar-la als càlculs d'estimació de minuts de vida de la bateria.

A banda de les característiques de la bateria, també cal saber el consum d'energia que fan les aplicacions. Android tampoc proporciona aquesta informació i, en conseqüència, s'ha estimat el consum segons la informació que Google va proporcionar la setmana de I /O *confenrence* l'any 2009 (vegeu Annex A i B), que organitza anualment.

Taula resum del consum de mAh per les aplicacions	
Visualitzar vídeo Youtube	340 mA
Navegar per Internet	225 mA
Ús normal (mitjana)	42 mA
Reproducció àudio mp3	150 mA
Joc utilitzant els sensors	500 mA

Amb aquestes dues dades (mAh i consum d'aplicacions) i l'estat de càrrega de la bateria, es pot calcular el temps de vida expressat en hores. La fórmula utilitzada és la següent:

Hores de bateria = (mAh de la bateria / consum de l'aplicació) / (estat de càrrega)

Per exemple, per calcular el temps de vida d'una bateria de 6500 mAh al 42 % de càrrega, si es vol utilitzar per escoltar mp3, cal fer el càlcul següent:

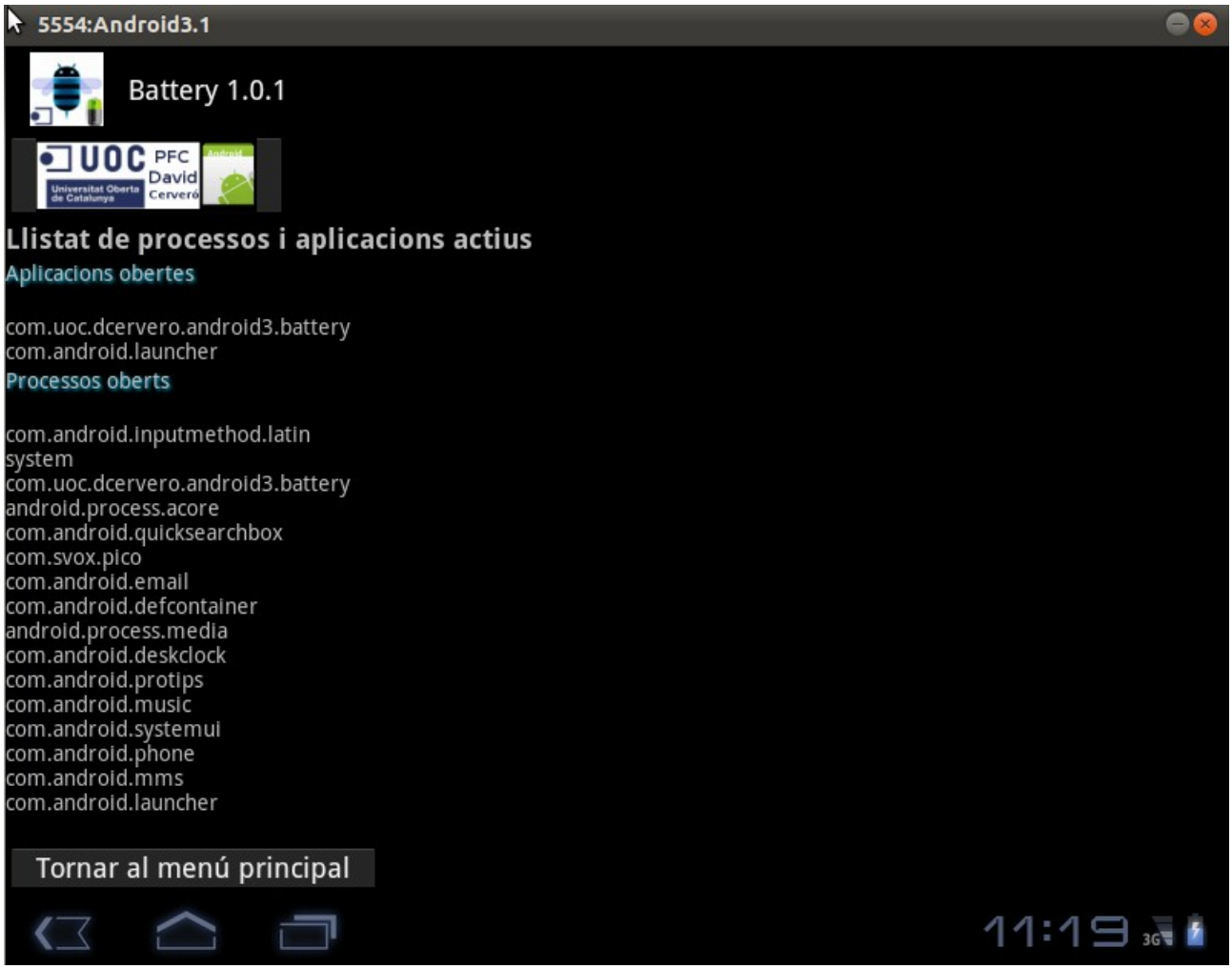
Hores de bateria = (6500 mAh / 150) / (100/42) = 43,3 hores / 2,38 = 18,2 hores

Pel que fa a la lògica de la pantalla, està definida al fitxer Estat.java. Inicialment el codi llança un broadcastReceiver amb la intenció de captar la informació provinent de la bateria. Es recull una sèrie d'informacions (càrrega de la bateria, escala de càrrega, voltatge, temperatura, tipus de bateria i estat). Totes aquestes dades es transformen a format text, si es el cas s'utilitzen per realitzar càlculs i finalment, es concatenen amb les variables de tipus text localitzades a strings.xml que acaben mostrant-se per pantalla.

El tractament que s'ha fet de les dades obtingudes per la llibreria BatteryManager es limita als càlculs numèrics per proporcionar les hores de vida de la bateria i el tractament de l'estatus de la bateria. En aquest últim cas, s'ha definit una estructura *case* per mostrar l'estat corresponent al codi numèric que proporciona la llibreria BatteryManager.

A banda de l'obtenció i posterior tractament de les dades, es defineix el comportament del botó buttonTancarP2. En aquest cas, en clicar el botó, es tancarà *l'Activity* i es mostrarà la pantalla inicial del menú, la qual estava en segon pla.

8.6. Finestra llistat de processos i aplicacions actius



En aquesta tercera pantalla es mostra el mateix encapçalament que a la primera i la segona finestra (icona aplicació, nom i imatge identificativa del TFC). Després d'aquest encapçalament identificatiu de l'aplicació, es mostra el títol de la finestra “Llistat de processos i aplicacions actius”. A continuació es pot visualitzar el llistat d'aplicacions en ús. Com a mínim, n’han d'aparèixer dues, l'aplicació battery i el launcher. El llistat següent reflecteix tots els processos que hi ha en memòria. En aquest cas, el llistat és més extens perquè el SO Android utilitza processos per anar sincronitzant dades i mantindre el SO disponible per a qualsevol demanda de l'usuari.

Aquesta tercera pantalla s'ha definit en tres fitxers:

/src/com/uoc/dcervero/android3/battery/Processos.java

/res/layout/processos.xml

/res/values/strings.xml

Tots els elements gràfics propis d'aquesta pantalla son del tipus Text*View*, excepte el botó final per tornar al menú principal. El títol s'ha formatat més gran. Els Text*View* utilitzat per encapçalar els llistats s'han formatat amb ombrejat blau, i els elements llistats (aplicacions i processos) s'han formatat en lletra estàndard. Les variables (string.xml) que alimenten aquest Text*View* es recuperen al fitxer Processos.java i es reomplen amb les dades obtingudes de l'ús de la llibreria android.app.ActivityManager. La resta d'elements gràfics són comuns a la pantalla principal (icona i imatge identificativa del TFC) o a la segona pantalla (botó de tornar al menú inicial).

Tots els textos que apareixen per pantalla es troben al fitxer /res/values/strings.xml i cada text està localitzat a una variable.

Pel que fa a la lògica de la pantalla, està definida al fitxer Processos.java. En aquest fitxer, mitjançant la llibreria ActivityManager, s'obtenen dos llistats:

```
List<RunningAppProcessInfo> procInfos = ActivityManager.getRunningAppProcesses();
List<ActivityManager.RunningTaskInfo> allTasks = ActivityManager.getRunningTasks(30);
```

Aquests dos llistats contenen el total d'aplicacions obertes per al cas de la variable procInfos i el total de processos oberts per la variable allTasks. Per tractar els dos llistats, primer es va recorrent la llista i s'extreu una dada (aplicació o operació) per cada iteració. Aquesta dada alimenta la variable corresponent al Text*View,* que mostrarà el llistat corresponent per pantalla. Cal destacar que per poder fer ús de la llibreria ActivityManager o, millor dit, per poder obtindre les dades que aquesta llibreria aporta, cal donar a l'aplicació els permisos pertinents. Per aquest motiu ha estat necessari afegir al fitxer AndroidManifest.xml l'ordre: següent

```
<uses-permission android:name="android.permission.GET_TASKS"/>
```

Aquesta sentència habilita que les crides .getRunningappProcesses i .getRunningTasks puguin rebre la informació pertinent del SO.

A banda de l'obtenció i posterior tractament de les dades, es defineix el comportament del botó buttonTancarP3. En aquest cas, en clicar el botó, es tancarà *l'Activity* i es mostrarà la pantalla inicial del menú, la qual estava en segon pla.

8.7. Llistat de fitxers

Totes els *strings* que es visualitzen per pantalla es troben al fitxer
/res/values/strings.xml

Totes les imatges que es fan servir es localitzen a path:
/res/drawable-hdpi
/res/drawable-mdpi
/res/drawable-ldpi

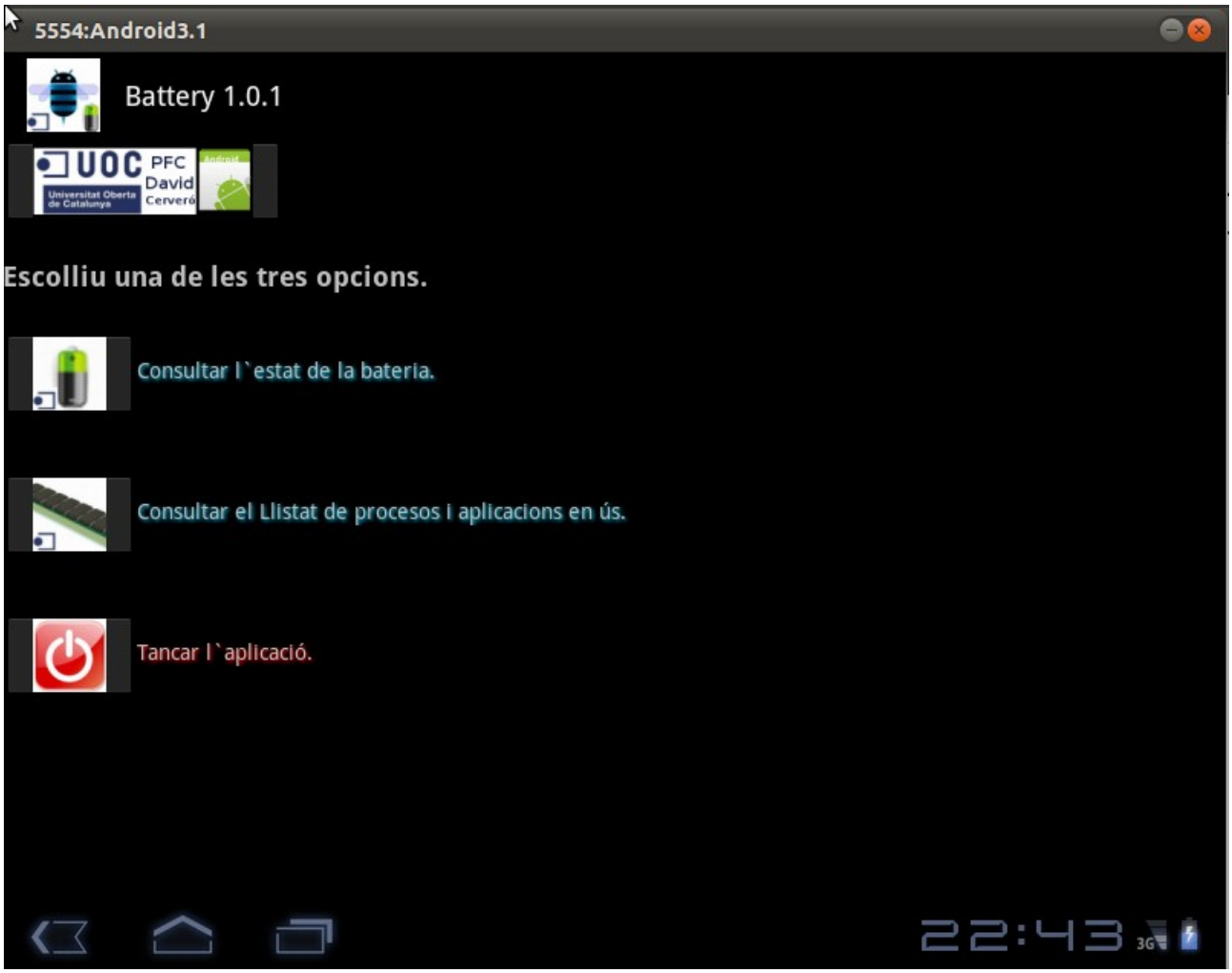
La definició gràfica de les diferents pantalles o *layouts* es localitza a:
/res/layout/main.xml
/res/layout/estat.xml
/res/layout/processos.xml

El comportament de cada pantalla està definit als fitxers:
/src/com/uoc/dcervero/android3/battery/Window.java
/src/com/uoc/dcervero/android3/battery/Estat.java
/src/com/uoc/dcervero/android3/battery/Processos.java

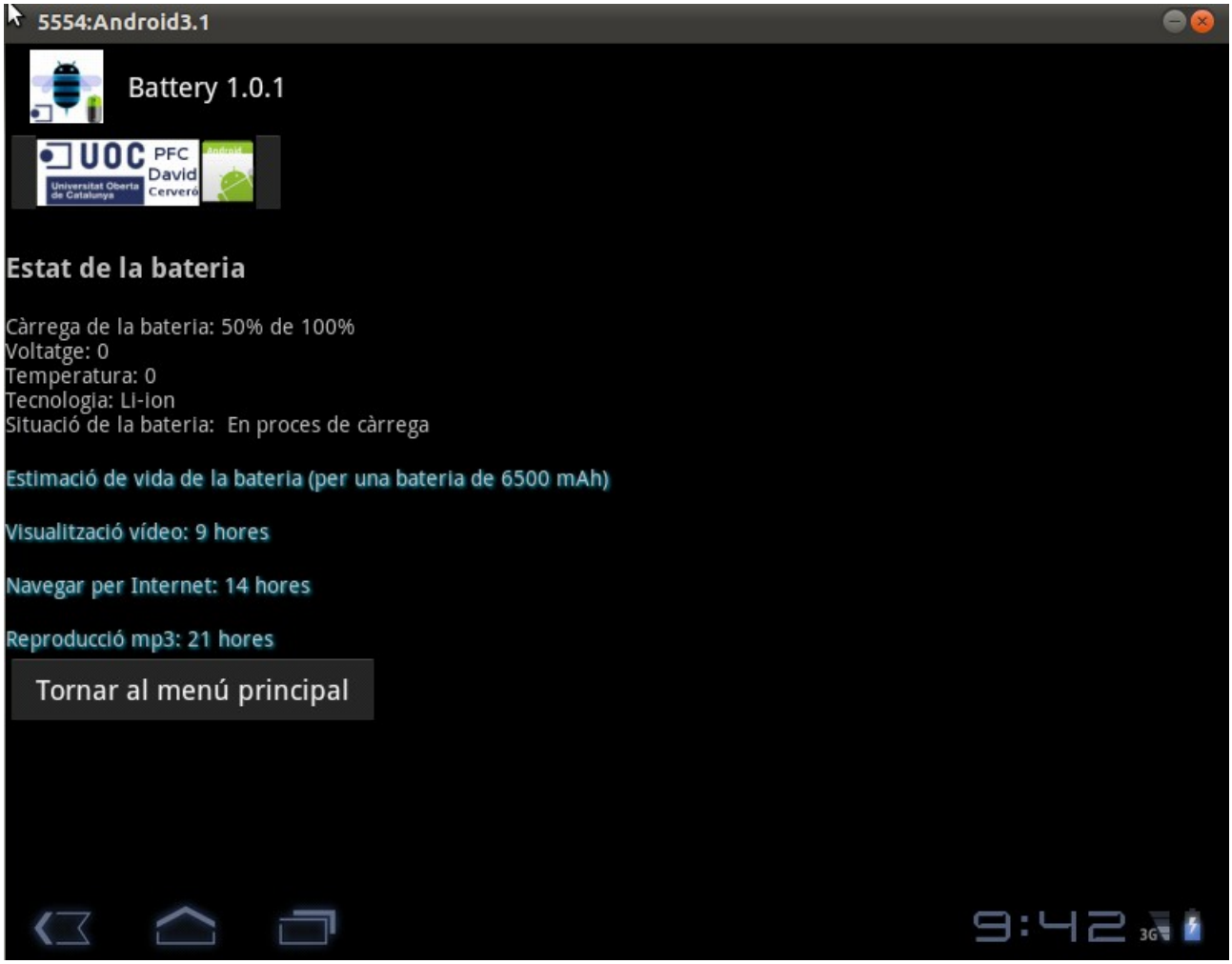
La declaració general de les activities implicades i els permisos necessaris està definit al fitxer:
/AndroidManifest.xml

8.8. Captures de pantalla de l'aplicació

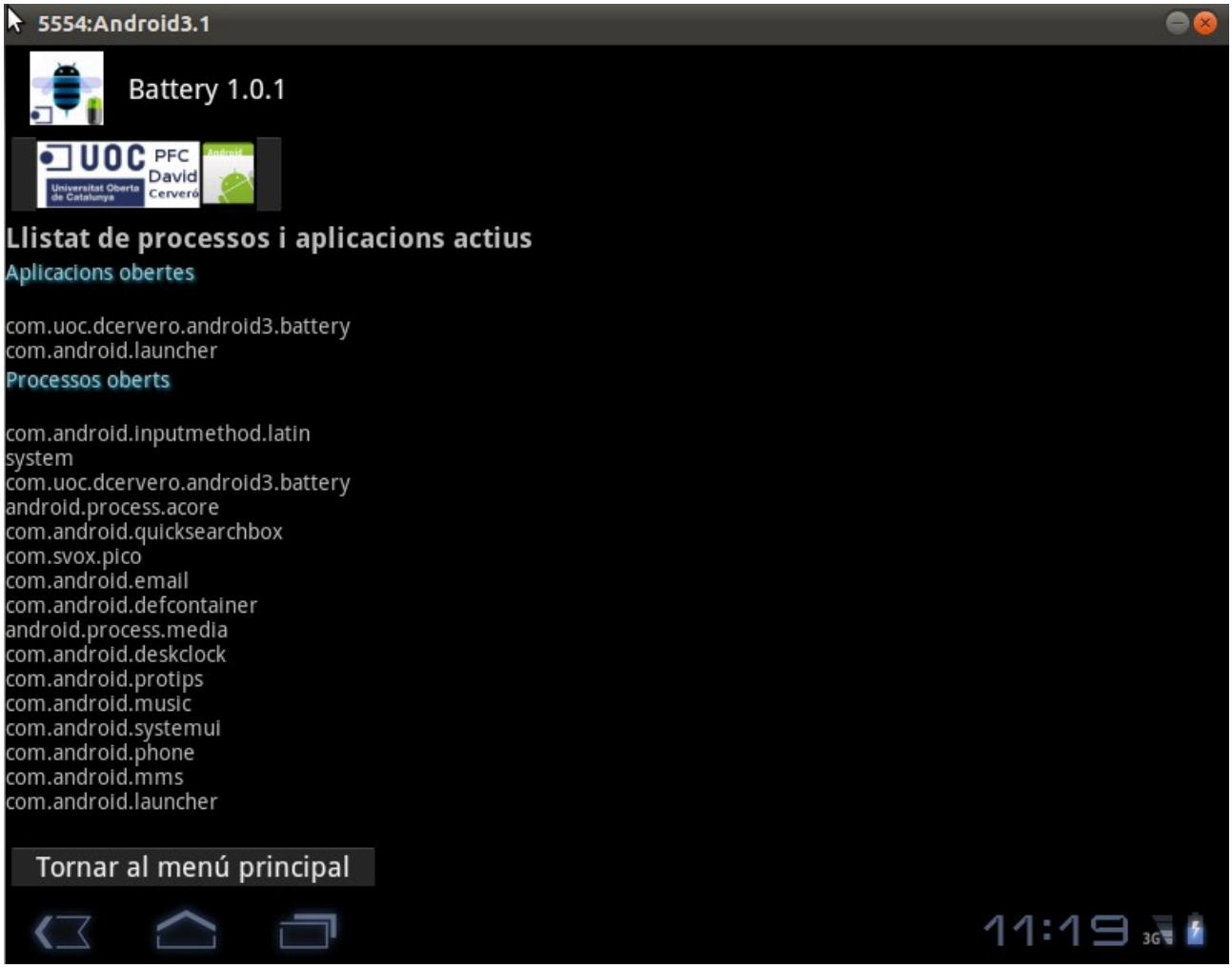
Finestra inicial.



Finestra d'informació sobre la bateria.



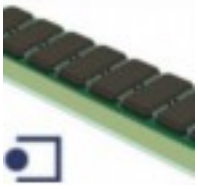
Finestra sobre el llistat de processos.



Icones .

Icona de l'aplicació

Botó accés estat bateria



Botó accés llistat processos

Icona del tfc

Botó de sortida

8.9. Incidències en el desenvolupament de l'aplicació

Per a la consecució d'aquest TFC, s'ha hagut de fer front a tres grans inconvenients. El primer es deriva de treballar amb l'AVD que proporciona el SDK Android. Treballar amb dispositius virtuals aporta molts avantatges, com la despreocupació pels diferents tipus de maquinari final, la immediatesa en poder provar noves implementacions de l'aplicació a desenvolupar o fins i tot la tranquil·litat de poder provar implementacions sense risc de fer mal bé el nostre *tablet*. Però a l'hora de treballar amb les dades provinents de la bateria, tota la informació obtinguda és irreal, ja que l'AVD no té una bateria real. Per aquest motiu alguna de les respostes de la llibreria BatteryManager és falsa o zero. El treball estadístic del consum energètic tampoc s'ha pogut treballar a l'AVD, no té cap sentit, ja que la bateria virtual mai s'esgota.

Un altre fet a remarcar és que per calcular el temps de vida de la bateria s'han hagut d'utilitzar càlculs estadístics. Tota aquesta informació es basa en una de les conferències de la I/O de Google que es va fer l'any 2009, en un article-web de Juan de Dios Maldonado (http://knol.google.com/k/android-programando-para-la-vida-de-la-batería) sobre aquesta conferencia, i finalment sobre la informació que es mostra al website de l'aplicació Powertutor (http://ziyang.eecs.umich.edu/projects/powertutor/documentation.html). El pitjor de tot és que aquesta informació trobada sobre el consum de la bateria per part de les aplicacions es refereix a la bateria d'un *smartphone*. Tot i que s'ha fet l'extrapolació a les característiques d'un *tablet*, és molt probable que existeixin derivacions importants en aquests càlculs, a causa de les diferents característiques entre un *smartphone* i un *tablet*.

El tercer inconvenient és el no-desenvolupament de la prestació de tancar aplicacions obertes. Les llibreries Android permeten el tancament d'aplicacions mitjançant la llibreria android.os i el mètode killProcess(PID), però no ha estat possible fer un estudi prou acurat per tal de desenvolupar aquestes prestacions.

Abans de començar la resolució d'aquest TFC semblava que seria molt més fàcil poder obtindre totes les dades necessàries de la bateria per proporcionar-les a l'usuari en format de característiques i estimació de temps d'ús, però un cop finalitzat, queda clar que tot i

poder obtindre algunes dades, per poder fer prediccions de la vida de la bateria cal fer un estudi molt més al detall. Curiosament en començar la part de l'aplicació del TFC que havia de treballar amb els processos i aplicacions obertes, semblava força més complexa. El TFC final demostra que les llibreries Android aporten totes les eines per poder obtindre la informació i fins i tot haver desenvolupat el tancament d'algunes aplicacions.

8.10. Possibles millores de l'aplicació

Derivades dels inconvenients trobats a l'hora de resoldre el TFC, s'hi proposen tres millores:

Treballar amb *tablets* reals enfront d'AVD

Com l'aplicació, treballa directament sobre un element de hardware, en aquest cas, la bateria. Seria molt recomanable poder treballar i fer tests de l'aplicació sobre un *tablet* real, i fins i tot poder contrastar el resultats amb altres models de *tablets*.

Estudi real del consum de la bateria

No s'han trobat dades ni estudis previs sobre el consum de les aplicacions en dispositius *tablets* amb SO Android. Per aquest motiu resulta molt recomanable plantejar un estudi a fons del consum de les aplicacions. Es podrien utilitzar aplicacions de tercers, com *PowerTutor* o *JuiceDefender* i caldria realitzar l'estudi en diferents *tablets*, amb diferents prestacions per poder publicar un estudi que servís de referència de cara a futurs treballs sobre la bateria dels *tablets*. A banda de fer servir aplicacions de tercers, la millor opció seria l'estudi directe sobre la bateria mitjançant equipament de mesura elèctrica.

Segon desenvolupament de l'aplicació

Es recomana desenvolupar el tancament de les aplicacions obertes. Aquesta implementació és viable gràcies a la llibreria android.os i el mètode killProcess(PID).

9. Fitxers

/res/value/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Battery 1.0.1</string>
    <string name="hello">\nEscolliu una de les tres opcions.\n</string>
    <string name="linia_vuida">\n </string>
    <string name="P1_text1">\nConsultar l`estat de la bateria.</string>
    <string name="P1_text2">\nConsultar el Llistat de procesos i
aplicacions en ús.</string>
    <string name="P1_text3">\nTancar l`aplicació.</string>
    <string name="P2_titol">\nEstat de la bateria</string>
    <string name="text_tornar">Tornar al menú principal</string>
    <string name="P2_text1">\nCàrrega de la bateria </string>
    <string name="P2_text2">\nEstimació de vida de la bateria (per una
bateria de 6500 mAh)</string>
    <string name="P2_text3">\nVisualització vídeo</string>
    <string name="P2_text4">\nNavegar per Internet</string>
    <string name="P2_text5">\nReproducció mp3</string>
    <string name="P2_text6">\nJugar\n</string>
    <string name="P3_text1">Llistat de processos i aplicacions
actius</string>
    <string name="P3_text2">Aplicacions obertes</string>
    <string name="P3_aplicacions"></string>
    <string name="P3_text3">Processos oberts</string>
    <string name="P3_processos"></string>
</resources>
```

/res/layout/main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<ImageButton
      android:layout_height="wrap_content"
    android:id="@+id/imageButton1"
    android:layout_width="wrap_content"
    android:src="@drawable/icontfc">
</ImageButton>
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    android:textStyle="bold"
    android:textSize="18sp"
    />
<LinearLayout
      android:id="@+id/linearLayout1"
      android:layout_height="wrap_content"
```

```
      android:layout_width="match_parent">
    <ImageButton
      android:id="@+id/buttonEstatBateria"
      android:src="@drawable/bateria"
            android:layout_width="wrap_content"
      android:layout_height="wrap_content">
    </ImageButton>
      <TextView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:text="@string/P1_text1"
      android:shadowColor="#00ccff"
        android:shadowRadius="1.5"
        android:shadowDx="1"
        android:shadowDy="1" />
</LinearLayout>
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/linia_vuida"
    />
<LinearLayout
      android:id="@+id/linearLayout1"
      android:layout_height="wrap_content"
      android:layout_width="match_parent">
    <ImageButton
      android:id="@+id/buttonLlistaProcesos"
      android:src="@drawable/proces"
            android:layout_width="wrap_content"
      android:layout_height="wrap_content">
    </ImageButton>
      <TextView
      android:layout_height="wrap_content"
      android:text="@string/P1_text2"
      android:layout_width="fill_parent"
      android:shadowColor="#00ccff"
      android:shadowRadius="1.5"
      android:shadowDx="1"
      android:shadowDy="1" />
</LinearLayout>
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/linia_vuida"
    />
<LinearLayout
      android:id="@+id/linearLayout1"
      android:layout_height="wrap_content"
      android:layout_width="match_parent">
    <ImageButton
      android:id="@+id/buttonExit"
      android:src="@drawable/exit"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content">
    </ImageButton>
    <TextView
      android:layout_width="fill_parent"
```

```
      android:layout_height="wrap_content"
      android:text="@string/P1_text3"
      android:shadowColor="#ff0000"
      android:shadowRadius="1.5"
      android:shadowDx="1"
      android:shadowDy="1"
      />
</LinearLayout>
</LinearLayout>
```

/res/layout/estat.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
 <ImageButton
    android:layout_height="wrap_content"
    android:id="@+id/imageButton1"
    android:layout_width="wrap_content"
    android:src="@drawable/icontfc">
 </ImageButton>
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P2_titol"
    android:textStyle="bold"
    android:textSize="18sp"
    />
<TextView
    android:id="@+id/P2_text1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P2_text1"
    />
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P2_text2"
    android:shadowColor="#00ccff"
    android:shadowRadius="1.5"
    android:shadowDx="1"
    android:shadowDy="1"
    />
<TextView  android:id="@+id/P2_text3"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P2_text3"
    android:shadowColor="#00ccff"
    android:shadowRadius="1.5"
    android:shadowDx="1"
    android:shadowDy="1"
```

```
    />
<TextView
    android:id="@+id/P2_text4"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P2_text4"
    android:shadowColor="#00ccff"
    android:shadowRadius="1.5"
    android:shadowDx="1"
    android:shadowDy="1"
    />
<TextView
    android:id="@+id/P2_text5"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P2_text5"
    android:shadowColor="#00ccff"
    android:shadowRadius="1.5"
    android:shadowDx="1"
    android:shadowDy="1"
    />
<LinearLayout
      android:id="@+id/linearLayout1"
      android:layout_height="wrap_content"
      android:layout_width="match_parent">
    <Button
            android:text="@string/text_tornar"
            android:id="@+id/buttonTancarP2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
    </Button>
</LinearLayout>
</LinearLayout>
```

/res/layout/processos.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
    <ImageButton
    android:layout_height="wrap_content"
    android:id="@+id/imageButton1"
    android:layout_width="wrap_content"
    android:src="@drawable/icontfc">
</ImageButton>
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P3_text1"
    android:textStyle="bold"
    android:textSize="18sp"
```

```
    />
<TextView
    android:id="@+id/P3_text2"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P3_text2"
    android:shadowColor="#00ccff"
    android:shadowRadius="1.5"
    android:shadowDx="1"
    android:shadowDy="1"
    />
    <TextView
    android:id="@+id/P3_aplicacions"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P3_aplicacions"
    />
<TextView
    android:id="@+id/P3_text3"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P3_text3"
    android:shadowColor="#00ccff"
    android:shadowRadius="1.5"
    android:shadowDx="1"
    android:shadowDy="1"
    />
 <TextView
    android:id="@+id/P3_processos"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/P3_processos"
    />
<LinearLayout
      android:id="@+id/linearLayout1"
      android:layout_height="wrap_content"
      android:layout_width="match_parent">
    <Button
      android:id="@+id/buttonTancarP3"
      android:text="@string/text_tornar"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content">
    </Button>
</LinearLayout>
</LinearLayout>
```

/src/com/uoc/dcervero/android3/battery/Window.java

```
package com.uoc.dcervero.android3.battery;
/* Treball Fi de Carrera de Ingenieria en Informàtica
 *
 * Universitat Oberta de Catalunya
 * Alumne: David Cerveró Aured
 * dcervero@uoc.edu
 * juny 2011
 *
 * */

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.View.View;
import android.View.View.OnClickListener;
import android.widget.ImageButton;

public class Window extends Activity {
    /** Called when the Activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        /* Butó per accedir a la finestra d'estat de la bateria  */
                                final    ImageButton    buttonEstatBateria    =
(ImageButton)findViewById(R.id.buttonEstatBateria);
        buttonEstatBateria.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View v) {
      /* En clicar, es llança una nova activitat per mostrar les opcions
       * en referencia a la bateria  */
                        Intent    intent    =    new    Intent(Window.this,
Estat.class);
                      startActivity(intent);
                  }
          });

     /*  Butó  per  accedir  a  la  finestra  de  llistat  de  processos  i
aplicacions  */
                              final    ImageButton    buttonLlistaProcesos    =
(ImageButton)findViewById(R.id.buttonLlistaProcesos);
        buttonLlistaProcesos.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View v) {
      /* En clicar, es llança una nova activitat per mostrar les opcions
       * en referencia als llistat de processos i aplicacions */
                        Intent    intent    =    new    Intent(Window.this,
Processos.class);
                      startActivity(intent);
                  }
          });

        /* Butó per tancar l'aplicació  */
                                    final    ImageButton    buttonExit    =
```

```
(ImageButton)findViewById(R.id.buttonExit);
        buttonExit.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View v) {
                        /* En clicar, es tanca l'Activity  */
                        finish();
                  }
          });
    }
}
```

/src/com/uoc/dcervero/android3/battery/Estat.java

```
package com.uoc.dcervero.android3.battery;
/* Treball Fi de Carrera de Ingenieria en InformÃ tica
 *
 * Universitat Oberta de Catalunya
 * Alumne: David CerverÃ³ Aured
 * dcervero@uoc.edu
 * juny 2011
 *
 * */

import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.BatteryManager;
import android.os.Bundle;
import android.View.View;
import android.View.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class Estat extends Activity {
    /** Called when the Activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.estat);

        /* Es llanÃ§a un broadcast per recollir les dades */
            BroadcastReceiver batteryReceiver = new BroadcastReceiver() {
                int scale = -1;
                int level = -1;
                int voltage = -1;
                int temp = -1;
                int status = -1;

                @Override
                public void onReceive(Context context, Intent intent) {
                  /* En rebre el broadcast, es recullen les dades
                   * relatives a la bateria */
                    level =
intent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
```

```
                    scale =
intent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);
                    temp =
intent.getIntExtra(BatteryManager.EXTRA_TEMPERATURE, -1);
                    voltage =
intent.getIntExtra(BatteryManager.EXTRA_VOLTAGE, -1);
                    status =
intent.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
                    String tecnologia =
intent.getStringExtra(BatteryManager.EXTRA_TECHNOLOGY);
                    /* Un cop es tenen les dades, es recuperen les
variables de text
                     * per poder reomplirles amb les dades */
                    final TextView P2_text1 =
(TextView)findViewById(R.id.P2_text1);
                    String texto = P2_text1.getText().toString();
                    texto += ": ";
                    texto += level;
                    texto += "% de ";
                    texto += scale;
                    texto += "% \nVoltatge: ";
                    texto += voltage;
                    texto += " \nTemperatura: ";
                    texto += temp;
                    texto += " \nTecnologia: ";
                    texto += tecnologia;
                    texto += " \nSituaciÃ³ de la bateria: ";
                    switch(status){
                        case BatteryManager.BATTERY_STATUS_UNKNOWN:
                              texto += " No hi ha bateria !!!";
                              break;
                        case BatteryManager.BATTERY_STATUS_CHARGING:
                              texto += " En proces de cÃ rrega";
                              break;
                        case BatteryManager.BATTERY_STATUS_DISCHARGING:
                              texto += " Descarregant-se !!!";
                              break;
                        case BatteryManager.BATTERY_STATUS_NOT_CHARGING:
                              texto += " En Ãºs";
                              break;
                        case BatteryManager.BATTERY_STATUS_FULL:
                              texto += " Carregada completament (100%)";
                              break;
                        default:
                              texto += " Desconegut !!! Possible
avaria !!!";
                              break;
                    }
                    P2_text1.setText(texto);

                    final TextView P2_text3 =
(TextView)findViewById(R.id.P2_text3);
                    String tempsvideo = P2_text3.getText().toString();
                    tempsvideo += ": ";
                    int tempsvideonum = (6500/340)/(100/level);
                    tempsvideo += tempsvideonum;
                    tempsvideo += " hores";
```

```
                    P2_text3.setText(tempsvideo);

                    final TextView P2_text4 =
(TextView)findViewById(R.id.P2_text4);
                    String tempsinternet = P2_text4.getText().toString();
                    tempsinternet += ": ";
                    int tempsinternetnum = (6500/225)/(100/level);
                    tempsinternet += tempsinternetnum;
                    tempsinternet += " hores";
                    P2_text4.setText(tempsinternet);

                    final TextView P2_text5 =
(TextView)findViewById(R.id.P2_text5);
                    String tempsmp3 = P2_text5.getText().toString();
                    tempsmp3 += ": ";
                    int tempsmp3num = (6500/150)/(100/level);
                    tempsmp3 += tempsmp3num;
                    tempsmp3 += " hores";
                    P2_text5.setText(tempsmp3);
                }

            };
            IntentFilter filter = new
IntentFilter(Intent.ACTION_BATTERY_CHANGED);
            registerReceiver(batteryReceiver, filter);

            /* ButÃ³ per tancar l'aplicaciÃ³  */
        final Button buttonTancarP2 =
(Button)findViewById(R.id.buttonTancarP2);
        buttonTancarP2.setOnClickListener(new OnClickListener() {
                @Override
                /* En clicar es tanca l'Activity  */
                public void onClick(View v) {
                      finish();
                }
          });
    }
}
```

/src/com/uoc/dcervero/android3/battery/Processos.java

```
package com.uoc.dcervero.android3.battery;
/* Treball Fi de Carrera de Ingenieria en InformÃ tica
 *
 * Universitat Oberta de Catalunya
 * Alumne: David CerverÃ³ Aured
 * dcervero@uoc.edu
 * juny 2011
 *
 * */


import java.util.List;

import android.app.Activity;
import android.app.ActivityManager;
import android.app.ActivityManager.RunningAppProcessInfo;
import android.os.Bundle;
import android.View.View;
import android.View.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class Processos extends Activity {
    /** Called when the Activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.processos);

        /* Es recullen les dades del sistema */
        ActivityManager ActivityManager =
(ActivityManager)this.getSystemService(Activity_SERVICE);
        /* De totes les dades del sistema s'obtenen dos llistats
         * un d'aplicacions
         * i un de processos */
        List<RunningAppProcessInfo>  procInfos =
ActivityManager.getRunningAppProcesses();
        List<ActivityManager.RunningTaskInfo> allTasks =
ActivityManager.getRunningTasks(30);
        /* Es recuperen les variables de text que s'aniran omplint amb
els diferents elements
         * de les llistes obtingudes
         * Per fer aquesta introducciÃ³, es tracta cada llista de manera
         * que en cada iteracÃ³ s'afegeix un elemnt */
        final TextView P3_aplicacions =
(TextView)findViewById(R.id.P3_aplicacions);
        String texto = P3_aplicacions.getText().toString();
        for (int x = 0; x < allTasks.size(); x++)
        {
            texto +=" \n";
            String pckName=allTasks.get(x).baseActivity.getPackageName();
            texto += pckName;
        }
        P3_aplicacions.setText(texto);
        final TextView P3_processos =
```

```
(TextView)findViewById(R.id.P3_processos);
        String texto3 = P3_processos.getText().toString();
        texto3 += " \n";
        for(int i = 0; i < procInfos.size(); i++)
        {
            texto3 += procInfos.get(i).processName;
            texto3 +=" \n";
        }
        P3_processos.setText(texto3);
        final Button buttonTancarP3 =
(Button)findViewById(R.id.buttonTancarP3);
        /* ButÃ³ per tancar l'aplicaciÃ³  */
        buttonTancarP3.setOnClickListener(new OnClickListener()
        /* En clicar es tanca l'Activity  */
        { @Override public void onClick(View v) { finish(); } });

    }
```

/AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.uoc.dcervero.android3.battery"
      android:versionCode="1"
      android:versionName="1.0">
    <uses-sdk android:minSdkVersion="11" />
    <uses-permission android:name="android.permission.GET_TASKS"/>
    <application android:icon="@drawable/icon"
                    android:label="@string/app_name">
        <Activity android:name=".Window"
                  android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </Activity>
            <Activity android:name=".Estat"/>
            <Activity android:name=".Processos"/>
    </application>
</manifest>
```

10. Annex

Com a annexos a aquest TFC s'adjunten els documents següents:

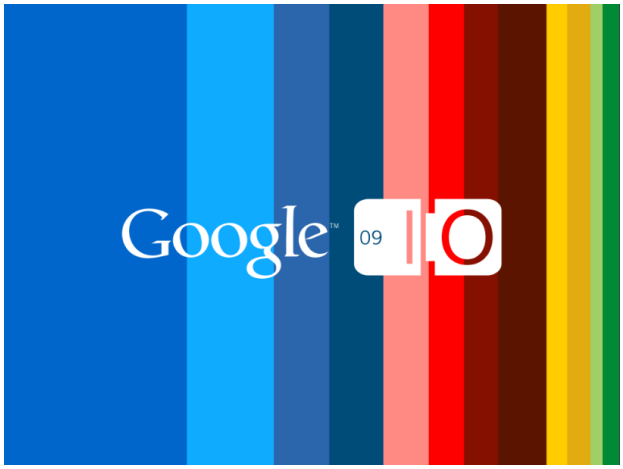
Annex A

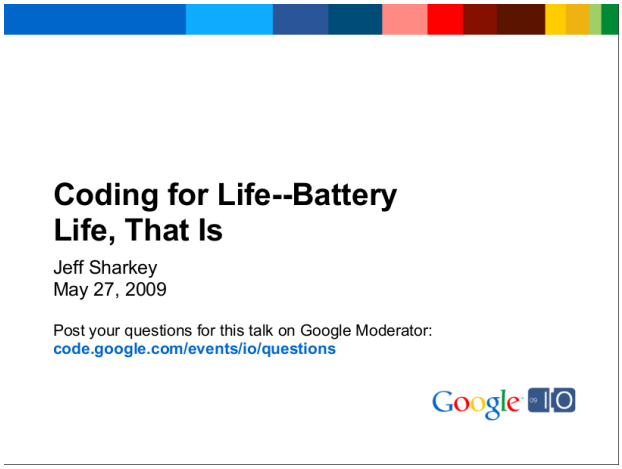
Document resum de la conferència del dia 27 de maig de 2009, al congrés anual Google I/O 2009

W_0300_CodingforLife-BatteryLifeThatIs.pdf

http://www.google.com/events/io/2009/sessions/CodingLifeBatteryLife.html

http://www.youtube.com/watch?v=OUemfrKe65c





Annex B

Web on s'analitza la conferència de l'Annex A

http://knol.google.com/k/android-programando-para-la-vida-de-la-bater%C3%ADa#

Annex C

Document sobre l'estudi del consum de les bateries d'*smartphones* amb SO Android. Documentació del programa PowerTutor

camera-ready.pdf

http://ziyang.eecs.umich.edu/projects/powertutor/documentation.html

Annex D

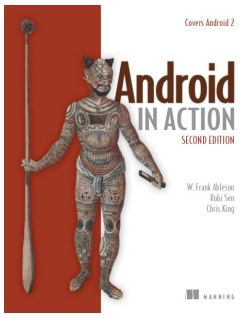
Llistat de característiques de *tablets*, utilitzat per calcular les prestacions d'una bateria model d'un *tablet*.

Tablets.pdf

11. Bibliografia

Llibres consultats

Andbook
Android programmings
Nicolas Gramlich
anddev.org-Community



Android in action Second edition
Frank Ableson
Robi Sen
Chris King
2011

Professional Android 2
Applications Development
Reto Meier
2010

TabletWord
The complete Guide to Tablets

Webs consultades

Android

http://developer.android.com/reference/packages.html

http://developer.android.com/reference/android/os/BatteryManager.html

http://developer.android.com/reference/android/app/package-summary.html

Apple

www.apple.com/es/ipad/specs/

www.apple.com/es/ipad/ios4/

Wikipedia

es.wikipedia.org/wiki/Android

es.wikipedia.org/wiki/ipad

es.wikipedia.org/wiki/apple

es.wikipedia.org/wiki/Apple_iOS

Google

http://www.google.com/events/io/2009/

http://www.google.com/events/io/2009/sessions/CodingLifeBatteryLife.html

http://www.google.com/events/io/2010/

http://www.google.com/events/io/2011/index-live.html

Xataka

www.xatakandroid.com

www.xataka.com

http://www.xatakandroid.com/xatakandroid/android-le-quita-el-puesto-a-blackberry-por-primera-vez-a-nivel-mundial

EL PAIS

http://www.elpais.com/tecnologia/

http://www.elpais.com/articulo/tecnologia/Temor/reviente/burbuja/tabletas/elpeputec/20110310elpeputec_1/Tes

ALTRES FONTS

1 www.androidsis.com

2 http://www.webadictos.com.mx/2011/02/05/tabla-comparativa-de-las-mejores-tablets-en-el-mercado/

3 http://www.genbeta.com/a-fondo/especial-sistemas-operativos-para-tablets-blackberry-tablet-os-el-motor-del-playbook

4 http://tabletpccomparison.com/

5 http://www.configurarequipos.com/doc1107.html

6 http://www.genbeta.com/actualidad/Google-presenta-oficialmente-los-detalles-de-android-30-honeycomb

7 http://www.muymovil.com/2011/01/27/caracteristicas-completas-de-android-3-0-honeycomb

8 www.edu4java.com

9 www.sgoliver.net/

10 www.android-spa.com/

11 www.elandroidelibre.com/

12 http://craftyman.net/

13 http://www.androidcentral.com/

14 http://yoandroid.com/

15 http://www.mail-archive.com/android-developers@Googlegroups.com/info.html

16 http://knol.google.com/k/android-programando-para-la-vida-de-la-bater%C3%ADa#

17 http://www.movilzona.es/2011/02/09/batteryview-ten-bajo-control-el-consumo-de-bateria/

18 http://ziyang.eecs.umich.edu/projects/powertutor/documentation.html

[i][TIC] Tecnologies de la Informació i la Comunicació

[ii] [hosting] Lloguer d'espai per ubicar un web o altres elements que es vulguin oferir a Internet.

[iii] [Apple] Empresa tecnològica americana. Els seus productes son els ordinadors Macintosh amb SO propietari, i altres elements com el iPod, iPhone o iPad. Destaca pel seu disseny i capacitat d'innovació. El seu president és Steve Jobs http://www.apple.com

[iv][Ipad] *tablet* de Apple. Es pot considerar com el primer *tablet* que va aparèixer al mercat. Vídeo de presentació del Ipad http://www.youtube.com/watch?v=yU6isGR3PaM

[v][iOS] Sistema operatiu de Apple per als dispositius iPhone, iPod i iPad. Versió actual 4.3

[vi][Samsung Galaxy Tab 10] *tablet* de Samsung. En aquests moments representa la competència directa al Ipad de Apple. http://galaxytab.samsungmobile.com/10.1/features.html

[vii][Android] SO per smatphones i *tablets*. Propietat de Google, que va comprar l'empresa Android Inc l'any 2005. Google va alliberar la majoria del codi d'Android i permet la lliure programació d'aplicacions per aquest SO. http://www.android.com/

[viii] [Linux] SO desenvolupat per Linus Torvalds l'any 1991 com una evolució de Unix. És de codi obert i existeixen diverses distribucions que competeixen amb el SO Windows de Microsoft. Una de les distribucions més esteses a nivell d'usuari és Ubuntu.

[ix][*smartphones*] Telèfons mòbils, que ofereixen connectivitat i interacció amb els PC

[x] [Google] Motor de cerca d'us majoritari, dissenyat i presidit per Larry Page i Sergey Brin. A més del motor de cerca, ha desenvolupat altres aplicacions, com el correu Gmail, GoogleDocs, GoogleEarth, navegador Chrome, Google Calendar, accionista majoritari de Mozilla Firefox. Va comprar youtube i Android.

[xi][Android SDK] Conjunt d'eines per poder desenvolupar aplicacions per Android. http://developer.android.com/sdk/index.html

[xii][Android Market] Plataforma que ofereix Android per distribuir aplicacions per aquest SO. https://market.android.com/

[xiii]http://www.iphonebatteryoptimizer.com/

[xiv] [iPod]Mp3 d'Apple http://www.apple.com/es/ipod/

[xv] [Pen stylus]Punter en forma de bolígraf que s'utilitza per interactuar amb pantalles tàctils

[xvi] [Capacintància] Propietat que tenen els cosos per mantenir una carga elèctrica

[xvii] [Memòria flash] Memòria d'emmagatzemament persistent basada en xip enlloc de dispositius magneto mecànics.

[xviii] [Memòria SD] Memòria d'emmagatzemament persistent extraïble, freqüentment utilitzada per transportar documents digitals. És la opció mes estesa entre els diferents formats disponibles al mercat-

[xix] [Memory stick] Equivalent a la memòria SD però propietat de Sony

[xx] [USB] Universal Serial Bus. Interficie de comunicació dissenyada per proporcionar connectivitat a multitud de hardware. En aquests moments s'ha llençat la versió 3.0

[xxi] [Pendrive] Memòria flash dissenyada per connectar-se per USB

[xxii] [HDMI] Interfície multimèdia per connectar senyal de vídeo i àudio.

[xxiii] [minijack estereo] Connexió habitual d'àudio

[xxiv] [ethernet] Connexió de xarxa per cable, habitualment per xarxes LAN.

[xxv] [Wifi] Connexió per xarxa sense fils

[xxvi] [3G] Transmissió de veu i dades per telefonia mòbil

[xxvii] [4G]Evolució de la telefonia 3G, aporta més velocitat i mes capacitat de transferència de veu i dades

[xxviii] [Tendències mercat SO per *tablets* 2012]http://www.androidsis.com/las-*tablets*-con-android-abarcaran-el-39-del-mercado-en-2012/

[xxix] [HD]High Definition. Format de vídeo de gran definició (1280x720 o 1920x1080)

[xxx] [aPad]http://www.irobotgroup.com/

[xxxi] [tendendies mercat SO per tablets]http://gabatek.com/tecnologia/android-se-acerca-apple-ios-en-el-mercado-de-tablets-lideraran-claramente-en-2015/

[xxxii] [Palo Alto]Ciutat dels Estats Units.http://www.cityofpaloalto.org/

[xxxiii] [Andy Rubin]http://www.nytimes.com/2007/11/04/technology/04Google.html?_r=2&hp=&pagewanted=all

[xxxiv] [GooglePhone]http://www.Google.com/phone/#manufacturer=all&category=all&carrier=all&country=es&reset_filters=1

[xxxv] [Empreses que formen la Open handset Alliance]

http://www.openhandsetalliance.com/oha_members.html

[xxxvi] [Kernel] Nucli d'un SO. És el programari responsable de facilitar accés al maquinari.

[xxxvii] [Informe 2010 Android Market]http://www.elandroidelibre.com/2011/02/informe-del-android-market-y-sus-aplicaciones-en-2010.html

[xxxviii] [Eclipse] Entorn de desenvolupament per programació, de codi obert. Www.eclipse.org

[xxxix] [mAh]MiliAmper per hora. Unitat elèctrica per indicar la càrrega elèctrica que passa pels terminals d'una bateria, si aquesta proporciona un corrent elèctric de 1 amper a la hora.